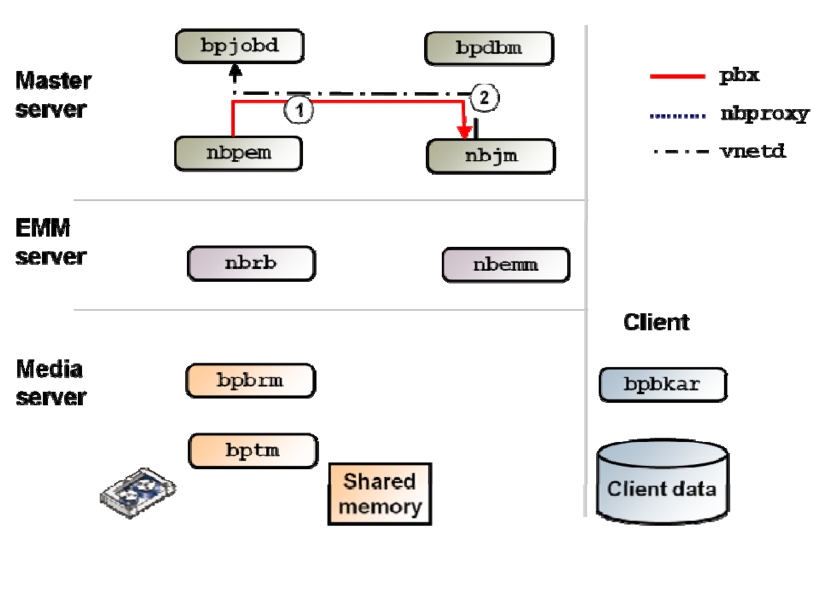
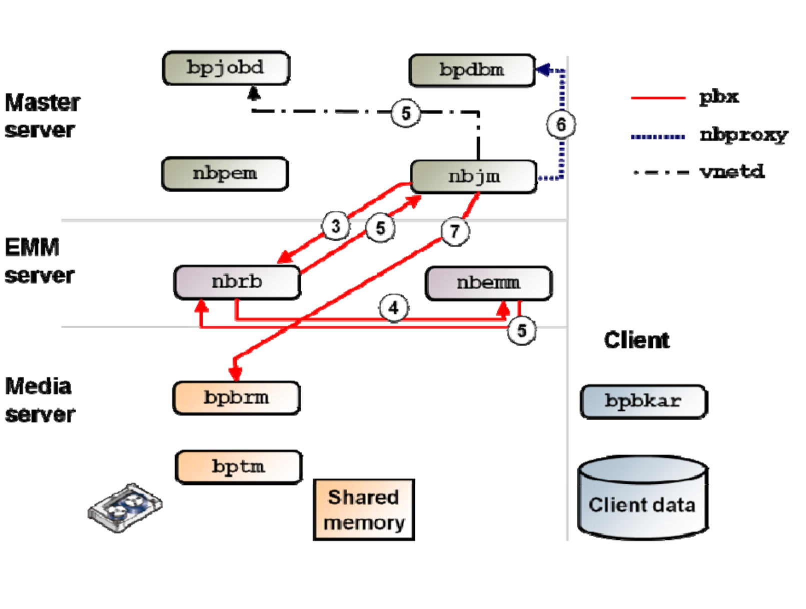
NetBackup^{TM} 7.x Backup Process Flow

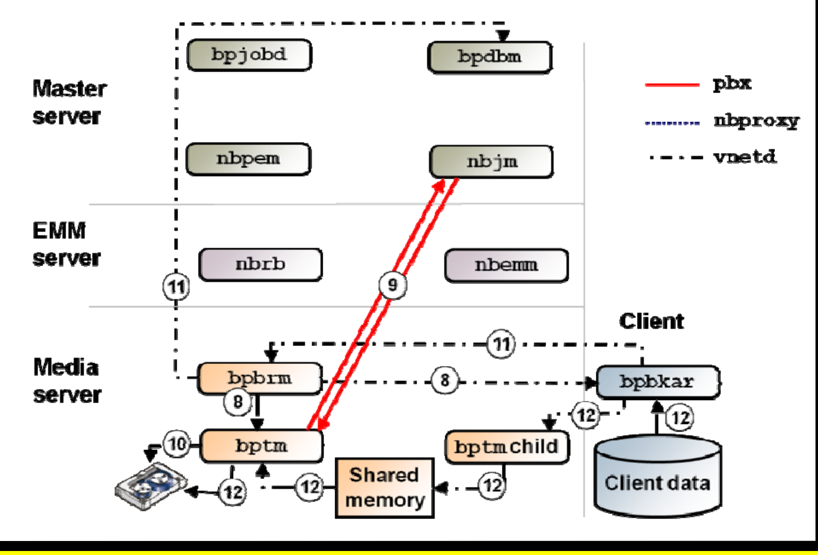
Steps 1 & 2 – Job Becomes Due



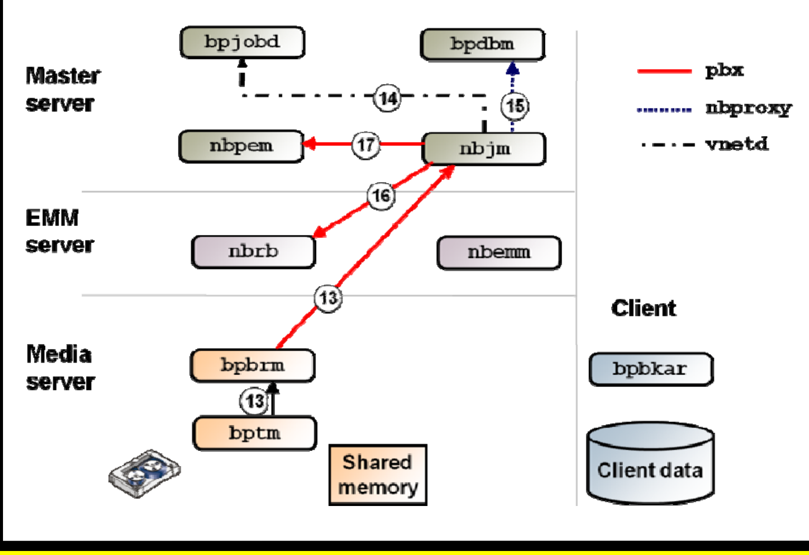
Steps 3-7 – Resource Acquisition



Steps 8-12 – Media Server Writes Backup



Steps 13-17 – Job Finish/Cleanup



Complete Backup Process Flow

1. When a PolicyClient task has its timer expire (indicating that it is due to run), an internal job task is created within **nbpem** that sends a Job Start to **nbjm** for the job which is due. nbpem provides to **nbjm** the parameters indicated in the backup policy and schedule that is generating the job.

2. **nbjm** adds the job to its job list or queue. It then communicates with **bpjobd** to inform it of the job, at which time the job becomes visible in the Activity Monitor in the queued state until resources are allocated for it.

3. **nbjm** sends a resource allocation request to the Resource Broker, **nbrb**, indicating the resources which are required for the backup operation and any resource consumption constraints for the job, including **max jobs per policy**, **max jobs per client**, and **max jobs this client**. These resource consumption constraints were provided to **nbjm** by **nbpem** when the job was initiated.

4. **nbrb** requests resources from the EMM service, **nbemm**, including storage unit, storage unit group, media, and devices or drives.

5. When physical resources are available, **nbemm** will allocate them and respond to **nbrb**, which in turn responds to **nbjm**. With resources allocated for the job, **nbjm** will notify **bpjobd** and the job moves to the active state.

6. **nbjm** is responsible for creating the files in the Images database that will house the backup information, the Header file and the Files (.f) file. **nbjm** initiates this activity by communicating with **bpdbm** (via **nbproxy**).

7. **nbjm** uses **bpcompatd** to communicate via PBX to start **bpbrm** on the media server that will write the backup image. The media server is selected based upon the destination storage unit that is selected.

8. **bpbrm** on the media server starts **bpbkar** (the client's backup and archive process) on the client system. **bpbrm** also starts **bptm** on the media server.

9. **bptm** initiates a connection with **nbjm** in order to get media and drive information for the backup job, which **nbjm** returns through a separate connection it initiates.

10. **bptm** will then initiate the mount of the media (tape) specified on the drive specified, or the mount of the disk specified. It will also spawns a **bptm** child process to receive the image from the client. The details of the Media Manager daemons (**ltid**, **txxd**, **txxcd**, and **avrd**) involved in the mounting of the media on the drive are not shown in this illustration, to reduce the complexity of the illustration.

11. **bpbkar** sends information about the backup image to **bpbrm**, which forwards it to **bpdbm** on the master server. This stream of metadata is sent throughout the backup and stored in the master server's Image database.

12. When mounting and positioning of the media in the drive, or of the disk to be used, have been accomplished, the client backup process, **bpbkar**, will begin sending backup data to the **bptm** child process on the media server system. The **bptm** child process receives the image and stores it block by block into a shared memory segment on the media server. The parent **bptm** process retrieves the image from shared memory and directs it block by block to the allocated storage media.

13. When the backup has been completed **bptm** will notify **bpbrm**, which in turn will notify the Job Manager, **nbjm**, that the job has finished. **bptm** will also notify **nbjm** that it is done with the media.

14. While the client and media server processes invoked to perform the backup operation (**bpbrm**, **bptm**, and **bpbkar**) are terminated, **nbjm** will update the status for the job by communicating with **bpjobd**. The job will be changed to Completed status, and the ending status of the job will be recorded.

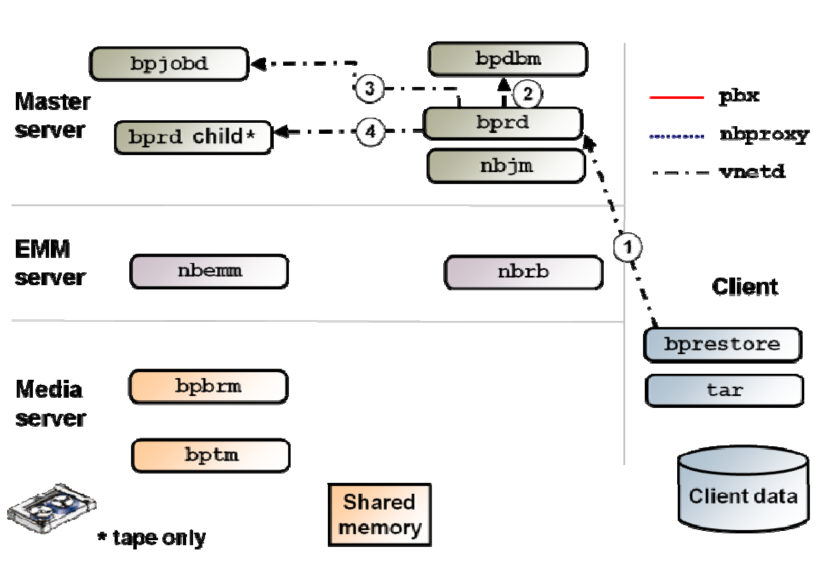
15. **nbjm** communicates with **bpdbm** (using **nbproxy**) to complete the writing and verification of the files for the backup image in the Images database.

16. With the backup job completed **nbjm** will de-allocate the resources used for the backup by communicating with **nbrb**.

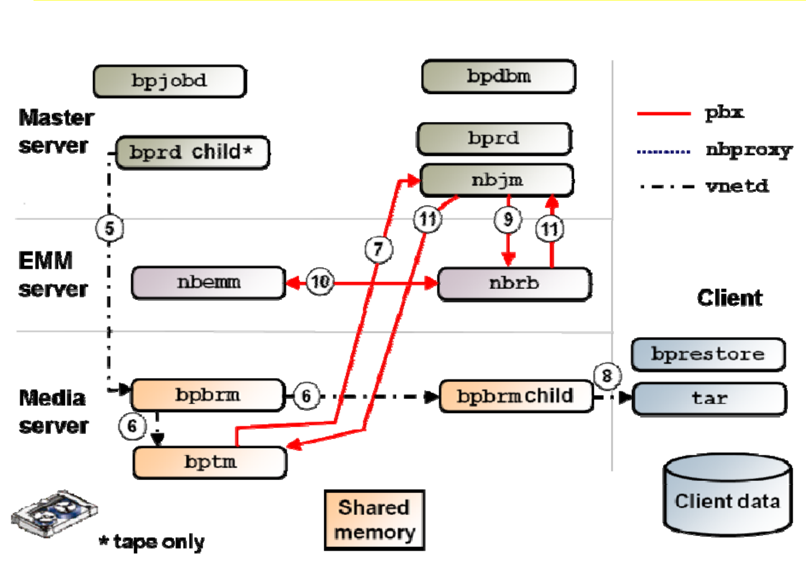
17. **nbjm**, will notify **nbpem** that the job has been completed. The completion status will be included in this notification. The PolicyClient task that created the job is responsible for requesting a re-try operation for this job on failure, or for computing the new 'due time' for this job on success.

NetBackup^{TM} 7.x Restore Process Flow

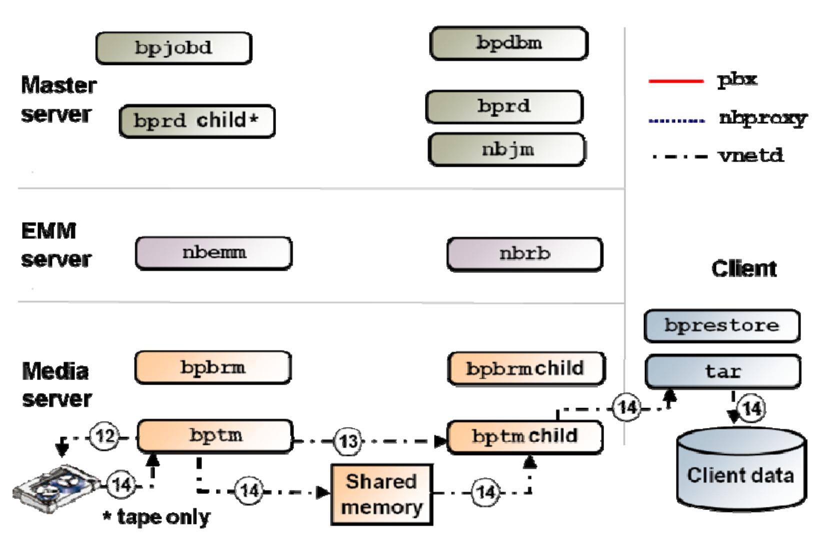
Steps 1-4 – Restore Initiated



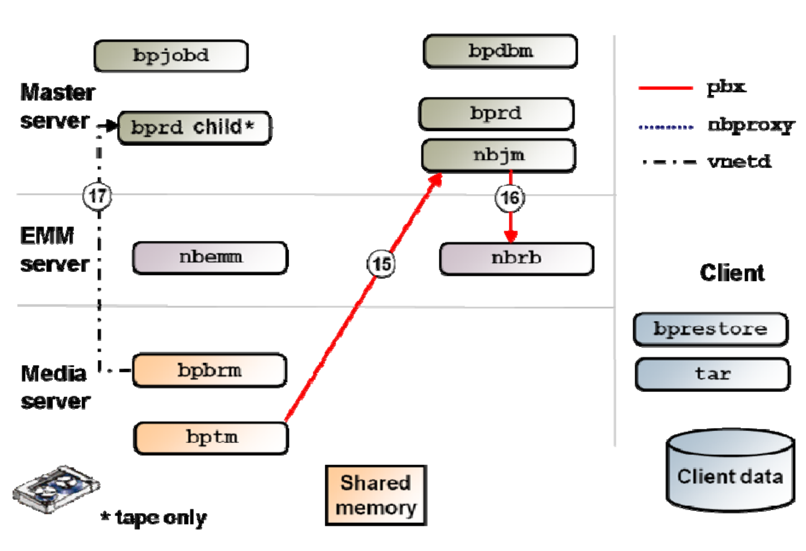
Steps 5-11 – Pre-processing



Steps 12-14 – Media Mounted



Steps 15-17 – Job Finish/Cleanup



Complete Restore Process Flow

1. A restore request is received by **bprd**. This request could be initiated from the **Backup, Archive, and Restore GUI** or from the command line (**bprestore**). The request contains a list of files to be restored.
2. The **bprd** process queries the Image database on the master server via **bpdbm** to get information required to restore the files that have been requested. **bprd** identifies the images required and the media or disk on which the images reside. If the restore is from tape, **bprd** writes a *neededimage* file which identifies the media required for the restore, the images required, and the position of the images on the media, and other information relevant to the restore. For restores from disk, there is no *neededimage* file.
3. The **bprd** process communicates with **bpjobd** to initiate the restore job. The restore job would now show up in the **Activity Monitor**. The restore job may show as Active even before resources are acquired.
4. For restores from tape, **bprd** initiates a **bprd** child process to manage the restore operation. This process reads the *neededimage* file to obtain the media and positioning information for the restore. For restores from disk, no **bprd** child process is created and all **bprd** functions are handled by the parent **bprd** process.
5. The **bprd** child process launches the **bpbrm** process on the media server which will be involved in the restore operation. In doing so, the **bprd** child provides the media and positioning information to **bpbrm**, along with the client name and other relevant information.
6. The **bpbrm** process on the media server initiates both a **bpbrm** child process and a **bptm** process . In starting the **bptm** process, **bpbrm** provides the media or disk information required for the restore.
7. With resource requirements determined by the previous pre-processing phase, **bptm** sends a resource allocation request to the Job Manager, **nbjm**, on the master server system, naming the media or disk that is required for the restore. These resources must be allocated before the job can begin.
8. The **bpbrm** child process is responsible for starting the **tar** process on the client system to which the restore will be directed and creating a connection between **tar** and **bptm**.
9. **nbjm** makes a resource allocation request to the Resource Broker, **nbrb**.
10. **nbrb**, communicates with **nbemm** to request and allocate the resources that will be used for the restore. This is an over-simplification of the activities that take place in the selection of resources by **nbrb**. Communications may also occur to **bpdbm** (using **nbproxy**) on the Master Server to identify other information relevant to resources required. This communication is not illustrated here in order to simplify the illustration.
11. **nbrb** returns the resource allocation to **nbjm**, which forwards it to **bptm**.
12. **bptm** makes a mount request of the media specifying the tape drives on which the media is to be mounted, or of the disk that is to be mounted. This illustration does not identify the details of the Media Manager daemons (**ltid**, **txxd**, **txxcd**, and **avrd**) that are involved in the mounting of the media.
13. Once the media has been mounted and **bptm** has positioned the media or disk to the location where the data to be restored resides, **bptm** spawns a child **bptm** process.
14. During the retrieval of data, the parent **bptm** process reads data from the media or disk and delivers it to shared memory block by block. The child **bptm** process delivers the data to the **tar** process on the client system, which writes the data to the client disk.
15. When the data has been restored, the **tar** process on the client, and the child **bptm** and child **bpbrm** processes on the media server will exit. **bptm** will unmount the media used during the restore, and then notify the Job Manager, **nbjm**, that the restore operation has been completed.
16. **nbjm** will de-allocate the resources used during the restore by communicating with the Resource Broker, **nbrb**.
17. **bpbrm** will notify the **bprd** child process that the operation is completed. The **bprd** child process will remove the *neededimage* file, all processes for the restore exit, and the restore is finished.

Symantec.