June 25, 2018

Consistent Database Backups Using Virtual Machine Clone Operations

Ian Gulliver

Hsiu-fan Wang

Consistent Database Backups Using Virtual Machine Clone Operations

Abstract

Consistent database backups may be created by cloning a virtual machine, gracefully stopping processes on the clone, and taking a snapshot of the clone in a consistent, clean state.

Introduction

Creating consistent database backups can be quite difficult. For example, creating such backups typically requires database shutdown, suspending writes, or careful journaling. Each of these solutions impacts performance and/or availability of the database. In some systems, filesystem snapshots of the database may facilitate creation of the database backups by using system level tools to create copies of devices holding data files. However, these snapshots require additional system configuration, and will not be entirely accurate as the device will be in an unclean state. For example, if taking a snapshot while a virtual machine is still live, database entries may be inconsistent.

Overview

An improved way of backing up a database is provided, as illustrated in Fig. 1. To backup a database in a virtual machine, the virtual machine is cloned, and the clone is configured to complete the in-progress operations from the original instance of the virtual machine, such that there are two running versions of the same virtual machine.

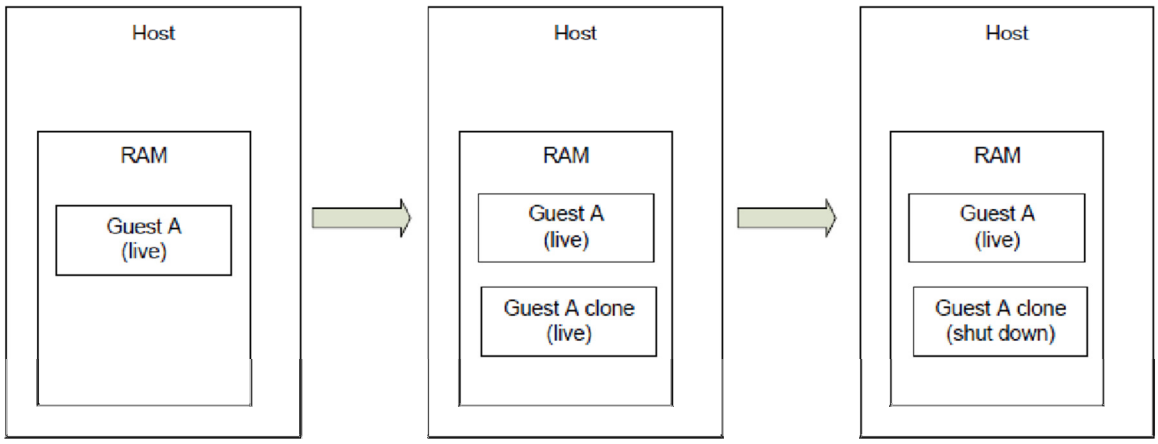

Fig. 1

The original instance remains running, while the cloned instance is gracefully shut down. Files stored by the clone are in a consistent, clean state as compared to those of the original virtual machine. A snapshot of the clone is taken, and saved as backup. After taking the snapshot, the clone may be removed.

Cloning

The cloning of the virtual machine may be similar to that performed in current systems for live migration of virtual machines. For example, storage devices associated with the virtual machine may be cloned to another storage location. In some instances, writes may be paused or buffered during cloning of the virtual machine, though preferably the original virtual machine continues operating as normal. In other instances, snapshots of the virtual machine may be taken to enable creation of the clone without pausing the original virtual machine.

The clone may be created on a same host machine as the original VM, such as shown in Fig. 1 above. However, in some instances the clone VM may be created at another host machine in the

same physical datacenter or in a remote datacenter. Creating the clone VM at a location remote from the original VM increases complexity.

Once the clone has been established, its network configuration is updated such that it is accessible to control systems. For example, for systems running Dynamic Host Configuration Protocol (DHCP), a FORCERENEW instruction may be issued by the network DHCP server to force the clone to request a new IP address. In this regard, both the original VM and the clone are running as duplicates, executing the same queries. Accordingly, the clone remains in sync with the original VM, as it commits the same transactions that were in progress at the time of the clone operation.

Stopping Processing

After the clone has been updated, the original VM continues normal operation, serving queries and accepting writes. The clone, however, will begin a graceful shutdown process. Any of a number of techniques may be used to facilitate shutdown of the clone cleanly and without causing corruption of the data served by the clone VM. For example, the clone VM may enter a lame duck mode, where it finishes execution of processes that have already started, but does not accept new queries. The clone VM may remain in this state until it has completed all processes. Once the processes have been completed, the clone VM is shut down. The network configuration may again be updated to remove connections to the clone VM, so that it no longer receives transactions from other devices. However, the VM may alternatively remain connected in the same configuration, without serving any requests.

As a result of the graceful shutdown process, files on the clone VM are in a consistent, clean state as compared with files on the original VM. In particular, the clone VM has the same record of completed transactions as the original VM, up to the clone point. Moreover, because

processes had an opportunity to wrap up cleanly, the data stored is accurate and not corrupted by, for example, failed or aborted transactions or incorrect ordering of transactions.

Snapshots

When the clone has been shut down, a snapshot of the clone is taken and saved for database backup. The snapshot may be, for example, a filesystem snapshot, capturing a state of files on the clone VM. For example, the snapshot may serve as an accurate account of the queries served by the original VM up until a time that the clone VM was shut down.

The snapshot may be saved, such as in a dedicated storage device, on the machine hosting the original VM and/or the clone VM, or in any other location. Alternatively or additionally, the snapshot may be stored in a remote location, such as to provide backup in a catastrophic event affecting the entire database. Fig. 2 below illustrates one example of a storage device storing a number of snapshots of clone VMs.

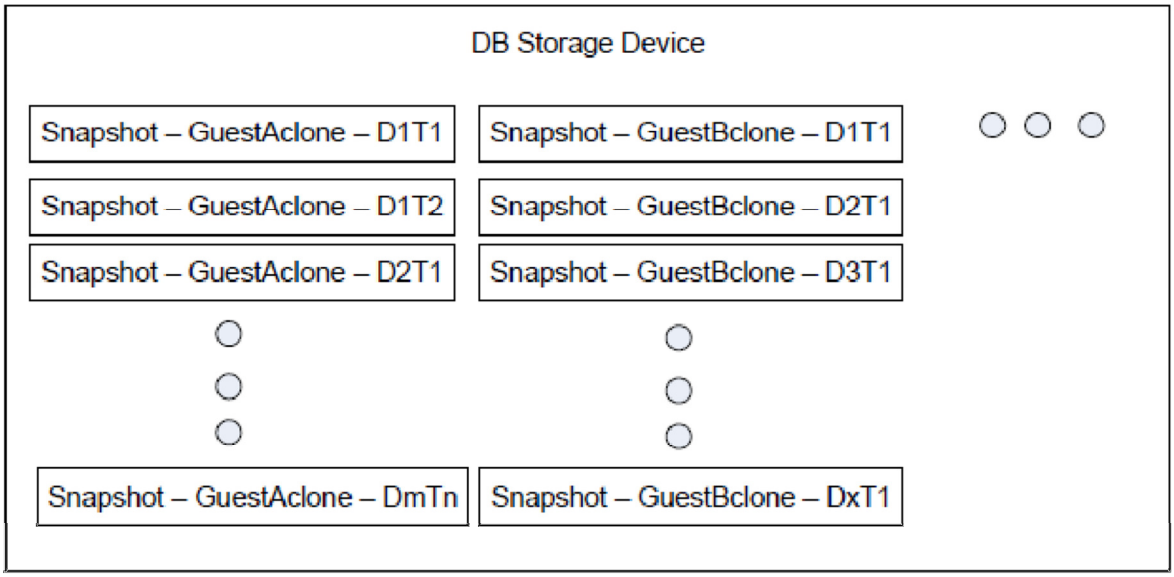

Fig. 2

As shown in this example, the storage device stores snapshots from clones of multiple guests over time. For example, snapshots for a clone of "Guest A" are stored at different times on a number of different days. In this example, snapshots for a clone of "Guest B" are also stored. "Guest B" may be another guest on the same host machine as Guest A, or it may be another guest on another device. Snapshots for any number of clones may be stored in the same device, or across different storage devices.

Snapshots may be taken and saved once a day, multiple times per day, several times per week, or with other frequency. While in some examples all VMs may be cloned at snapshotted at a same time, in other examples the cloning and snapshotting processes may be staggered among VMs. Moreover, the frequency with which VMs are cloned and snapshotted may differ from one VM to the next. For example, as shown in Fig. 2, Guest A snapshots are stored twice a day, and thus are stored with a different frequency than those of Guest B, which are stored once a day.

Snapshots may be retained for a period of time, such as several days, weeks, months, etc. For example, as shown in Fig. 2, the snapshots for Guest A are retained for "m" number of days. Snapshots for different clones within the same database may be stored for different time periods, such as the Guest B clone snapshots which are stored for "x" number of days. In other examples, the snapshots are only stored until a new snapshot for the VM in taken. For example, once Guest A is cloned and snapshotted again at Day 1 Time 2, the snapshot for Day 1 Time 1 may be overwritten.

Once the snapshot is taken and stored, the clone may be discarded. In other examples, the clone may resume processing in sync with the original VM, such that the clone continues running and executing queries along with the original VM until a next time a snapshot is needed.

Summary

Using the foregoing techniques, snapshots of a database may be taken without disrupting operation of the servers and virtual machines, and without sacrificing accuracy. These snapshots can be used for data recovery, for implementing new servers or virtual machines and bringing them up to speed, for virtual machine migration, etc.