BoxBackup setup (Debian/Ubuntu)

Ferry Boender

Revision History

Revision 1.2 Feb 19, 2017 Revised by: FB

1. Introduction

BoxBackup is an online remote backup tool for Unix systems (BSDs, Linux, MacOSX). It is robust, secure, low on resources and can perform both in continues backup mode and snapshotting. In continues backup mode changes will be pushed to the server soon after they happen; in snapshot mode mode BoxBackup behaves more like traditional backup programs and creates snapshots every fixed amount of time.

This article will describe how to set up a BoxBackup server and client on Debian and Ubuntu machines. Much of this article can also be used on other Unix-like systems, however it will not discuss how to compile BoxBackup.

1.1. Assumptions

This article assumes the following:

- The server (where backups will be stored) is called `server`. Its FQDN will be `server.example.com`.
- The client (which will be backed up) is called `client`. Its FQDN will be `client.example.com`.
- Backups will be stored in a directory `/storage/backup` on `server`.

2. For the lazy

This chapter is for the lazy. It quickly describes how to setup BoxBackup, but provides no in-depth information. Please note that commands are prefixed by a prompt that indicates on which machine and in which directory to run the command.

The server will allow the client 100Gb of storage with a hard limit of 120Gb. The client will have account ID `00000001` (more on account IDs later). BoxBackup's user-land RAID-like configuration will not be used. We will have to move some files between the `client` and the `server`. For this we use the `scp` program and the user account named `user` on both machines.

2.1. Server configuration

Install the server:

```
server:/root# aptitude install boxbackup-server
```

Delete Debian's generated configuration (We'll recreate it ourselves):

```
server:/root# rm -rf /etc/boxbackup/
```

Find out filesystem block size:

```
server:/root# /sbin/dumpe2fs -h /dev/sdb1 | grep 'Block size'
dumpe2fs 1.41.4 (27-Jan-2009)
Block size:               4096
```

Create RAID-like configuration file:

```
server:/root# raidfile-config  /etc/boxbackup/ 4096 /storage/
```

Create backup storage directory:

```
server:/root# mkdir /storage/backup/
server:/root# chown bbstored:bbstored /storage/backup
server:/root# chmod 750 /storage/backup
```

Create BoxBackup server configuration and certificates:

```
server:/root# bbstored-config /etc/boxbackup/ server.example.com bbstored
server:/root# bbstored-certs ca init
server:/root# bbstored-certs ca sign-server /etc/boxbackup/bbstored/server.example.com-csr.
server:/root# cp ca/servers/server.example.com-cert.pem /etc/boxbackup/bbstored
server:/root# cp ca/roots/clientCA.pem /etc/boxbackup/bbstored
server:/root# chown -R bbstored:bbstored /etc/boxbackup/
server:/root# chmod 700 /etc/boxbackup/
```

Start the server:

```
server:/root# /etc/init.d/boxbackup-server start
server:/root# grep "Box Backup" /var/log/syslog
Jan 31 21:45:13 server Box Backup (bbstored)[30775]: NOTICE: Starting daemon, version 0.11r
```

> **Important:** Keep the `/root/ca` directory around. It is required to sign new BoxBackup clients' certificate files.

2.2. Client configuration

Create an account on the server:

```
server:/root# bbstoreaccounts create 00000001 0 100g 120g
```

Install the client software:

```
client:/root# aptitude install boxbackup-client
client:/root# rm /etc/boxbackup/bbackupd.conf
client:/root# rm /etc/boxbackup/bbackupd/*
```

Create client configuration and certificates:

```
client:/root# bbackupd-config /etc/boxbackup/ lazy 00000001 server.example.com /var/run /va
client:/root# scp /etc/boxbackup/bbackupd/00000001-FileEncKeys.raw user@server.example.com:
client:/root# scp /etc/boxbackup/bbackupd/00000001-csr.pem user@server.example.com:
```

Sign certificates on the server:

```
server:/root# bbstored-certs ca sign /home/user/00000001-csr.pem
server:/root# scp ca/clients/00000001-cert.pem user@client.example.com:
server:/root# scp ca/roots/serverCA.pem user@client.example.com:
server:/root# /etc/init.d/boxbackup-server restart
```

Install signed certificates on the client and start client:

```
client:/root# mv /home/user/00000001-cert.pem /etc/boxbackup/bbackupd/
client:/root# mv /home/user/serverCA.pem /etc/boxbackup/bbackupd/
client:/root# /etc/init.d/boxbackup-client start
client:/root# grep "Box Backup" /var/log/syslog
Jan 31 21:55:13 client Box Backup (bbackupd)[20975]: NOTICE: Starting daemon, version 0.11r
Jan 31 21:55:13 client Box Backup (bbackupd)[20975]: NOTICE: Beginning scan of local files
```

> **Important:** Keep the `0000001-FileEncKeys.raw` file around. It is required to restore backups.

If anything fails, please take a look at the rest of this document. It explains in more detail how to setup BoxBackup and may provide clues as to the problems you are encountering.

3. Setting up the server (bbstored)

3.1. Installation

The first thing is setting up the server. This is where the backups will be stored. It will run the `bbstored` program, which is a server daemon that clients can connect to in order to upload data.

The server will be designated as `server` in the commands given in this chapter. Each command is prefixed with a shell-prompt to indicate on which machine (`client` or `server`) the command should be entered. The server's Fully Qualified Domain Name will be `server.example.com`. The client's will be `client.example.com`. We will have to move some files (certificates) around between the server and the client in order to sign them. For this we will use a user account with username `user`. It is assumed the root user on both systems can use secure copy (`scp`) to move files from one system to the other under this user account.

Let's get started by installing the BoxBackup server. We use Debian, but it should also work under Ubuntu. It is unknown to me which different versions of BoxBackup are compatible with each other, so I assume you'll be using the same version on both the client and the server. If you're not using Debian or Ubuntu, please check your distribution's package list to see if a pre-compiled version of BoxBackup is available. If not, you will have to compile it yourself. This is beyond the scope of this article, so check the BoxBackup documentation for instructions.

```
server:/root# aptitude install boxbackup-server
```

Debian/Ubuntu will install the BoxBackup server and some configuration tools. It will also create a user called `bbstored`. The BoxBackup server (`bbstored`) will run as this user.

Debian also might generate a configuration file and certificates for you. We will **delete** those because they confuse the process of installing and configuring BoxBackup (as what Debian does for you and what you need to do manually is terribly documented and it stores things in the wrong place).

```
server:/root# rm -rf /etc/boxbackup/
```

3.2. Creating a store

BoxBackup server stores backups in a `store`. We will configure this store at `/storage/backup`. This directory will contain a store for each client that backs up using the server. E.g. the backups for the client with account ID `00000001` (more on account IDs later on) will be stored in `/storage/backup/00000001/`.

BoxBackup provides a RAID-like setup where backups are written to multiple locations; usually different physical disks. This RAID-like setup **has nothing to do with actual RAID**, so don't get confused by that. This article will **not** be using the RAID-like facilities of BoxBackup.

Before we can create our store, we must first determine the block size of our file system . This can be done using the dumpe2fs tool. Provide the proper device name to the tool:

```
server:/root# mount | grep storage
/dev/sdb1 on /storage type ext3 (rw)
```

In this case the device is `/dev/sdb1`. Let's get the block size:

```
server:/root# /sbin/dumpe2fs -h /dev/sdb1 | grep 'Block size'
dumpe2fs 1.41.4 (27-Jan-2009)
Block size:               4096
```

As we can see, our block size is 4096. We can now configure a Boxbackup store.

```
server:/root# raidfile-config  /etc/boxbackup/ 4096 /storage/
WARNING: userland RAID is disabled.
Config file written.
```

If the BoxBackup user (`bbstored`) has write permissions to the `/storage/` directory, you can skip the following steps and continue with the `bbstored-config` step.

Now we create a directory for the backups and set the proper permissions:

```
server:/root# mkdir /storage/backup/
server:/root# chown bbstored:bbstored /storage/backup
server:/root# chmod 750 /storage/backup
```

3.3. Server configuration file / certificates

Time to create the bbstored configuration. We do this with the `bbstored-config` tool. It takes three parameters:

`config_dir`

> The **configuration location**. In our case, `/etc/boxbackup/`.

`server_name`

> The **FQDN of the server**. We use `server.example.com`.

`username`

> The **user which bbstored will run as**. Debian created the `bbstored` user for us, so we'll use that. It's the same as the daemon name, which is customary.

Run the command:

```
server:/root# bbstored-config /etc/boxbackup/ server.example.com  bbstored

Checking permissions on /storage//backup
Checking permissions on /storage//backup
Checking permissions on /storage//backup

Setup bbstored config utility.

Configuration:
   Writing configuration file: /etc/boxbackup//bbstored.conf
   Writing empty accounts file: /etc/boxbackup//bbstored/accounts.txt
   Server hostname: server.example.com
   RaidFile config: /etc/boxbackup//raidfile.conf

Creating blank accounts file
Generating private key...
Generating RSA private key, 2048 bit long modulus
................................................................
...........................................................................+++
............................................................+++
e is 65537 (0x10001)
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter ’.’, the field will be left blank.
-----
Country Name (2 letter code) [AU]:
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (eg, YOUR name) []:
Email Address []:

Please enter the following ’extra’ attributes
to be sent with your certificate request
A challenge password []:An optional company name []:

Writing configuration file /etc/boxbackup//bbstored.conf

===================================================================

bbstored basic configuration complete.

What you need to do now...

1) Sign /etc/boxbackup//bbstored/server.example.com-csr.pem
   using the bbstored-certs utility.

2) Install the server certificate and root CA certificate as
      /etc/boxbackup//bbstored/server.example.com-cert.pem
      /etc/boxbackup//bbstored/clientCA.pem

3) You may wish to read the configuration file
      /etc/boxbackup//bbstored.conf
   and adjust as appropraite.

4) Create accounts with bbstoreaccounts

5) Start the backup store daemon with the command
      /usr/local/bin/bbstored /etc/boxbackup//bbstored.conf
   in /etc/rc.local, or your local equivalent.

===================================================================
```

This will generate a configuration file at `/etc/boxbackup/bbstored.conf` and generate a server certificate sign request file at `/etc/boxbackup/bbstored/server.example.com-csr.pem`.

The next step is to create and sign the server certificate. For this we need to generate a root certificate first. We use the `bbstored-certs` tool to do this:

```
server:/root# bbstored-certs ca init

Generating RSA private key, 2048 bit long modulus
...................................................................
.................................................................+++
................................................+++
e is 65537 (0x10001)
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (eg, YOUR name) []:
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:An optional company name []:

Signature ok
subject=/CN=Backup system client root
Getting Private key
Generating RSA private key, 2048 bit long modulus
.........................+++
............................................+++
e is 65537 (0x10001)
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (eg, YOUR name) []:
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:An optional company name []:

Signature ok
subject=/CN=Backup system server root
Getting Private key
```

This generates a `ca` directory at our current location. This directory contains a bunch of certificates, including the root certificate. We need this root certificate to sign the server and client certificates.

> **Important:** We need to keep this `ca` directory around as it is required to sign client certificates! Store it in a **secure** location!

Okay, let's sign the server certificate using our root certificate:

```
server:/root# bbstored-certs ca sign-server /etc/boxbackup/bbstored/server.example.com-csr.

This certificate is for backup server

server.example.com

Signing the wrong certificate compromises the security of your backup system.

Would you like to sign this certificate? (type 'yes' to confirm)
yes
Signature ok
subject=/CN=server.example.com
Getting CA Private Key

Certificate signed.

Install the files

ca/servers/server.example.com-cert.pem
```

```
ca/roots/clientCA.pem

on the server.
```

We do as the output says and copy the signed server certificates to the configuration directory:

```
server:/root# cp ca/servers/server.example.com-cert.pem /etc/boxbackup/bbstored
server:/root# cp ca/roots/clientCA.pem /etc/boxbackup/bbstored
server:/root# chown -R bbstored:bbstored /etc/boxbackup/
server:/root# chmod 700 /etc/boxbackup/
```

3.4. Starting the server

That concludes the configuration part of the server. We will have to create client accounts, but that will be covered in the chapter on client configuration. Let's try out the server:

```
server:/root# /etc/init.d/boxbackup-server start
Starting boxbackup-server: NOTICE: Starting daemon, version 0.11rc2, config: /etc/boxbackup
bbstored.
```

It seems to start. We can check `/var/log/syslog` to see if it really did:

```
server:/root# grep "Box Backup" /var/log/syslog
Jan 31 21:53:14 server Box Backup (bbstored)[15884]: NOTICE: Starting daemon, version 0.11r
```

If that's all the output, everything is now okay. Final verifications to see if the server is really running can be made using `ps` and `netstat`:

```
server:/root# ps axf | grep bbstored
 15884 ?        S      0:00 /usr/sbin/bbstored /etc/boxbackup/bbstored.conf
 15885 ?        S      0:14  \_ /usr/sbin/bbstored /etc/boxbackup/bbstored.conf
```

It's running.

```
server:/root# netstat -pant | grep bbstored
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program
tcp        0      0 82.171.91.37:2201       0.0.0.0:*               LISTEN      15884/bbsto
```

It's listening for incoming connections on IP `82.171.91.37`, at TCP port `2201`. Time to set up the client!

4. Setting up a client (bbackupd)

In this chapter we will set up a client. The client will run the `bbackupd` program, which is a daemon that scans the files you want to backup and sends them to the server if it encounters new or modified files. It does this efficiently: if there are no changes, nothing is sent to the server. Otherwise it only sends the parts of the files that have changed; not the whole file is transmitted unless it is not yet on the server.

In this chapter we designate the client (the machine you wish to backup) as `client`. The backup server which will store our backups is designated `server`. Each command is prefixed with a shell-prompt to indicate on which machine (`client` or `server`) the command should be entered. The server's Fully Qualified Domain Name will be `server.example.com`. The client's will be `client.example.com`. We will have to move some files (certificates) around between the server and the client in order to sign them. For this we will use a user account with username `user`. It is assumed the root user on both systems can use secure copy (`scp`) to move files from one system to the other under this user account.

4.1. Creating an account

Before we start installing things on the client machine, we should first create a BoxBackup store account on the server. We use the `bbstoreaccounts` tool for this. It takes five parameters:

`command`

The `command` determines **which action** we want to perform on the account. In this case we wish to create an account, so we will use the `create` command.

`account_id`

The Account ID is a a **hexdecimal account number**. We will use 00000001. A second client would be 00000002, a tenth client would be 0000000A, etc.

`raid_disk`

We must specify **a RAID (BoxBackup's userland RAID-like concept) disk** on which to store the backups for this client. In our case, we didn't setup multiple RAID disks, so we have only one: **0**.

`soft_limit`

The **Soft Limit** determines when the client will voluntarily stop sending file changes to the server. Presumably (though I'm not sure) it will not start transferring new changes when the soft limit has been exceeded on the server. However, if a single change exceeds the soft limit after the client has started uploading it, it will continue uploading until it hits the hard_limit. We'll set the soft limit to 100Gb.

`hard_limit`

The **hard limit**. This should NEVER be bigger than the free space you have. We set it too 120Gb.

Let's create the account:

```
server:/root# bbstoreaccounts create 00000001 0 100g 120g
NOTICE: Account 0x00000001 created.
```

4.2. Installation

Time to move on to the actual client. Note that some of these commands are run on the client and some are run on the server. **Double check the prompt to see on which machine the command needs to be run!**.

```
client:/root# aptitude install boxbackup-client
```

Debian will configure some things for us, but that's just confusing (and didn't work for me), so we **remove Debian's configuration and keys**:

```
client:/root# rm /etc/boxbackup/bbackupd.conf
client:/root# rm /etc/boxbackup/bbackupd/*
```

4.3. Client configuration / certificates

Now we generate the configuration ourselves. We supply the `lazy` mode parameter which will continuously scan our system for changes. This sounds resource intensive, but it actually spreads the load better over time. We also supply the account ID (which we generated earlier on), the server name (FQDN), a run directory where BoxBackup can store temporary information and a path we want backed up. (We add more later on). We will be backing up `/var/lib/svn`.

```
client:/root# bbackupd-config /etc/boxbackup/ lazy 00000001 server.example.com /var/run /va

Setup bbackupd config utility.

Configuration:
   Writing configuration file: /etc/boxbackup//bbackupd.conf
   Account: 00000001
   Server hostname: server.example.com
   Directories to back up:
      /var/lib/svn/

Note: If other file systems are mounted inside these directories, then
they will NOT be backed up. You will have to create separate locations for
any mounted filesystems inside your backup locations.

Generating private key...
Generating RSA private key, 2048 bit long modulus
...................................+++
.................+++
e is 65537 (0x10001)
You are about to be asked to enter information that will be incorporated
```

```
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter ’.’, the field will be left blank.
-----
Country Name (2 letter code) [AU]:
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (eg, YOUR name) []:
Email Address []:

Please enter the following ’extra’ attributes
to be sent with your certificate request
A challenge password []:An optional company name []:

Generating keys for file backup
Writing notify script /etc/boxbackup//bbackupd/NotifySysadmin.sh
Writing configuration file /etc/boxbackup//bbackupd.conf

===================================================================

bbackupd basic configuration complete.

What you need to do now...

1) Make a backup of /etc/boxbackup//bbackupd/00000001-FileEncKeys.raw
   This should be a secure offsite backup.
   Without it, you cannot restore backups. Everything else can
   be replaced. But this cannot.
   KEEP IT IN A SAFE PLACE, OTHERWISE YOUR BACKUPS ARE USELESS.

2) Send /etc/boxbackup//bbackupd/00000001-csr.pem
   to the administrator of the backup server, and ask for it to
   be signed.

3) The administrator will send you two files. Install them as
      /etc/boxbackup//bbackupd/00000001-cert.pem
      /etc/boxbackup//bbackupd/serverCA.pem
   after checking their authenticity.

4) You may wish to read the configuration file
      /etc/boxbackup//bbackupd.conf
   and adjust as appropriate.

There are some notes in it on excluding files you do not
wish to be backed up.

5) Review the script
      /etc/boxbackup//bbackupd/NotifySysadmin.sh
   and check that it will email the right person when the store
   becomes full. This is important -- when the store is full, no
   more files will be backed up. You want to know about this.
```

```
6) Start the backup daemon with the command
      /usr/local/bin/bbackupd /etc/boxbackup//bbackupd.conf
   in /etc/rc.local, or your local equivalent.
   Note that bbackupd must run as root.

===================================================================

Remember to make a secure, offsite backup of your backup keys,
as described in step 1 above. If you do not, you have no backups.
```

> **Important:** As mentioned in the output, it is vital that you keep a copy of `/etc/boxbackup//bbackupd/00000001-FileEncKeys.raw` in a secure place, otherwise you will NOT be able to restore backups!

4.4. Signing certificates

The previous step has generated a client SSL certificate sign request at `/etc/boxbackup/bbackupd/00000001-csr.pem`. This file must be sent to the server so that it can be signed and added to the server's known SSL certificates. Here we use `scp` to copy the file to the server, but you can use any other means to get them there.

```
client:/root# scp /etc/boxbackup/bbackupd/00000001-csr.pem user@server.example.com:
    00000001-csr.pem                   100%  899     0.9KB/s   00:00
```

Now we must sign the certificate on the server. **Note** that the following commands are ran **on the server, not the client**. Also note that your current working directory must be the one in which we previously ran the `bbstored-certs ca init` command. (the `ca` directory must be present).

```
server:/root# bbstored-certs ca sign /home/user/00000001-csr.pem

This certificate is for backup account

00000001

Ensure this matches the account number you are expecting. The filename is

/home/user/00000001-csr.pem

which should include this account number, and additionally, you should check
that you received it from the right person.

Signing the wrong certificate compromises the security of your backup system.

Would you like to sign this certificate? (type 'yes' to confirm)
yes
Signature ok
subject=/CN=BACKUP-00000001
Getting CA Private Key
```

```
Certificate signed.

Send the files

ca/clients/00000001-cert.pem
ca/roots/serverCA.pem

to the client.
```

We send the signed certificate and the server's certificate to the client:

```
server:/root# scp ca/clients/00000001-cert.pem user@client:
    00000001-cert.pem          100%  997      1.0KB/s   00:00
server:/root# scp ca/roots/serverCA.pem user@client:
    serverCA.pem               100% 1021     1.0KB/s   00:00
```

And **on the client we install them** in the correct directory:

```
client:/root# mv /home/user/00000001-cert.pem /etc/boxbackup/bbackupd/
client:/root# mv /home/user/serverCA.pem /etc/boxbackup/bbackupd/
```

4.5. Starting the client

Now we can start the client:

```
client:# /etc/init.d/boxbackup-client start
```

In the syslog on the **client** we should see:

```
Jan 31 21:55:13 client Box Backup (bbackupd)[20975]: NOTICE: Starting daemon, version 0.11r
Jan 31 21:55:13 client Box Backup (bbackupd)[20975]: NOTICE: Beginning scan of local files
Jan 31 21:55:14 client Box Backup (bbackupd)[20975]: NOTICE: About to notify administrator
Jan 31 21:59:45 client Box Backup (bbackupd)[20975]: NOTICE: Finished scan of local files
Jan 31 21:59:46 client Box Backup (bbackupd)[20975]: NOTICE: About to notify administrator
Jan 31 21:59:46 client Box Backup (bbackupd)[20975]: NOTICE: File statistics: total file si
```

In the syslog on the **server** we should see:

```
Jan 31 21:55:13 server Box Backup (bbstored)[15884]: WARNING: Message from child process 16
Jan 31 21:55:13 server Box Backup (bbstored)[16488]: NOTICE: Login from Client ID 0x0000000
Jan 31 21:58:46 server Box Backup (bbstored)[19666]: NOTICE: Session finished for Client ID
```

4.6. Verifying backups

After the initial backup is done, we can verify that files have been transfered using the `bbackupquery` tool. This tool should be run on the client:

```
client:/root# bbackupquery
NOTICE: Box Backup Query Tool v0.11rc2, (c) Ben Summers and contributors 2003-2008
Login complete.

Type "help" for a list of commands.

query > ls
00000002 -d---- var-lib-svn-
00001798 -d---- etc-
query > cd var-lib-svn-
query > ls
00000003 f----- .htpasswd
00000004 f----- .svnaccess
query > exit
```

Not all files may have been transferred initially. For instance, Box Backup doesn't transfer files if they've been modified recently. It will try again later. If a file is continuously modified, it will eventually (after a max age) upload the file anyway.

4.7. Adding additional paths

If you want to add additional paths which need to backed up, edit the `/etc/boxbackup/bbackupd.conf` file and search for the `BackupLocations` directive. You can add paths here. For example, to backup `/var/lib/svn` and `/etc`:

```
BackupLocations
{
        var-lib-svn-
        {
                Path = /var/lib/svn/
        }
        etc-
        {
                Path = /etc/
        }
}
```

Afterwards simply restart `bbackupd`:

```
client:/root# /etc/init.d/bbackupd restart
```

4.8. Example client configuration

Here's a full example of a client configuration. This configuration has been stripped of comments for brevity. Please consult your own configuration for the meaning of each directive.

```
StoreHostname = server.example.com
AccountNumber = 0x00000001
KeysFile = /etc/boxbackup//bbackupd/00000001-FileEncKeys.raw

CertificateFile = /etc/boxbackup//bbackupd/00000001-cert.pem
PrivateKeyFile = /etc/boxbackup//bbackupd/00000001-key.pem
TrustedCAsFile = /etc/boxbackup//bbackupd/serverCA.pem

DataDirectory = /var/run

NotifyScript = /etc/boxbackup//bbackupd/NotifySysadmin.sh

UpdateStoreInterval = 3600
MinimumFileAge = 21600
MaxUploadWait = 86400

KeepAliveTime = 120

FileTrackingSizeThreshold = 65535
DiffingUploadSizeThreshold = 8192
MaximumDiffingTime = 120

CommandSocket = /var/run/bbackupd.sock

Server
{
    PidFile = /var/run/bbackupd.pid
}

BackupLocations
{
    var-lib-svn-
    {
        Path = /var/lib/svn/
    }
    etc-
    {
        Path = /etc/
    }
    var-www-
    {
        Path = /var/www/
        ExcludeFilesRegex = .*\.(mp3|MP3|avi|log|bak)$
        ExcludeFilesRegex = .*\.log\.1$
        ExcludeFilesRegex = .*\.log\..*\.gz$
    }
    home-
    {
        Path = /home/
    }
}
```

5. Troubleshooting

The primary methods of troubleshooting are experimentation and the `/var/log/syslog` file.

5.1. Server connectivity

We can use various tools to test if the server is running and can be reached. To check if the server is running, use `ps`:

```
server:/root# ps axf | grep bbstored
 15884 ?        S      0:00 /usr/sbin/bbstored /etc/boxbackup/bbstored.conf
 15885 ?        S      0:14  \_ /usr/sbin/bbstored /etc/boxbackup/bbstored.conf
```

If it’s not running, check `/var/log/syslog` for any errors.

To see if the server is listening for incoming connections from the clients:

```
server:/root# netstat -pant | grep bbstored
tcp        0      0 82.171.91.37:2201       0.0.0.0:*               LISTEN      15884/bbsto
```

This shows the server is listening on IP `82.171.91.37` on TCP port `2201`. The daemon should listen on an which can be reached by the clients. If it listens on IP `0.0.0.0`, it will accept connections from any machine.

To test if we can reach the server from the client, we can use a port scanner such as `nmap`:

```
client:/root#  nmap -p 2201 server.example.com
Starting Nmap 4.62 ( http://nmap.org ) at 2011-02-01 14:21 CET
Interesting ports on server.example.com (82.171.91.37):
PORT     STATE SERVICE
2201/tcp open  ats
```

If `nmap` lists the port as `open`, the client can reach the server.

5.2. Syslog errors and warning

Both the server and the client will log errors to `/var/log/syslog`. Here are a couple of errors you might run into:

5.2.1. Expired certificate

If you spot this in the syslog:

```
Jan 31 17:32:45 server Box Backup (bbstored)[30775]: WARNING: Message from child process 31
Jan 31 17:32:45 server Box Backup (bbstored)[31604]: ERROR: SSL error during Accept: error:
Jan 31 17:32:45 server Box Backup (bbstored)[31604]: WARNING: Exception thrown: ConnectionE
Jan 31 17:32:45 server Box Backup (bbstored)[31604]: ERROR: Error in child process, termina
```

One of the certificates has expired. You may be dealing with a bug in older Debian/Ubuntu versions. More information is available here:

```
http://www.mail-archive.com/debian-bugs-rc@lists.debian.org/msg242206.html
```

5.2.2. Certificates missing

THe following error is reported when certificates can't be loaded. In this case, because they couldn't be found.

```
Jan 31 18:38:32 server Box Backup (bbstored)[12102]: ERROR: SSL error during Load certifica
Jan 31 18:38:32 server Box Backup (bbstored)[12102]: ERROR: SSL error during Load certifica
Jan 31 18:38:32 server Box Backup (bbstored)[12102]: ERROR: SSL error during Load certifica
Jan 31 18:38:32 server Box Backup (bbstored)[12102]: WARNING: Exception thrown: ServerExcep
Jan 31 18:38:32 server Box Backup (bbstored)[12102]: FATAL: Terminating due to exception Se
```

Double check the instructions to make sure you've moved certificates to the correct place.

5.2.3. Permission denied

Various permission denied problems may arise:

```
Feb  1 10:47:21 server Box Backup[29518]: ERROR: FileHandleGuard: failed to open file '/etc
Feb  1 10:47:21 server Box Backup[29518]: WARNING: Exception thrown: CommonException(OSFile
```

Check the permission and ownership on the server to make sure that the files in `/etc/boxbackup` belong to the user `bbstored`. Also check that the `bbstored` user can write to `/storage/backup`.

5.2.4. Certificate verification failure

On the **client**, you may see the following error:

```
Jan 31 17:32:46 client Box Backup (bbackupd)[7339]: ERROR: SSL error during Connect: error:
Jan 31 17:32:46 client Box Backup (bbackupd)[7339]: WARNING: Exception thrown: ConnectionEx
Jan 31 17:32:46 client Box Backup (bbackupd)[7339]: NOTICE: About to notify administrator a
Jan 31 17:32:46 client Box Backup (bbackupd)[7339]: ERROR: Exception caught (Connection TLS
Jan 31 17:32:56 client Box Backup (bbackupd)[7339]: NOTICE: File statistics: total file siz
Jan 31 17:33:13 client Box Backup (bbackupd)[7339]: NOTICE: Terminating daemon
```

This might be related to the Debian bug mentioned in the *Expired certificate* troubleshooting chapter. Check the server's `/var/log/syslog` for more details.

5.2.5. Network connectivity problem

In case of the following:

```
Jan 31 18:34:28 client Box Backup (bbackupd)[27060]: ERROR: Failed to connect to socket (ty
Jan 31 18:34:28 client Box Backup (bbackupd)[27060]: WARNING: Exception thrown: ConnectionE
Jan 31 18:34:28 client Box Backup (bbackupd)[27060]: NOTICE: About to notify administrator 
Jan 31 18:34:29 client Box Backup (bbackupd)[27060]: ERROR: Exception caught (Connection So
Jan 31 18:34:39 client Box Backup (bbackupd)[27060]: NOTICE: File statistics: total file si
```

Perform the checks given in *Server connectivity* part of the troubleshooting chapter.

5.3. Missing files

Sometimes, after a backup, files may not be present on the remote server. Not all files may have been transferred initially. For instance, Box Backup doesn't transfer files if they've been modified recently. It will try again later. If a file is continuously modified, it will eventually (after a max age) upload the file anyway.

5.3.1. Other problems

The BoxBackup homepage lists some other possible problems:

`http://boxbackup.org/trouble.html`

6. About this document

6.1. Copyright / License

6.2. Feedback

All feedback on this document is welcome at <ferry DOT boender AT gmail DOT com>.