TSMManager

Considerations when using backupsets.

Contents

What is a backupset ?

A backupset is a snapshot of the active files for a single node. It is generated fully on the server and does not involve the node. The backupset can be retrieved directly from the server by the node over the LAN or it can be placed on a medium that the node can read and thus be restored to the node locally without involving the server.

Backupsets versus archives

The backupset is often considered a replacement for doing archives, but both has both positive and negative features :

	Positive	Negative
Backupsets	?? Generated on the server, does not put a load on client and LAN. ?? Can be restored locally on the node if a common media is used. ?? No time dependency, all the week can be utilized to generate the backupsets. ?? Writing is directly to tape, no frontend diskpool is necessary. ?? Practically no impact on database size.	?? Each backupset occupies at least one full volume. ?? Generating backupsets can be a very slow process. ?? Single file restore can only be done using the command line client and you must know the filename beforehand. ?? No support for backupsets in TSM's scheduling mechanism. ?? No central overview over which generations ran OK and which did not.
Archives	?? Single file restore is easy because you can see the filetree displayed graphically. ?? Full support for archives in TSM's scheduling mechanism.	?? Puts a load on both clients and LAN. ?? Big installations will often not be able to finish the complete archiving within the timespan of a weekend. ?? Many archiving schedules will report 'Failed' due to locked files even though they ran OK. ?? Generating many archives simultanously requires a diskpool as frontend to the tapepool. ?? Can make your database grow a lot !

What can be done to improve the usability of backupsets ?

Each backupset occupies a full volume.

?? If you store the backupset on a VTS, no space is wasted.

?? If you are running more than one TSM server, then you can store backupsets as virtual volumes on another server. This way each backupset is considered an archive file on the target server and multiple backupsets can be stored on each physical volume.

?? It is possible to use virtual volumes on a single TSM server. This requires the DRM feature, but could be an excellent solution if Tivoli made a few improvements. See appendix A for an example of how to do this.

Scheduling the generation of backupsets.

One way of scheduling backupsets is by inserting a number of “generate backupset...” commands into a server script and schedule the execution of this script. If you have 4 or more drives available, you can utilize more than one script and thus run the processes in parallel. But, it is your responsibility to keep the scripts updated with new nodes and balancing the nodes between scripts to best utilize your drives.

Using backupsets for disaster recovery

If you only use backupsets as a replacement for archives, then skip this chapter, but if backupsets are your prime medium of restore in a disaster situation, then there are some point to be aware of.

If you create a backupset that contains ALL the files from a node, then by restoring this set completely, you will overwrite the system part of your node. (c: for windows, rootvg for AIX etc. etc.)
This is probably not what you want. In a disaster restore situation, you will normally restore the system files from an image backup or through normal OS installation.
Afterwards you want to restore ALL OTHER files.

You CAN do this using a backupset, but if you want to restore only part of a backupset, you will have to run the complete retrieval process for each desired filesystem. Each invokation of "restore backupset" can only hold one filespec and will require a full read of the entire backupset, even though you are restoring only a few Mbytes.

It may take several hours to run through a large backupset, so if you have to do this 10 times, because you want to restore 10 different filesystems, the time involved may be just too much !

There are two solutions to this :

1. When you generate the backupset, specify all the desired filesystems on the "generate backupset" command. This way you only get the filesystems you want and can restore them all in one restore pass. BUT, it requires a lot of discipline to maintain the "generate backupset" commands. Whenever a server has a new filesystem added to it, you must go into your script or wherever you keep the generate command, and update it. Can you be sure that you are always notified about the new filesystems added to your servers and can you be sure that the backupsets will really contain ALL the filesystems needed for disaster recovery ?

2. Logically split each server that you backup into two nodes. Backup all the system files using one nodename (nodeA) and backup all other data using another nodename (nodeB).
Now, when you generate the backupset, generate it for all filesystems (*) for nodeB only. The backupset will then contain all non-system files and can be restored in one pass.

How can TSMManager help you in using backupsets ?

It can :

?? Schedule the process of backupset generation. Either do ALL nodes or let you decide.

?? Dynamically utilize your drives to the fullest.

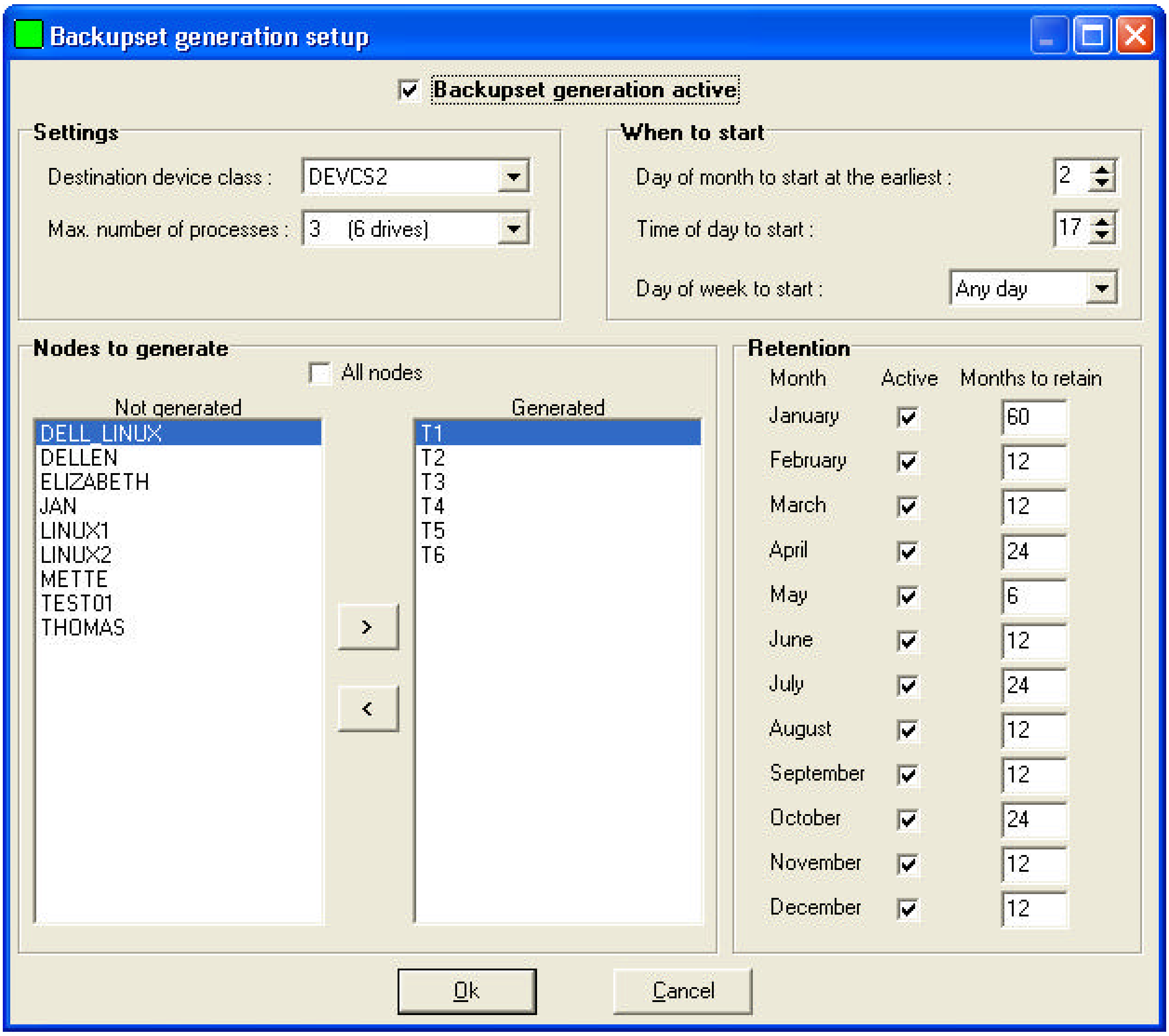
?? Control retention based on which month it is. This means that you can set it up to retain the backupsets from january for maybe 60 months, and the backupsets for the other months for maybe 12 months or any other combination you may wish for.

?? After each monthly run, it will generate a status report showing what went OK and what did not.

?? Restart all failed backupsets

?? Restart a single backupset.

Here you see the setup window for backupsets :



Backupset generation setup

☑ Backupset generation active

Settings

Destination device class : DEVCS2

Max. number of processes : 3 (6 drives)

When to start

Day of month to start at the earliest : 2

Time of day to start : 17

Day of week to start : Any day

Nodes to generate

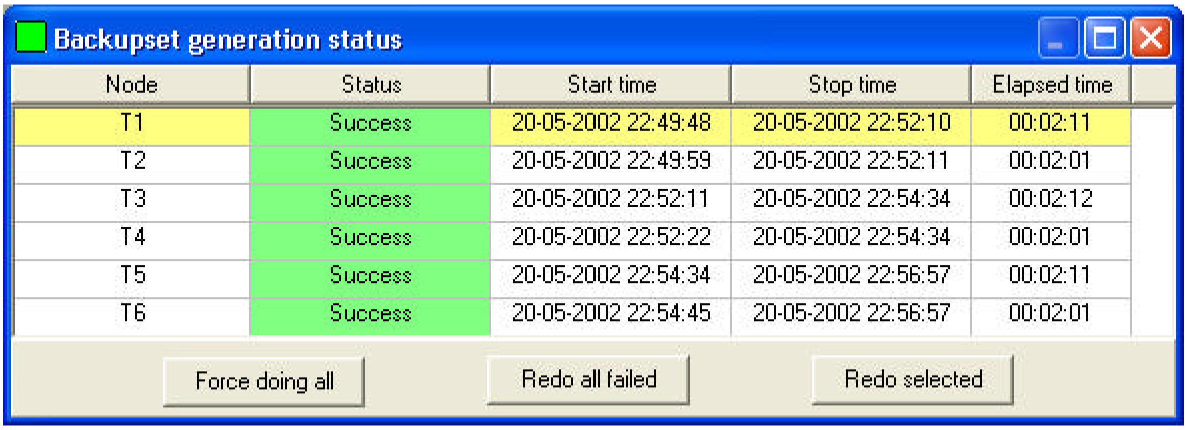
□ All nodes

Not generated	Generated
DELL_LINUX	T1
DELLEN	T2
ELIZABETH	T3
JAN	T4
LINUX1	T5
LINUX2	T6
METTE	
TEST01	
THOMAS	

Retention

Month	Active	Months to retain
January	☑	60
February	☑	12
March	☑	12
April	☑	24
May	☑	6
June	☑	12
July	☑	24
August	☑	12
September	☑	12
October	☑	24
November	☑	12
December	☑	12

As you can see, it is pretty straight forward. We designed it with one monthly run in mind. If you wish to create backupsets more often than that, then you can disable the automatic generation and do it manually from the backupset status window seen below :



Node	Status	Start time	Stop time	Elapsed time
T1	Success	20-05-2002 22:49:48	20-05-2002 22:52:10	00:02:11
T2	Success	20-05-2002 22:49:59	20-05-2002 22:52:11	00:02:01
T3	Success	20-05-2002 22:52:11	20-05-2002 22:54:34	00:02:12
T4	Success	20-05-2002 22:52:22	20-05-2002 22:54:34	00:02:01
T5	Success	20-05-2002 22:54:34	20-05-2002 22:56:57	00:02:11
T6	Success	20-05-2002 22:54:45	20-05-2002 22:56:57	00:02:01

The data in the status window is not realistic, it is a result of our test with a lot of small test nodes doing 2 backupsets in parallel.

Appendix A Using virtual volumes on a single TSM server

Virtual volumes are perfect for backupsets, because they are created with just the size of the data put into them and they are stored together on physical tapes, so no tape space is wasted. Virtual volumes are normally used with 2 servers, one source and the other the target server, but we have been testing it with one server and it seems to work OK except for some snags.

The simple setup is as follows :

Define a node of type server under which to store the virtual volumes :

```
        reg no virtnode virtnode t=s maxnummp=6
```

Define a remote server which in reality is the same server :

```
        def server virtserv hla=127.0.0.1 nodename=virtnode password=virtnode
```

Define a device class using this server :

```
        def devc virtclas devt=server servername=virtserv mountlimit=6
```

Now you are ready to generate backupsets that store data in device class virtclas. With this setup, the data is stored in the storagepool pointed to by the default archive copy group of the domain in which virtnode resides.

If you want to direct the data to a special pool not used for archiving you must do it like this :

```
        def dom virtual
        def pol virtual virtual
        def mg virtual virtual virtual
        assign defmg virtual virtual virtual
        def co virtual virtual virtual t=a dest=specialpool
        act pol virtual virtual
        reg no virtnode virtnode t=s maxnummp=6 dom=virtual
        def server virtserv hla=127.0.0.1 nodename=virtnode password=virtnode
        def devc virtclas devt=server servername=virtserv mountlimit=6
```

The snags we ran into are that if you do a "delete backupset xxx yyy", the administrator session is hung. This is because there is a session open for virtnode. If you cancel the session for virtnode, then the administrative session is freed and the backupset is actually deleted.
Another snag is that when running multiple backupset generations in parallel, the communication sometimes gets locked up and only a "cancel session" of one of the server to server sessions will make it continue. As it is now, it cannot be recommended for production use, but if Tivoli could fix the hangs, then it would be a superb way of handling backupsets on one server.