Recovery Console Optimization

Revision v2 - 2022-03-07

Author: Eric Harless, Head Backup Nerd - N-able

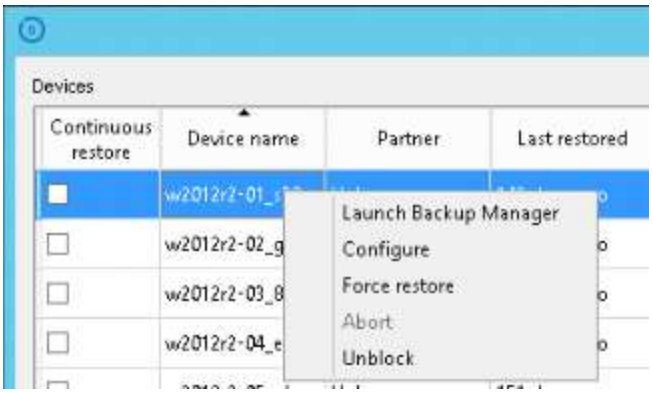
Twitter @Backup_Nerd Email:eric.harless@n-able.com

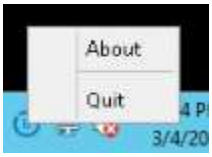
Reddit https://www.reddit.com/r/Nable/

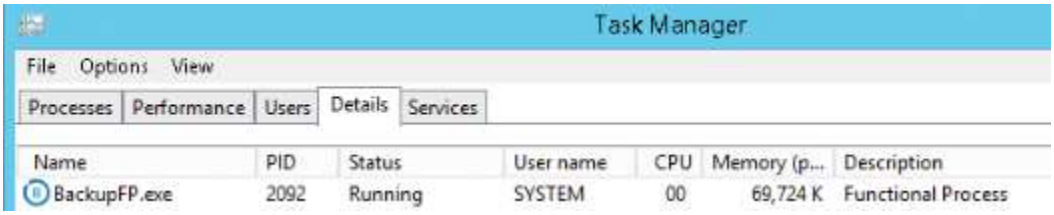
----- Legal: ----

Editing Recovery Console settings

When you install the Recovery Console, a configuration file is created in the installation folder (`C:\Program Files\RecoveryConsole\config.ini`). Advanced users can change or add some basic and advanced settings through this file. Prior to making changes to the config.ini you will need to shut down or quit the Recovery Console. You may want to do this gracefully by unchecking Continuous restore and let the running restores complete first. You can also forcefully abort the restore jobs in the console, or just simply stop the Recovery Console Service and any running BackupFP.exe processes. All of the optimizations in this document should be made while the Recovery Console is fully stopped. After which, you can restart the Recovery Console for the changes to take effect.







Maximizing concurrent restores

The default number of concurrent restores for a recovery console is 5, however achieving that may not be possible if you do not have that number of available CPU cores on the system. There are also other factors like available ram, bandwidth, disk space and disk I/O that can limit the ability to perform multiple restores at the same time.

It is also recommended that you enable Hyper-threading on your Recovery Console system, so that you effectively double the number of virtual CPU cores present.

One way to calculate maximum concurrent restores is set the system to set the recovery console to use up to N-2 CPU cores. For example, a system with dual Xeon 8-way processors with Hyper-threading enabled would present itself as a 32 virtual core system. If this system is doing nothing but recovery operations I would suggest setting the `VdrRunningFpLimit=` value no higher than 30, but possibly less if restores are not it's only function. You can check the CPU utilization percentage while the recovery console is stopped to determine what an idle system looks like.

Section	Parameter	Definition	Supported values
`[General]`	`VdrRunningFpLimit=`	The number of sessions allowed to run at a time	Any whole number 5 (Default) Recommend no higher than the # of available CPU Cores minus 2

Each concurrent restore operation can consume up to 500MB of ram during the restore phase, and several GBs each if you are performing automated boot testing (which would also consume 2 or more additional CPU cores during the boot test cycle). A good starting point would be to have between 1-2GB of ram for each concurrent restore operation on top of any ram required for the underlying operating system.

During the initial Virtual disaster recovery phase, the VM's virtual disk is provisioned, then the initial restore is written to a virtual differencing disk that is later merged into the primary virtual disk. For this reason, you could occasionally require 2X the VMs provisioned capacity to perform the initial restore. If all configured systems are not all performing their initial restore at the same time this requirement should not pose a significant problem.

While bandwidth and disk I/O limitations won't prevent concurrent restores the can lengthen the restore times and prevent you from completing in the desired timeframe.

Increasing restore threads

By default, each restore session in the Recovery Console can utilize up to 10 restore threads. If you are performing 5 or more concurrent restores then this will most likely be sufficient to fully utilize your available bandwidth. If you are not fully utilizing the available bandwidth or have only a few concurrent restores configured you can add the `RestoreDownloadThreadCount=` to the `[General]` section of the configuration file belonging to the Recovery Console.

Section	Parameter	Definition	Supported values
`[General]`	`RestoreDownloadThreadCount=`	The number of simultaneous connections during recovery (helps determines data transfer speed).	1 to 50 10 (Default) Increase in stages, setting this value too high could degrade performance.

Addressing blocked devices

The recovery console checks if the virtual machine has been in use before each virtual disaster recovery session. The target virtual machine isn't supposed to be in use while the Continuous Restore mode is active. If the Backup Manager (or the Recovery Console) detects that the virtual machine was started in-between restore sessions, further restores will be blocked and you will get an error message. The way that this error is triggered is if the Modified Date (and time) on any of the VHD/VHDX files is after the last Restore Session date and time. This is done to prevent possible data loss.

If you are sure no important data is added to the virtual machine you can unblock it.

To unblock the individual target machine:

- Right-click on device in the Recovery Console
- Click **Unblock**

You can also disable these checks permanently for all devices in the Recovery Console so that the previous version can be overwritten without warning messages. To do this, add `VdrRestorePolicyForceOverwrite=1` to the `[General]` section of the configuration file belonging to the Recovery Console.

Section	Parameter	Definition	Supported values
`[General]`	`VdrRestorePolicyForceOverwrite=`	Toggle to enforce or ignore blocked restores dur to changes found with the target VM.	0 (Enforce Blocks) 1 (Ignore Blocks)

Anti-virus related issues

Note, that some AV programs will mount the restored virtual disk files to scan them which can cause a future Blocked restores, increase ram and CPU usage as well as create significant Disk I/O. If you trust the source you are restoring from and are performing sufficient AV and malware scanning of the source systems then you may want to consider adding the following paths and executables to the antivirus exclusions list of any device where Recovery Console is installed:

Paths:

- C:\Program Files\RecoveryConsole
- [Virtual Disaster Recovery Continuous Restore Destination Path]

Executables:

- C:\Program Files\RecoveryConsole\RecoveryConsole.exe
- C:\Program Files\RecoveryConsole\ClientTool.exe
- C:\Program Files\RecoveryConsole\ProcessController.exe
- C:\Program Files\RecoveryConsole\BackupFP.exe
- C:\Program Files\RecoveryConsole\BackupIP.exe
- C:\Program Files\RecoveryConsole\BackupUP.exe
- C:\Program Files\RecoveryConsole\BRMigrationTool.exe

A sample PowerShell script to set exclusions for Windows Defender can be found here [Backup-Scripts/Security at master · BackupNerd/Backup-Scripts (github.com)]

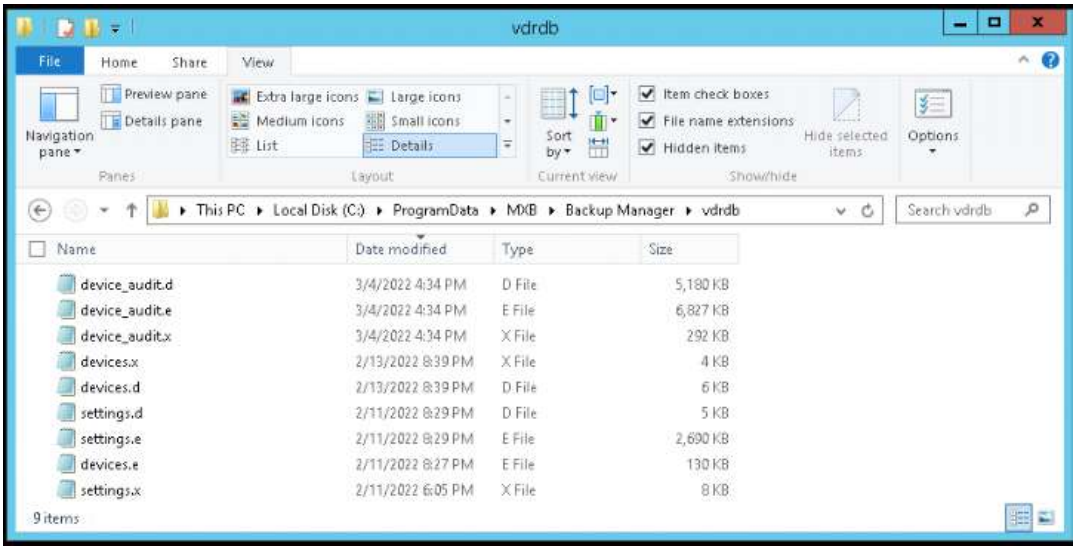
Scanning for restorable systems

When the Continuous Restore feature is on for a device, the Recovery Console connects to the N-able Cloud every 30 minutes and checks for updates. This time interval can be customized in the configuration file by adding `SessionsUpdatePeriodicityForReadOnlyModeInSeconds=` to the `[General]` section of the configuration file belonging to the Recovery Console. Combined with the other settings in this document, decreasing the above value may help allow for more daily attempts to perform continuous restores.

Section	Parameter	Definition	Supported values
`[General]`	`SessionsUpdatePeriodicityForReadOnlyModeInSeconds=`	The number of seconds to wait between checking for new restorable sessions	Any whole number 1800 (Default) 300 (Minimum)

Cloning an existing Recovery Console installation

If you have too many devices in your Recovery Console, you may be unable to complete all delta restores in the desired time frame. One possible solution is to split your operations amongst multiple Recovery console instances. You can do this without manually keying in all your existing configurations. To clone the existing installation, first stop the Recovery Console, then copying the path and full contents of `C:\ProgramData\MXB\Backup Manager\vdrdb` to the same path on the new systems where you will later install fresh copies of the Recovery Console. You can then simply uncheck (or remove) the machines that you do not want to run a Continuous restore.



Auto Starting a Recovery Console installation

Although the Recovery Console has a Windows native service, it is still partially a user mode application. If you happen to log off or reboot the system running the Recovery Console then it will not resume Continuous Restores until user logon. For most environments you can lock the computer instead of logging off and operations will continue, but that still doesn't address the issued when dealing with automated patching and scheduled system reboots.

To set the Recovery Console so that it can be restarted in an unattended fashion you will need to make the following adjustments and utilize the provided scripts.

1. Use the Services control panel to set the Recovery Console Service to Log on using an Administrator login. You may want to consider a script to periodically check that it is still running as Administrator and notify on any changes. I believe that the periodic auto update routine might revert this back to a Local System account.

2. Create or import XML a Windows Task Scheduler Task to Start the Recovery Console. [Backup-Scripts/Recovery Console at master · BackupNerd/Backup-Scripts (github.com)]

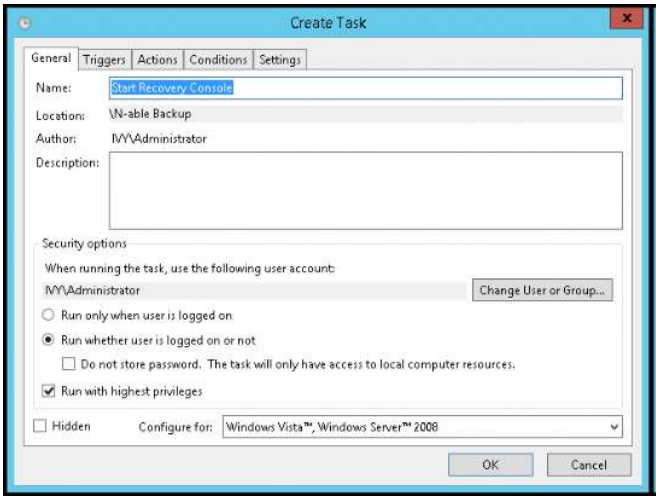


Figure 1: Set the task to run as a Local Administrator

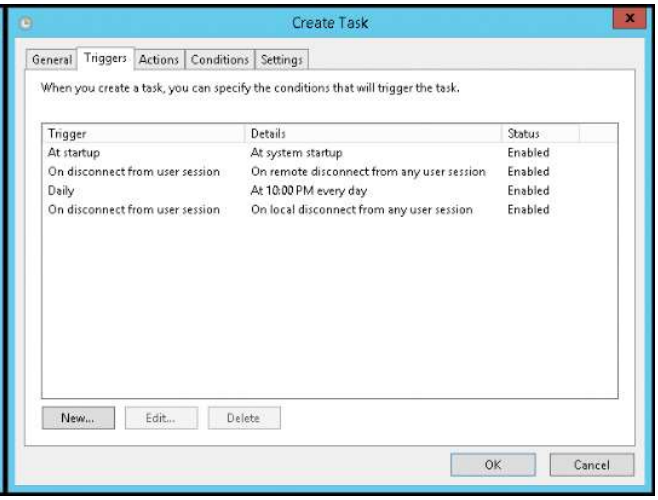


Figure 2: Set Triggers for startup, logoff and daily execution

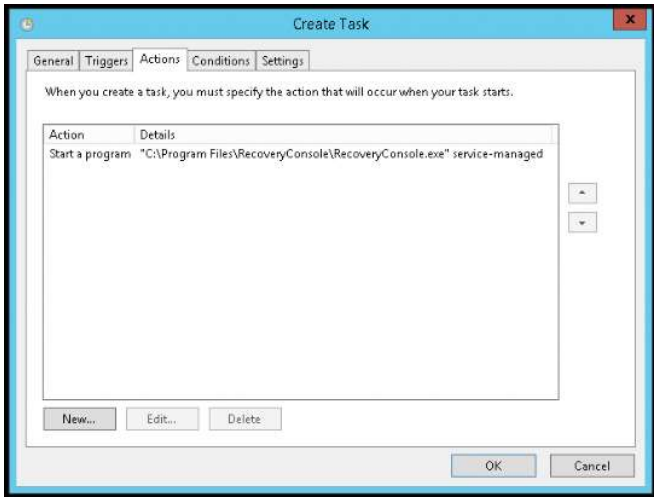


Figure 3: Set the action to start the Recovery Console with a 'service-managed' parameter

3. Add a PowerShell script to shutdown/restart the Recovery Console. This script will stop the Recovery console from running in the background (session 0) and restart it in the current logged in user session. When the system boots, or a user logs off the Scheduled Task with restart the Recovery Console in session 0. To open the Recovery Console GUI to make edits after you have logged in, it must first be stopped in session 0 and opened in the current user session. Executing this script will interrupt running restore operations. [Backup-Scripts/Recovery Console at master · BackupNerd/Backup-Scripts (github.com)]
4. Add a PowerShell script to monitor running restore instances without opening the Recovery Console. [Backup-Scripts/Recovery Console at master · BackupNerd/Backup-Scripts (github.com)]