VMFS Backup and Recovery from Storage Controller

Anurag Singh[1,2], Chhavi Sharma[1], Deepti Banka[1,2], Giridhar Appaji Nag Yasa[1], Pallav Dhobley[1,3], Prashasti Baid[1,2], and Shivangi Singh[1,2]

[1]NetApp India Pvt. Ltd.
[2]Birla Institute of Technology & Science, Pilani, India
[3]Indian Institute of Technology, Bombay, India

***Abstract*—Virtualized infrastructure is widely used in IT deployments today. Also, users now demand high data availability at all times. Hence a mechanism for backing up and restoring data in Virtual Machine File Systems (VMFS) is important. The existing backup and restore workflows involve agents installed in each Virtual Machine (VM) that need to be manually invoked. In this paper, we present a new model for taking backups and restoring contents of a VMFS. The main difference in our approach is the presence of the backup agent in the storage controller instead of in each individual VM, thereby making the backup and restore process more efficient as the data resides in the storage contoller itself. Our approach has three advantages (i) it saves on network bandwidth (ii) Enables the storage controller to prioritize between backup requests and other I/O requests (iii) Makes the backup and restore process independent of the VM state.**

I. Introduction

USERS today demand high data availability and reliability. Efficient backup mechanisms protect user data in case of accidental deletions, physical equipment failure, virus attacks and environmental disasters.

Virtual infrastructure is widely deployed today. Virtualization helps consolidate the underlying hardware from both computation and capacity perspectives. It also brings down the associated administrative and provisioning costs along with significant user time saving.

With the importance of Virtual Machines (VMs), an efficient backup and restore mechanism for Virtual Machine File System (VMFS) has a significant value. Existing mechanisms like VCB[4] and VADP[5], [6] suffer from various limitations, they fetch the data to be backed up, sends it to the backup agent or server which redirects the entire data to the backup destination. Thus, the entire data is sent once (twice, sometimes) over the wire to complete a backup request. The storage controller, being load unaware, does not have the ability to prioritize the user I/O requests above the backup request. The existing designs also do not have provisions for file level recovery of data from a VMFS.

The recently conducted Info-Pro survey[17] profiling the storage industry revealed that backup administration consumes a good part of storage administrators' time. Also, the survey showed that most users today would like to protect data in a deployed virtual infrastructure without the use of backup agents and with the provision for file level recovery from virtual image backups. These insights further motivate the need for a new backup mechanism for a virtual machine file system.

In this paper, we present a design and describe our implementation for the backup and restore utility of a virtual machine file system which overcomes certain limitations of the existing mechanisms along with easing the job of a backup administrator. This proposed design also does better in terms of resource utilization.

The rest of the paper is organized as follows: Section II elaborates on the need for a

Student Authors: Anurag Singh, Deepti Banka, Pallav Dhobley, Prashasti Baid and Shivangi Singh

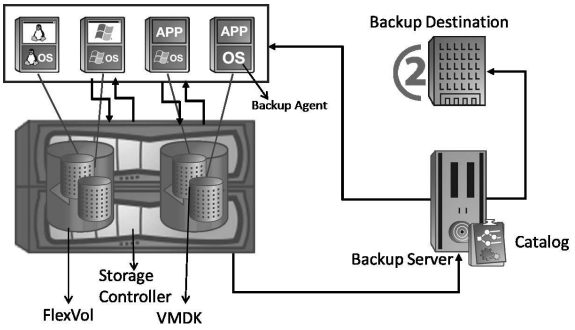

Fig. 1: Existing architecture

new architecture for VM backups detailing the limitations in the existing design. In Section III of the paper, we discuss the work already done in the area and try to compare our design with existing models. Section IV of the paper focuses on the design proposed by us and the implementation details are listed in Section V . This is followed by explaining the limitations of our approach and the future work we wish to pursue further in the area.

II. Motivation

Figure 1 depicts the existing way backup of a Virtual File System is done. Through a backup agent installed in each VM running on the hypervisor, a backup request is sent to the storage controller. This request comes as a regular read request to the controller and the requested Virtual Disk is sent to the backup server. The backup server stores the requisite metadata information in a catalog and sends the data to the backup destination.

This architecture suffers from various limitations:

- **Large Network Bandwidth:** As seen in Figure 1, backup of a single virtual disk takes three network hops, two involving complete data movement and one as a request. When many virtual machines are running off the hypervisor, lot of available bandwidth would be consumed by the backup tasks.
- **Backup Agent in each VM:** For the architecture to be fully functional, a backup agent needs to be installed in each VM. For backing up all the VM's, storage admin is required to manually trigger the backup agent from each VM.
- **VM in power on state:** Since the request for backup or restore needs to be issued from the VM, this design limits the backup/restore mechanisms to only when the Virtual machine is in the power on state.
- **No request prioritization by the storage controller:** This design does not distinguish a read request from a backup request or a write request from a restore request. Within this design, the storage controller lacks intelligence and acts just as an intermediary for fetching and writing data. Thus, it is unable to prioritize regular jobs over backup jobs.
- **Coarse granularity backup and restore:** The storage controller sees the entire file system on the virtual disk as one single file. So the entire file system is treated as one single object and the backup and restore occur only at a whole file system level. Often in practical applications, user wants to recover a single file.

Storage Functions	**2011**	**2012**
Storage Migrations	8%	17%
Backup Administration	*21%*	*15%*
Storage Provisioning	18%	14%
Storage Administration	17%	10%
Performance Troubleshooting	8%	10%
Others	28%	35%

TABLE I: Results from InfoPro Survey: Time distribution of storage administration staff

In addition to these limitations, a couple of other factors emphasize the importance of a new architecture. Any kind of data loss is intolerable today and thus storage administrators end up devoting a lot of time to backup administration (See Figure I). Thus a simplified and efficient backup and restore utility for a virtual file system is really in demand today. Insights from the recent InfoPro Sur-

Features for virtual infrastructure backups	**% of admins desiring the feature**
File level recovery from virtual machine image backups	95%
Protect virtual infrastructure without backup agent	14%
Restart virtual intrastructure from backup copy	10%

TABLE II: Results from InfoPro Survey: User requirements of virtual infrastructure backup features

vey[17] in Figure II show the importance of file level recovery for a VM file system without involvement of the backup agents.

Our proposed design helps to overcome some of the limitations and tries to keep up with the user requirements.

III. Related Work

Many traditional backup tools have been designed for taking efficient tape backups. The dump and tar[1] utilities are fairly common on UNIX® file systems which take full backups of the entire file system and also support incremental backups. Incremental backups are smaller in size and much faster to create. Rsync[15] efficiently mirrors a file system across a network using a specialized network protocol to transfer only those parts of the files that have changed. Unmodified files are hard linked between different snapshots so storage of backups is space efficient.

The falling cost of disk relative to tape makes backup to disks more attractive especially since random access permitted in a disk enable new backup approaches. Many recent backup tools like Cumulus[2] and Brackup[3] take advantage of this trend. Both these systems separate metadata from the data and each snapshot saves its own copy of metadata but data is shared across multiple snapshots to save space wherever possible.

With the increasing popularity of virtual infrastructure in huge deployments, various tools have been designed specifically for backup and restore of virtual file systems. Most of these existing tools store the backup data on a backup destination and the corresponding metadata information on the backup server. Such a utility requires a backup agent running in each VM and each backup agent utilizes the computing prowess of the hypervisor thereby, increasing the load. VMware Consolidated Backup (VCB)[4] application centralizes this backup and restore process and offloads it to a proxy server thereby, forgoing the need for agents in each VM. However, organizations today want to backup and restore at finer granularities and not at the entire filesystem level. To enable this provision at different granularities, vStorage APIs for Data Protection[5], [6] was designed.

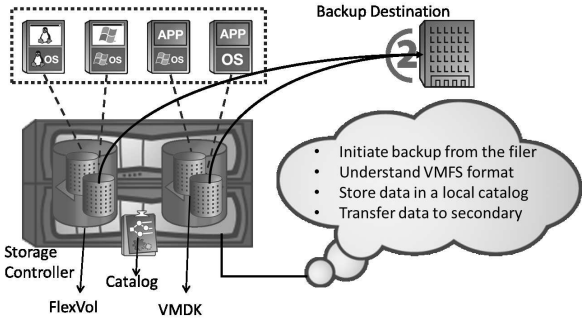

Fig. 2: Proposed architecture

Our proposed design incorporates all of the above mentioned features. Additionally, it saves network bandwidth, makes the storage controller load aware and the entire process independent of the VM state.

IV. Design

In this section, we propose our design for backing up or restoring a virtual machine file system. The major variation in our approach is an intelligent storage controller which now treats a backup request differently from a read request and a restore request differently from a write request.

Figure 2 depicts our proposed backup de-

sign. As seen in the Figure, no intervention of the hypervisor or virtual machines is required here. The backup can be initiated from the storage controller itself. We propose to place the backup and restore agent in the storage controller where the virtual disk resides as a single file in volumes(depicted in the Figure as FlexVol®[11]). Since this backup agent can be invoked from the storage controller, no network bandwidth is consumed while initiating the request.

The entire contents of a virtual machine file system i.e. all the files and directories are stored as a single file in the storage controller. When the backup agent is triggered, the corresponding virtual disk stored as a file is fetched. This virtual disk is cracked and the contents of the disk are parsed through. The storage controller can now access individual files and directories inside the virtual machine file system. Hence, backup and restore becomes possible at a finer granularity i.e. at a file and a sub-directory level. The data is moved to the backup destination. The corresponding metadata is stored in a catalog at the controller itself which can be periodically transferred to the backup destination for making the system failure proof. The entire process beginning from the triggering of the backup agent to the final placement of data at the backup destination is completed in just a single network hop.

To further enhance our design, we also achieved de-duplication over the wire for better transfer efficiency while backing up data to the destination. This saving on the network bandwidth is similar in some sense to EMC's Avamar[9]work.

This proposed design thus, gives the following advantages:

- Save on network bandwidth
- Allow setting different priorities for backup/restore requests compared to regular read/write reuests
- Independent of VM state - can run when the VM is in on, off or suspended state
- Backup and Restore at a finer granularity
- Agentless backup

V. Implementation

We used the VMware ESX® hypervisor[7][8]. A number of VMs ran on this ESX server and their corresponding virtual disks are stored in the storage controller as a single file called the Virtual Machine DisK(VMDK). This storage controller was running the NetApp® Data ONTAP®[10] operating system.

We also assumed the VMDK to be NTFS[12] formatted as it is the most commonly used file system today. NTFS-progs[13] is a library which enables NTFS formatted disks to be read on other file systems. Combining the open source VMDK format with NTFS-progs, the VMDK was cracked.

To ensure no metadata loss even during recovery, all the requisite metadata information is stored in a SQLite[14] catalog. We chose to use SQLite as it is an open source software and relatively small in size compared to other databases. Also, there is no server requirement for SQLite unlike other databases. This SQLite catalog is periodically transferred to the backup destination to protect against crash of the storage controller. In the SQLite catalog, MD5 hashes of each block are also stored to enable de-duplication over the wire while transferring data between the storage controller and the backup destination. That is we achieve better transfer efficiency through our approach by storing the checksums at a block level in the catalog.

VI. Limitations

Though our approach is better than those existing in many ways, it does suffer from its own limitations.

The entire backup utility is moved to the storage controller and therefore, the total load on the controller definitely increases. To compensate in some way, the controller is made intelligent enough to schedule backup activity at low load times.

Secondly, extra bandwidth is required for the periodic transfer of the SQLite catalog. This periodic transfer is essential for meta-

data protection in case of failure of the storage controller. This catalog would keep growing in size with time and hence a significant amount of network data movement is an overhead. The total network bandwidth consumed however would still be much less.

This approach limits the amount of flexibility available to the users as they would have to adhere to the storage vendors' backup policies.

Also, the currently deployed mechanisms are independent of the file system of the VMDKs. In our approach, since the contents of the virtual disk need to be parsed through, prior knowledge of the file system is essential. Currently, it is enabled for NTFS. For different file systems, the corresponding library would need to be ported to Data ONTAP®.

VII. Conclusions

Virtualized infrastructure is gaining popularity and is heavily used in IT deployments today. On the other hand, storage systems are required to be highly efficient and reliable. Keeping in mind both these facts, it is important to have an efficient mechanism for backup and restoration of VMFSs.

Our proposed architecture aims to achieve the backup and restoration of Virtual File System from within the Storage controller. We successfully implemented this proposed design on a NetApp storage controller for a NTFS formatted VMDK. This approach clearly has a lot of advantages over the existing deployed designs including low network utilization, an intelligent storage controller which can differentiate between backup/restore requests and regular read/write requests, backup and restore at a finer granularity and the entire process being independent of the VM state.

VIII. Future Work

As future research in continuation of this work, we would try and remove some of the limitations in our design.

Our current implementation compares the MD5 hashes at a file level which needs to be brought down to a block level to achieve even better transfer efficiency. To reduce the computational overhead for the controller, we plan to leverage the Rsync[15][16] protocol in our design for achieving de-duplication over the wire while transferring data between the storage controller and the backup destination.

Also, we plan to achieve ultimately a fully de-duplicated target in accordance with the user needs today.

IX. References

[1] W C Preston. Backup & Recovery. OReilly, 2006
[2] Vrable, Savage and Voelker, Cumulus: Filesystem Backup to the Cloud, *7th USENIX Conference on File and Storage Technologies, 2009*
[3] http://search.cpan.org/ bradfitz/Brackup/
[4] White Paper, Understanding VMware Consolidated Backup, 2007
[5] http://www.vmware.com/products-/vstorage-apis-for-data-protection/overview.html
[6] http://www.hds.com/assets/pdf/h ds-vmware-based-data-protection.pdf
[7] http://www.vmware.com/files/pdf/ VMware-ESX-and-VMware-ESXi-DS-EN.pdf
[8] http://www.cognizant.com/Insights Whitepapers/vmware_esx_wp.pdf
[9] White Paper, EMC Avamar Backup and Recovery for VMware Environments, August 2010
[10] Michael Eisler, Peter Corbett, Michael Kazar, and Daniel S. Nydick, Data ONTAP GX: A scalable sotrage cluster, *5th USENIX Conference on File and Storage Technologies, 2007*
[11] John K. Edwards, Daniel Ellard, Craig Everhart, Robert Fair, Eric Hamilton, Andy Kahn, Arkady Kanevsky, James Lentini, Ashish Prakash, Keith A. Smith, Edward Zayas, FlexVol: flexible, efficient file volume virtualization in WAFL, *USENIX 2008*
[12] http://www.ntfs.com/
[13] http://gnuwin32.sourceforge.net/packages/ ntfsprogs.htm
[14] http://www.sqlite.org/
[15] Andrew Tridgell. Efficient Algorithms for Sorting and Synchronization.PhD thesis, Australian National University, Feb. 1999.
[16] http://librsync.sourcefrog.net/
[17] https://451research.com/report-long?icid=2411