January 2020

HYPERVISOR-AGNOSTIC SYSTEM AND METHOD FOR MODIFICATION OF SNAPSHOT FILES FOR RECOVERY

Mayuresh Vartak

Geetha Srikantan

Vishwas Srinivasan

Harpreet Gandhi

Swapnil Daingade

HYPERVISOR-AGNOSTIC SYSTEM AND METHOD FOR MODIFICATION OF SNAPSHOT FILES FOR RECOVERY

AUTHORS:

Mayuresh Vartak
Geetha Srikantan
Vishwas Srinivasan
Harpreet Gandhi
Swapnil Daingade

ABSTRACT

This proposal provides a general-purpose technique to reconfigure the execution environment for virtual machines to facilitate a file patch service for modifying snapshot files during virtual machine recovery. The technique is hypervisor-agnostic and may be implemented within many different types of environments.

DETAILED DESCRIPTION

Figure 1, below, illustrates an example hyper-converged cluster implementation for a hyper-converged infrastructure platform in which the hyper-converged cluster may include a set of physical hosts. Hypervisor software is installed on each host of the hyper-converged cluster and each host may include a management virtual machine (VM), often referred to as a Controller VM (CVM). Hypervisor depends on the CVM provided and orchestrates underlying storage and snapshot related services and functionality for the hyper-converged infrastructure platform.

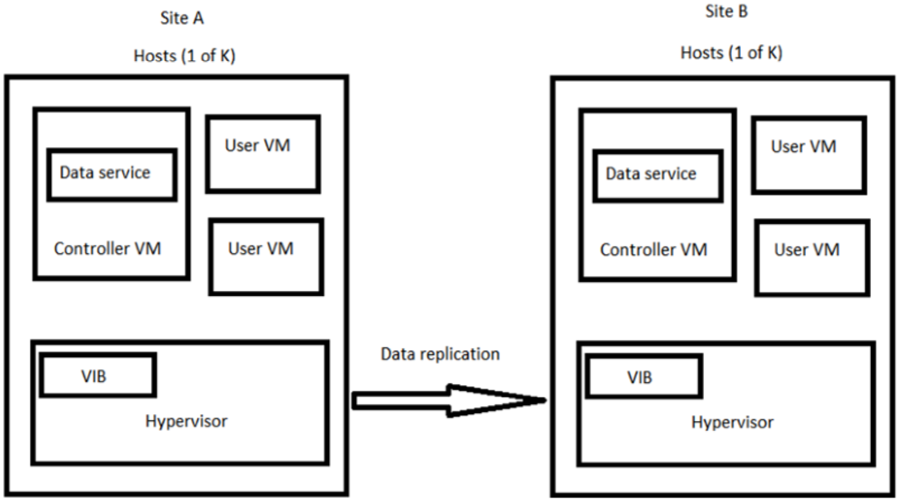

Figure 1

Hyper-converged clusters allow administrators to create datastores to deploy virtual machines in order to create logical separation of resources for easier management. Data protection (DP) and disaster recovery (DR) software is resident on the CVM of each host. Disaster recovery setup consists of two hyper-converged clusters that are deployed in separate physical environments such that they do not share any failure domains. A typical example of such a setup is two clusters deployed as active-active pair for data replication in data centers from different regions. For DP/DR setup, hyper-converged clusters are paired to establish inter-cluster communications. Datastores of each hyper-converged cluster are mapped to datastores on the peer cluster. Datastore mapping is directionless. Based on physical resources dedicated for DP/DR, any or all virtual machines present on mapped datastores can be added for protection. All configuration and data for protected VMs are replicated periodically to the peer cluster based on RPO limits set by an administrator.

Once protected, the configuration and data for a virtual machine is captured in a point-in-time copy, typically referred to as a snapshot. A snapshot can contain virtual machine data that spans datastores and can be replicated to a remote cluster. Replication of a snapshot causes the configuration and data to be available on the corresponding mapped datastores on the peer cluster. The same snapshot can be used for local recovery or for remote recovery at the peer cluster. At the time of recovery, a copy of the all the snapshot files needs to be provided to the recovery folder such that the copy is customized for the environment in which the virtual machine will be recovered.

A typical recovery workflow of a virtual machine from its snapshot may involves following steps:

1. Cloning the files from the snapshot to the recovery folder;
2. Changing attributes in the cloned files such that the virtual machine being launched will match the recovery environment;
3. Powering on the virtual machine; and
4. Connecting the recovered virtual machine to the new environment (e.g., the correct network, folder, resource pool, and other Hypervisor-specific parameters).

In step 2, properties of the virtual machine that may be edited may include:

1. Datastores (for recovery to a remote cluster);
2. Identifiers of the virtual machine (for both local and remote recovery); and/or
3. Snapshot identifiers (if the virtual machine also has hypervisor specific snapshots associated with it).

Editing the properties of a virtual machine is typically accomplished via business logic of a data protection workflow. The service that is utilized to edit virtual machine properties in order to customize it for recovery environment is henceforth referred to as "filePatch service". The context in which the file patch service can be used is shown below in Figure 2.

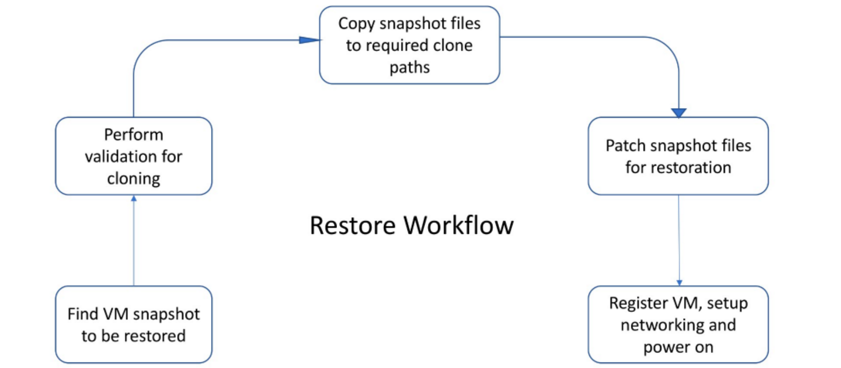
Copy snapshot files to required clone paths
Perform validation for cloning
Patch snapshot files for restoration
Restore Workflow
Find VM snapshot to be restored
Register VM, setup networking and power on

Figure 2

With regard to VMware® vSphere® hypervisor environments (as shown in Figure 3, below), one approach for modifying files during recovery may be facilitated via software utilities that may be provided within a vSphere Installation Bundle (VIB). A VIB is a software binary that is installed on a hypervisor host and is executed as a process on the host.

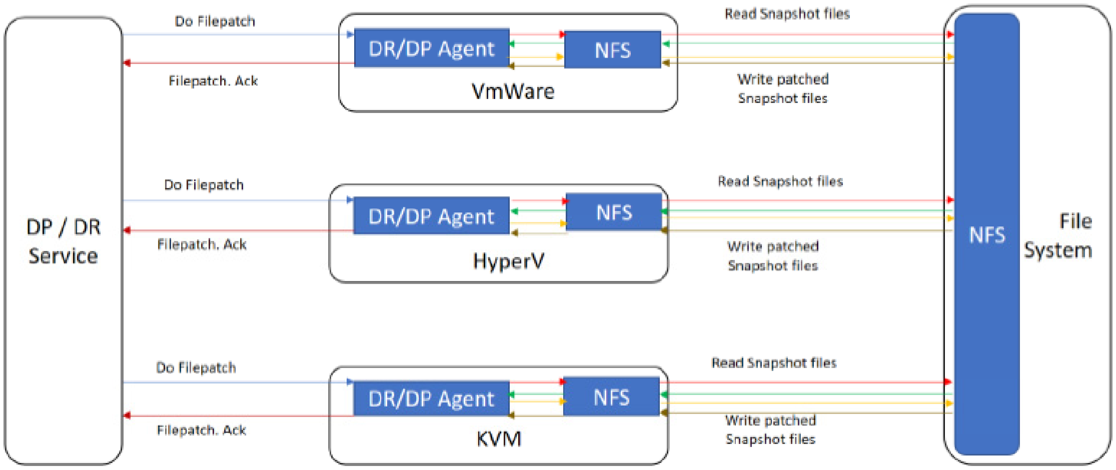

Figure 3

Since the VIB is resident on a host while the business logic for DP/DR software is installed on the CVM, an existing file patch service typically includes the following sequence of operations between the CVM and the host:

1. DP/DR services on the CVM communicate with software tools within the VIB on the host over a network;
2. The VIB utility on the host implements the detailed logic of the file patch service by making a Unix® system call to open the file on the datastore;
3. The Unix® system call is received by the host operation system, which handles the call via a host Network File System (NFS) client in order to access the datastore from an NFS server hosting the datastore(s);
4. The NFS server hosting the datastore(s) is a hyper-converged cluster storage services process;
5. After the file is retrieved from the storage services process, the host then returns the data to the software utilities on the VIB;
6. The file patch service software utility in the VIB operates on the file contents to edit requested virtual machine properties and save the modified file via a Unix® system call;
7. The Unix® system call to save this file is processed by the host operating system via the operating system NFS client and the NFS server.

These operations may be repeated for each file and file patch pattern for which edits may be needed.

Since the above method implements filepatch service inside VIB of hypervisor host, it is hypervisor dependent and requires separate software modules for each hypervisor host. The following proposal provides a technique through which a file patch service may be implemented as a service running in the CVM rather than the VIB, thereby providing a highly available, stateless service that communicates with the underlying file system via the NFS, as shown below in Figure 4.

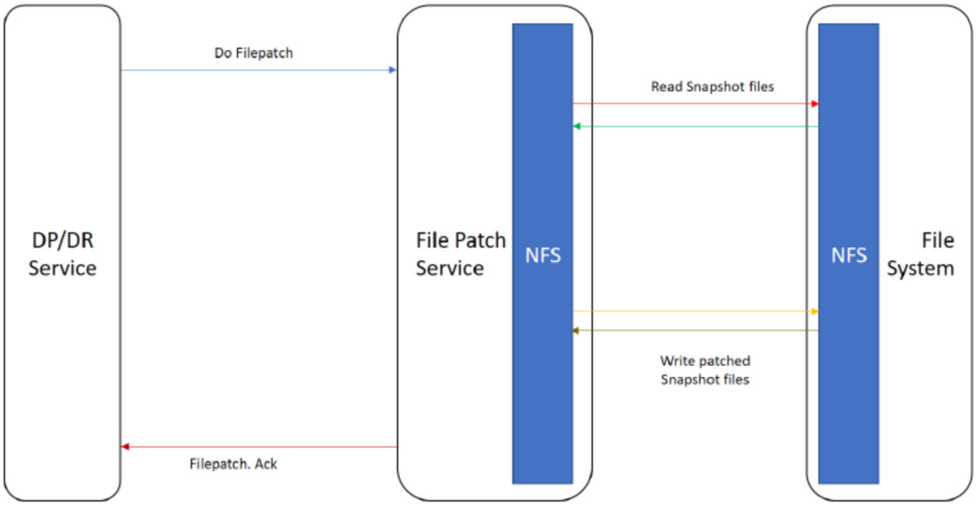

Figure 4

A new recovery workflow that may be provided in accordance with the technique of this proposal may be performed as follows:

1. A DP/DR service or client initiates a remote procedure call to the file patch service with a snapshot identifier of a snapshot that is to be modified;
2. The file patch service acknowledges the request and creates a task associated with request;
3. The file patch service contacts the file server for the file system corresponding to the datastore over NFS in order to access files associated with the file patch request;
4. The contents of the file are read over the network and requested attributes are changed in the file, which is written back to the file system over NFS; and

5. The DP/DR client is notified upon completion of the modification operation in which the notification may indicate a success status if the operation is successful and may indicate an error otherwise.

To serialize concurrent updates on the same snapshot files, all modification requests for the same snapshot files may be routed through the same instance of the file patch service.

While the existing usage of a file patch service in VMware® environments is as a separate service, the file patch service provided by the technique of this proposal may be generally applicable within any virtualized or cloud environment. For example, the file patch service of this proposal may be utilized within other hypervisor and/or non-hypervisor environments and, thus, is hypervisor-agnostic.

Several advantages may be provided by the technique of this proposal. For example, the workflow involving the file patch service of this proposal skips the usage of a VIB. Using a VIB may be fraught with potential issues. One potential issue may concern certification and approval of a VIB. While development of the software utilities within a VIB may be a one-time activity, there are several procedures involved with regard to certifying and approving a VIB for usage on VMware® hosts. Further, there may be restrictions regarding when and how a VIB may be used. For example, the restrictions for VIBs can change between different versions of vSphere®, which may cause ongoing revisions of VIB utilities. Moreover, the tools and system utilities of VMware® hosts may not be the same for different host operating system versions, which may involve further updates to VIB software. Because of security and resource issues, VIB environment limits type of operations and resources consumed, which restricts type of operations that are allowed inside VIB. In contrast, this proposal provides a technique to modify files for different scenarios/implementations by skipping operations associated with a VIB.

Additionally, the VIB approach only works for VMware® environments. For other hypervisors, such as Microsoft® Windows® HyperV®, Linux® Kernel-based Virtual Machine (KVM), etc., other mechanisms may need to be developed, which could involve additional development obstacles, time, etc. Further, container-based environments may yet involve other development obstacles, time, etc.

In contrast, this proposal provides a technique that can be utilized within both VMware® hypervisor and non-VMware® hypervisor environments and, therefore, is hypervisor-agnostic. The hypervisor-agnostic file modification technique of this proposal eliminates the need to maintain separate VIBs, tools, etc. for each hypervisor environment in which the technique may be implemented, which can provide savings for development and/or maintenance times. Further, the technique of this proposal reduces resources consumed on a distributed storage cluster platform and is highly performant.

In summary, this proposal provides a general-purpose technique to reconfigure the execution environment for virtual machines to facilitate a file patch service for modifying snapshot files during virtual machine recovery. The technique is hypervisor-agnostic and may be implemented within many different types of environments.