**[High Availability with Provisioning Services](http://www.citrixvirtuoso.com/?p=94" \o "High Availability with Provisioning Services)**

May 19 2011

High Availability (HA) is one of the foundations of information security. The availability of services is critical in a Provisioning Services (PVS) in which PVS is utilized to deliver centralized virtualized desktops and applications via XenApp. If PVS is not able to deliver vDisks to Target Devices then your XenDesktop and/or XenApp environment will not be available. Accomplishing HA with PVS involves providing redundancy for the following components:

* Bootstrap
* Database
* Network
* Storage
* RAM
* Streaming servers
* Target Device booting
* vDisks
* Write Cache

**Bootstrap**

The Bootstrap instructs Target Devices connecting via PXE and TFTP/DHCP on how to boot. The Bootstrap is configurable through the Provisioning Services Console and can be configured with up to four PVS server IP addresses. At least two PVS servers should be listed in the Bootstrap file.

**Database**

A PVS Farm uses a Microsoft SQL database to store all Farm information. Provisioning Services supports a variety of Microsoft SQL database platforms (MS SQL 2005, MS SQL 2008, MS SQL 2008R2, and Express as of this writing). The availability of the database is critical to the operation of the PVS Farm. If the database is not available then no Target Devices will be able to boot and receive a streamed vDisk.

PVS provides for two configurations to make the database Highly Available:

* **Offline Database**: When a PVS server hosting the Stream service starts it connects to the database to cache certain database information in the PVS server’s RAM. This cache can be used to continue operations in the event the database becomes unavailable. If the database subsequently becomes available the Stream service synchronizes with the database.

Offline Database support should not be used in place of true database redundancy (i.e. Mirroring). Offline Database support is also not enabled by default.

* **Database Mirroring**: Database Mirroring is storing two copies of the same database on different SQL servers in an active/passive cluster, often in different locations. If using SQL Server Express be sure to verify if Mirroring is an option.

**Network**

The stream process of a vDisk from a PVS server to a Target Device is a live stream sent over network interfaces. The Target Device should maintain constant, uninterrupted communication with the PVS server. If the stream is interrupted the Target Device will likely fail. For this reason it is important to protect the integrity of the streaming traffic.

**Network Interfaces**

The failure of a network interface will cause a PVS server to lose connection with the Farm database and/or stop its Stream service. Provisioning Services servers should be equipped with multiple network interfaces capable of link aggregation. Link aggregation allows for multiple network interfaces to be bonded for throughput and/or failover purposes. A bonded pair configured for failover will ensure the continued operation of the Stream service should one of the interfaces fail.

**Streaming Network**

The PVS streaming traffic should be isolated from production network (file, print, Active Directory, etc) to ensure maximum available bandwidth and availability. This is accomplished by placing the streaming traffic on its own network or VLAN. The PVS server and each Target Device will need to have an interface attached to the streaming VLAN.

**DHCP**

DHCP is utilized to provide network devices with IP information, such as addresses, subnet masks, and Options 66 and 67. If DHCP is not available then Target Devices will not be able to connect to PVS (if booting via PXE or TFTP/DHCP) or will otherwise not be able to connect to the network. Multiple DHCP servers (with non-overlapping scopes) should be available on the network to provide IP address information.

**Storage**

PVS servers and vDisk storage locations should be configured for redundancy, usually accomplished via RAID. Without disk redundancy in place a single failed disk will result in the loss of the PVS server or the vDisks.

**RAM**

Properly determining the RAM needs of your Provisioning Services servers is important to ensure positive user experience and availability. A server equipped with inadequate RAM will experience slowness and perhaps cause the stream to Target Devices to fail.

The amount of RAM required for a PVS server is determined by the number of vDisks it will serve and how much is streamed from each vDisk. A generic formula is 3GB + (# of active vDisks \* 2). This formula is a good starting point, though it assumes each vDisk will only stream 2GB of data at most. To ensure a more accurate measurement:

1. Ensure the Provisioning Services server is not streaming any vDisks.
2. Begin monitoring the amount of network traffic transmitted over the streaming interface.
3. Boot a single vDisk, log on, run the typical applications, and shut down the vDisk. Make a note of how much traffic was transmitted during the stream process.
4. Repeat for each vDisk.

**Streaming Servers**

The Stream service is what makes PVS tick. The Stream service is responsible for delivering vDisks to Target Devices. If the Stream service is not available then no Target Devices will receive a vDisk. Fortunately this is one of the easier components to configure for HA.

* Install multiple PVS servers. Follow the standard n+1 model (i.e. plan for the complete failure of a server).
* Ensure the Stream service is running from multiple PVS servers.
* All PVS servers that will provide vDisks must have access to the vDisks.
* The PVS servers should reside within the same Farm to ensure consistent configuration.
* vDisks should be configured to use the Load Balancing algorithm rather than configured to stream from a single PVS server.
* vDisks should be configured for High Available (in File Properties) to allow a second PVS server to pick up the stream for a failed PVS server.

**Target Device Booting**

After a Target Device powers on it needs to contact Provisioning Services to receive the bootstrap file (ARDBP32.BIN). If a Target Device is not able to receive a bootstrap file then it will not receive a vDisk. Three methods exist for providing the bootstrap file to a Target Device: Boot Device Manager (BDM), PXE, or TFTP/DHCP.

**Boot Device Manager**

Boot Device Manager (BDM) uses a boot device (USB, .iso, or hard disk partition) containing the required information for a Target Device to boot and connect to a PVS server to obtain its vDisk. This is an easy solution to implement if using virtual machines as a centralized .iso file can be utilized. BDM also has the additional advantages of being the fastest boot method and also does not add any network overhead. BDM can be a bit trickier to implement if streaming to physical machines.

The BDM solution does contain a single-point-of-failure in regards to the location of the boot device. If that boot device fails or is otherwise unavailable (including an .iso residing on a file share) the Target Device will not be able to connect to PVS.

Two options are available for providing HA when using BDM:

* DFS can be utilized for the ISO Library containing the .iso file.
* Multiple boot devices can be used, such as booting to DVD/CD-ROM first and then to USB if the first boot device fails.

**PXE**

This is the easiest solution to set up. The Target Device simply boots and sends out a PXE broadcast request. Assuming the PXE service is started on a PVS server the PVS server will respond with the necessary information for the Target Device to boot.

PXE has two potential failure points:

* PXE is a single-point-of-failure when only a single PVS PXE server is available on the network.
* The Target Devices and the PXE server must also reside on the same logical (broadcast) network unless an IP Helper address is used to route the requests to a different network. If the Target Devices cannot be routed to the network then their boot will fail.

Providing HA for PXE is simply a matter of ensuring multiple PVS PXE servers are available.

**TFTP/DHCP**

When using DHCP a Target Device boots and communicates with a DHCP server to receive its IP information including two additional options: 66 and 67. Option 66 points the Target Device to a PVS server. Option 67 provides the name of the bootstrap file (ARDBP32.BIN) the Target Device is to download.

This solution has several potential failure points:

* Single DHCP server: If only a single DHCP server is utilized and that server fails the Target Devices will not receive the required PVS boot information.
* Option 66: Most DHCP servers only allow for a single entry (IP address or hostname) for Option 66. If the listed server is not available the Target Device boot will fail.

Several options:

* Multiple DHCP servers: When using DHCP to provide PVS boot information multiple DHCP servers (with non-overlapping scopes) should be available on the network. All of the DHCP servers should provide Options 66 and 67.
* Inserting a hostname in Option 66 allows the use of DNS Round Robin. The downside is that DNS Round Robin provides no method for verifying a service is available, so if one of two DHCP servers is unavailable 50% of the boot requests will still be routed to the down server resulting in 50% of the Target Devices failing to boot.
* Hardware Load Balancer: A hardware load balancer, such as a NetScaler or F5, has the ability to monitor and load balance back end services. When using this configuration the Target Device communicates with a virtual server running on the load balancer. This sounds easy, though in practice it can be difficult to configure.

My preference is to use a combination of two of the three methods (BDM, PXE, and TFTP/DHCP), particularly if you do not have the ability to provide HA for any single one of the methods. As an example, in a virtual environment I will typically configure my virtual machines to first boot to DVD/CD-ROM using BDM. Their second boot method will be to network allowing the Target Devices to boot using PXE or TFTP should the BDM boot fail.

**vDisks**

A vDisk is a virtual hard disk. A vDisk provides a bootable hard disk to diskless Target Devices (physical or virtual). If a Target Device’s vDisk is not available then the Target Device will not be able to boot. vDisks must be available to all PVS servers in the Farm to achieve HA.

vDisks can reside on a Windows share (CIFS), Fibre or iSCSI SAN, or PVS local storage (local disks or a dedicated SAN LUN).

* **CIFS**: Placing the vDisks on a Windows share is an easy solution to implement. All that is required is an accessible Windows share with enough disk space to hold the vDisks. While easy to implement, utilizing CIFS does create a single-point-of-failure and can become a performance bottleneck. DFS can be utilized to provide redundancy for the file share.
* **SAN**: Utilizing a SAN for vDisk storage allows all PVS servers simultaneous access to the vDisks, though complications can arise when modifying vDisks. Typically this configuration requires the use of a read-only LUN requiring manually modifying the LUN from read-only to read-write when a vDisk update is necessary. SANs are enterprise class storage devices and built with fault tolerance in mind.
* **Local**: Keeping the vDisks on the Provisioning Services server’s local hard disks is the easiest solution to implement. Utilizing local storage avoids a single-point-of-failure and offers HA, though the vDisks do need to be manually synchronized (unless DFS-R is utilized) across the Provisioning Services servers in the farm.

**Write Cache**

The Write Cache is a temporary file utilized by Standard and Differential mode vDisks to store changes performed to the vDisk, such as application updates, installations, etc. Write Cache files generated from Standard vDisks are removed upon reboot. Write cache files generated from Differential vDisks are removed when the base vDisk is updated. As every change to the operating system is stored in the write cache, this file can grow large.

The write cache can be located in several locations with each offering benefits and drawbacks in regard to cost, high availability, and performance.

* **Target Device RAM**: A portion of the Target Device’s RAM is utilized for Write Cache. This allows a limited form of HA. If streaming to a physical Target Device then full HA is achieved. If streaming to a virtual Target Device then the virtual machine cannot be moved between hypervisor hosts as the RAM is tied to the hypervisor host.
* **Target Device Local Storage**: With this option a portion of the Target Device’s disk storage is set aside for Write Cache. This can either be a physical device’s hard disk or storage dedicated to a virtual machine. This configuration permits HA, though if streaming to a virtual machine the attached storage should be accessible to all hypervisor hosts to achieve true HA.
* **Provisioning Services Local Storage**: This option utilizes the Provisioning Services server’s local hard disks for the write cache. This configuration offers no HA between PVS servers in a Farm.
* **Provisioning Services Shared Storage**: This option utilizes shared storage connected to the Provisioning Services server. This configuration allows for HA as every server has access to the write cache.