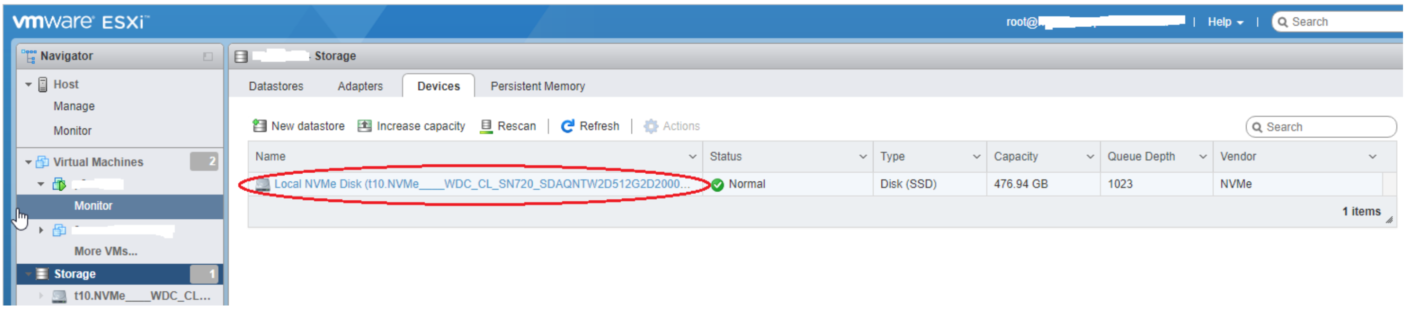
Vmware

How to Create VMFS Datastore on Vmware OS bootdrive

Normally a vmfs partition is already created by ESXi on the boot drive. You can verify by going logging into the ESX Web GUI and navigating to **Storage --> Devices --> Boot disk** (**Figure 1**).

Figure 1

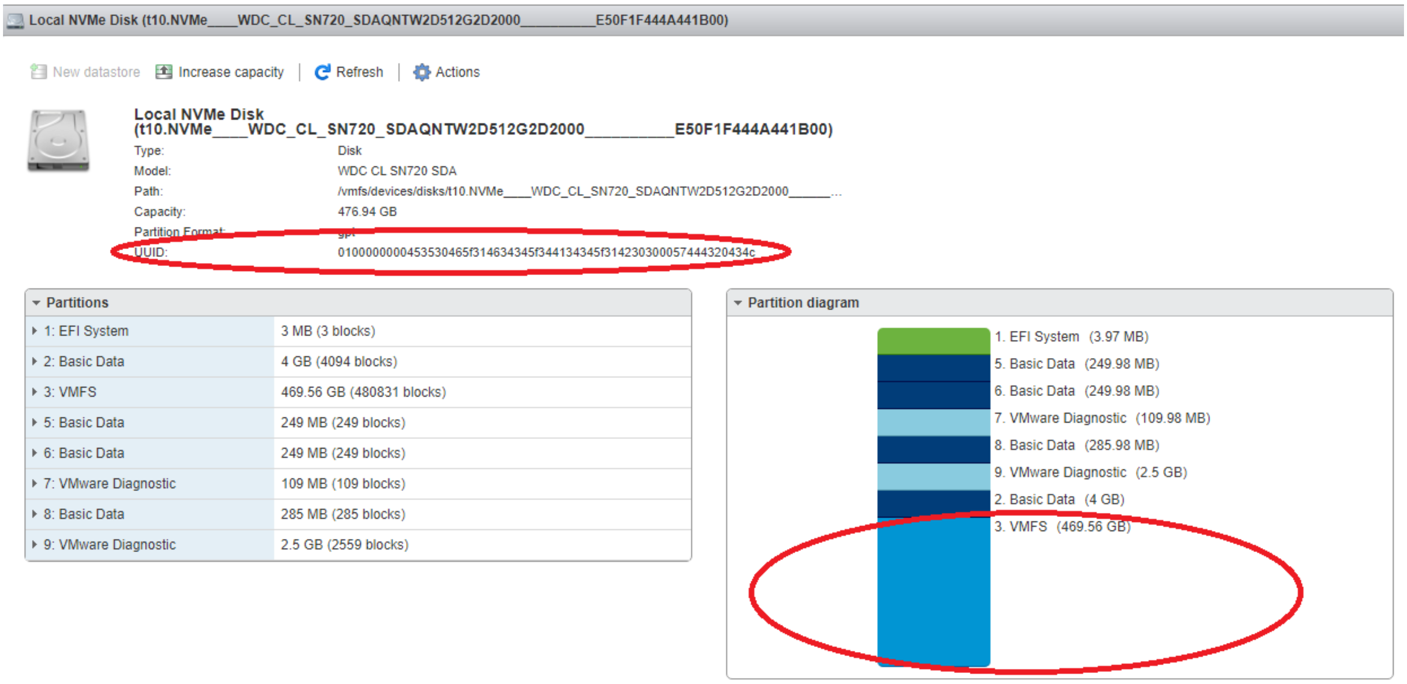


Name	Status	Type	Capacity	Queue Depth	Vendor
Local NVMe Disk (t10.NVMe____WDC_CL_SN720_SDAQNTW2D512G2D2000...	✓ Normal	Disk (SSD)	476.94 GB	1023	NVMe

1 items

Clicking on the drive will show the **Partition diagram** of the boot drive and the vmfs partition is usually partition **3**. Make a note of the **UUID** of the device and the partition number (**Figure 2**).

Figure 2



Local NVMe Disk (t10.NVMe____WDC_CL_SN720_SDAQNTW2D512G2D2000__________E50F1F444A441B00)
New datastore
Increase capacity
Refresh
Actions

Local NVMe Disk (t10.NVMe____WDC_CL_SN720_SDAQNTW2D512G2D2000__________E50F1F444A441B00)

Type: Disk
Model: WDC CL SN720 SDA
Path: /vmfs/devices/disks/t10.NVMe____WDC_CL_SN720_SDAQNTW2D512G2D2000______...
Capacity: 476.94 GB
Partition Format: gpt
UUID: 0100000000453530465f314634345f344134345f3142303000574443204 34c

Partitions

1: EFI System	3 MB (3 blocks)
2: Basic Data	4 GB (4094 blocks)
3: VMFS	469.56 GB (480831 blocks)
5: Basic Data	249 MB (249 blocks)
6: Basic Data	249 MB (249 blocks)
7: VMware Diagnostic	109 MB (109 blocks)
8: Basic Data	285 MB (285 blocks)
9: VMware Diagnostic	2.5 GB (2559 blocks)

Partition diagram

1. EFI System (3.97 MB)
5. Basic Data (249.98 MB)
6. Basic Data (249.98 MB)
7. VMware Diagnostic (109.98 MB)
8. Basic Data (285.98 MB)
9. VMware Diagnostic (2.5 GB)
2. Basic Data (4 GB)
3. VMFS (469.56 GB)

Create a vmfs datastore named **datastore-ssd** on the the boot device using the **UUID** and the partition number you noted above using the command below (Device format is /vmfs/devices/disks/vm1.**<UUID>**:**3**):

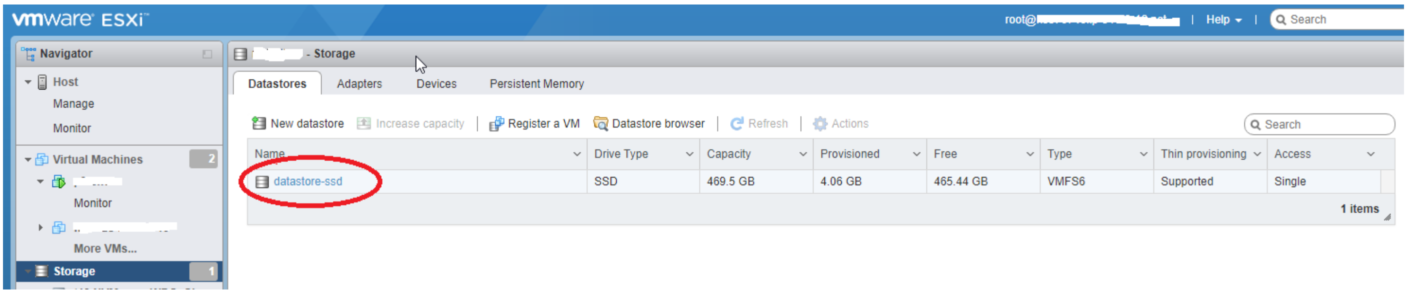
```
vmkfstools --createfs vmfs6 -S datastore-ssd
/vmfs/devices/disks/vml.0100000000453530465f314634345f344134345f314230300057444320434c:3
```

If successful, you should get an output similar to below:

```
create fs
deviceName:'/vmfs/devices/disks/vml.0100000000453530465f314634345f344134345f314230300057444320
434c:3', fsShortName:'vmfs6', fsName:'datastore-ssd'
deviceFullPath:/dev/disks/t10.NVMe____WDC_CL_SN720_SDAQNTW2D512G2D2000__________E50F1F444A441B
00:3 deviceFile:t10.NVMe____WDC_CL_SN720_SDAQNTW2D512G2D2000__________E50F1F444A441B00:3
ATS on device
/dev/disks/t10.NVMe____WDC_CL_SN720_SDAQNTW2D512G2D2000__________E50F1F444A441B00:3: not
supported
.
Checking if remote hosts are using this device as a valid file system. This may take a few
seconds...
Scanning for VMFS-6 host activity (4096 bytes/HB, 1024 HBs).
Creating vmfs6 file system on
"t10.NVMe____WDC_CL_SN720_SDAQNTW2D512G2D2000__________E50F1F444A441B00:3" with blockSize
1048576, unmapGranularity 1048576, unmapPriority default and volume label "datastore-ssd".
Successfully created new volume: 5fc102ac-257939bc-7e5b-0cc47ac8166c
```

Additionally, the datastore you created should appear under **Storage --> Datastores** in your ESXi Web Gui (**Figure 3**).

Figure 3



Name	Drive Type	Capacity	Provisioned	Free	Type	Thin provisioning	Access
datastore-ssd	SSD	469.5 GB	4.06 GB	465.44 GB	VMFS6	Supported	Single

1 items

Vmware ESXi 6.x NUT Client Installation and Configuration

Requirements

- SSH must be enabled on your ESXi installation
- Community acceptance level must be enabled on your ESXi installation in order to install the client

Installation

1. Secure copy the NutClient-ESXi-2.x.x.tar.gz to your ESXi server’s /tmp directory by either using WinSCP/pscp in Windows or scp in Linux.
2. Set ESXi Community Acceptance level:

```
esxcli software acceptance set --level=CommunitySupported
```

3. Untar NutClient-ESXi-2.x.x.tar.gz:

```
tar -xzf NutClient-ESXi-2.x.x.tar.gz
```

4. Install Client:

```
sh upsmon-install.sh
```

5. If installation was successful you should see the following output:

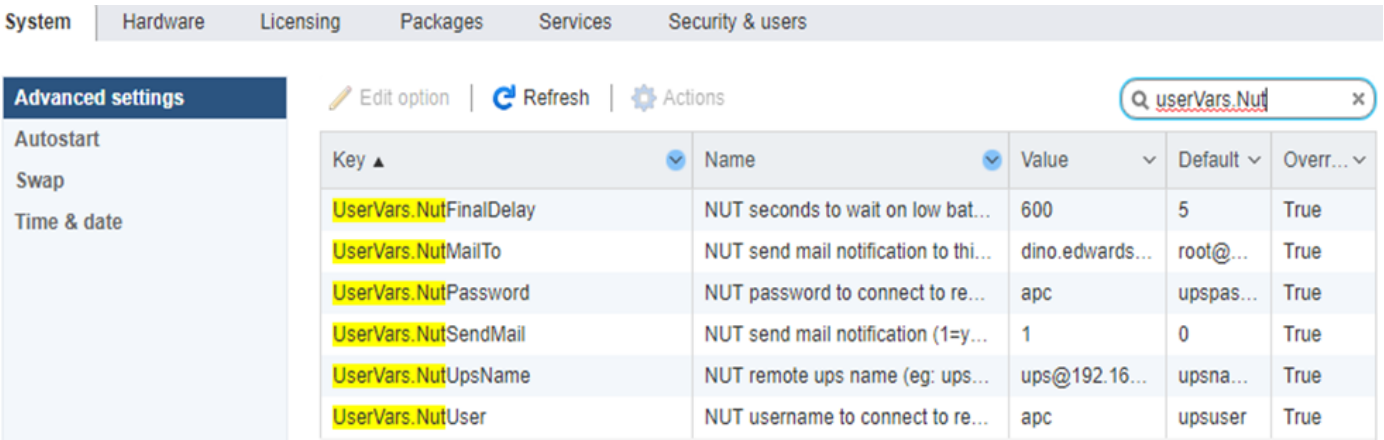
```
Installation Result

   Message: Operation finished successfully.

   Reboot Required: false

   VIBs Installed: Margar_bootbank_upsmon_2.7.4-2.1.0
   VIBs Removed:


   VIBs Skipped:
```

6. You can delete the files in the /tmp directory and disable the SSH service if desired.

ESXi Configuration

1. In the ESXi Web client, navigate to **Host —> Manage —> System —> Advanced Settings**. In the **Search** box enter **UserVars**.**Nut (Figure 1).**

Figure 1



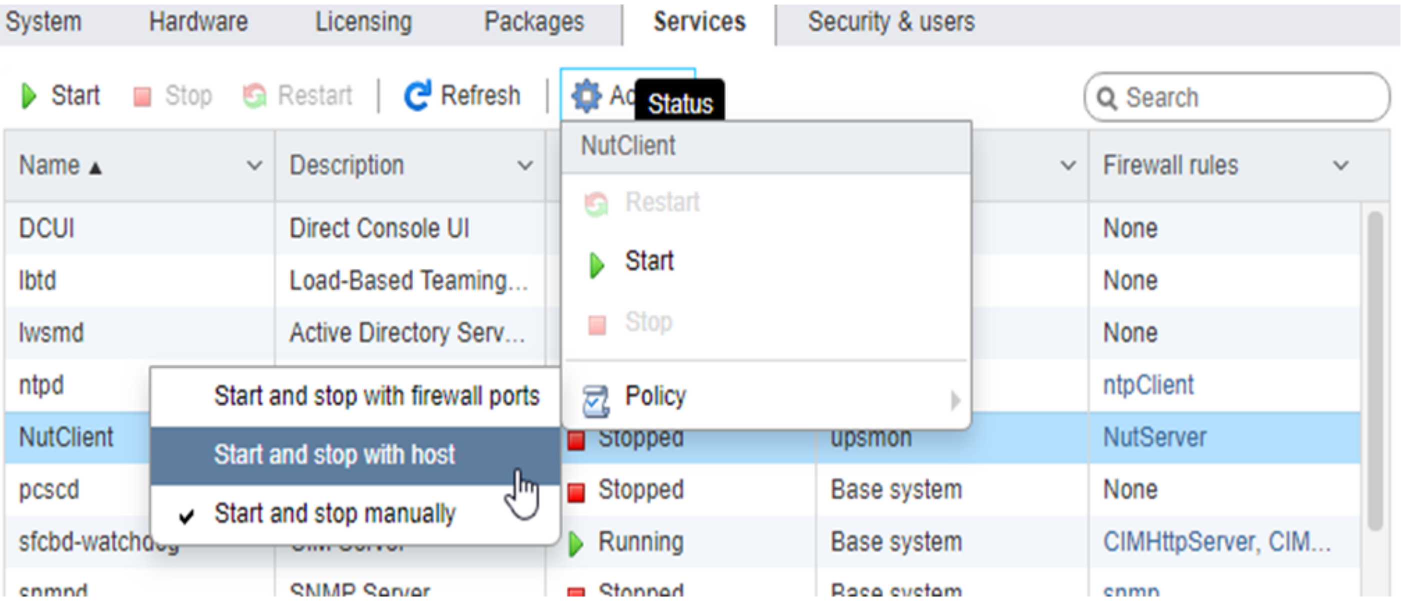
Key ▲	Name	Value	Default	Overr...
UserVars.NutFinalDelay	NUT seconds to wait on low bat...	600	5	True
UserVars.NutMailTo	NUT send mail notification to thi...	dino.edwards...	root@...	True
UserVars.NutPassword	NUT password to connect to re...	apc	upspas...	True
UserVars.NutSendMail	NUT send mail notification (1=y...	1	0	True
UserVars.NutUpsName	NUT remote ups name (eg: ups...	ups@192.16...	upsna...	True
UserVars.NutUser	NUT username to connect to re...	apc	upsuser	True

2. Configure the following variables:

- **NutUpsName:** Name of the UPS on the NUT server (in the form of inverter_name@server_name or server_ip). Several inverters can be entered separated by a space. There will be no system shutdown until the last UPS still standing has given the shutdown command.
- **NutUser:** Name of the NUT server login account
- **NutPassword:** NUT Server Connection Account Password
- **NutFinalDelay:** Seconds to wait after receiving the low battery event to shut down the system
- **NutSendMail:** Set to 1 for the NUT client to send an e-mail to each important event of the UPS
- **NutMailTo:** E-mail address to send UPS events to

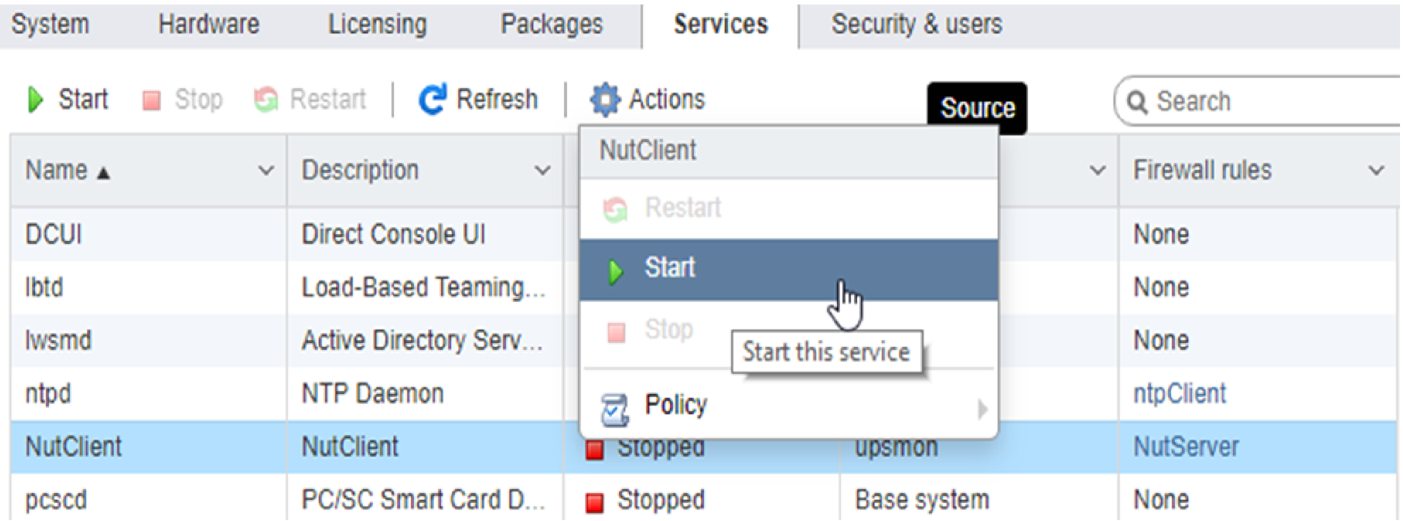
3. In the ESXi Web client, navigate to **Host —> Manage —> Services —> NutClient —> Actions —> Policy —> Start and Stop with Host (Figure 2).**

Figure 2



4. In the ESXi Web client, navigate to **Host —> Manage —> Services —> NutClient —> Actions —> Start (Figure 3).**

Figure 3



Tips

- Use the ESXi host configuration tab in the vSphere Client to decide how to start and stop (or suspend) virtual machines. This order will be respected by the UPS shutdown procedure.
- The clean shutdown of the OS in the virtual machines is only possible if the vmware tools are installed.
- To uninstall the NUT client, use the upsmon-remove script that is in the file that you downloaded:

```
/tmp # sh upsmon-remove
```

- To estimate the time needed for the server to shut down on UPS alert, type the command below on the host ESXi (by ssh or on the console). The shutdown procedure is immediately started:

```
/opt/nut/sbin/upsmon -c fsd
```

- If the NUT Client is configured correctly, the ESXi /var/log/syslog.log should have a message similar to below where ups@UPSHOST is the ups name and the UPS host you setup earlier :

```
2019-09-22T13:28:07Z upsmon[2111424]: Communications with UPS ups@UPSHOST established
```

Create OVA using vmware ovftool on ESXi or Vcenter

- Download ovftool for Windows 64 from vmware.com and install.
- From an elevated command prompt, navigate to **c:\Program Files\VMware\VMware OVF Tool**
- Run any of the following commands depending on your environment:

Vcenter Example (local)

```
ovftool --noSSLVerify vi://[USERNAME]@[VHOST]/[DATACENTER]/vm/[MACHINE_NAME]
C:\ova\[OVA_MACHINE_NAME].ova
```

ESXi Example (Share):

```
ovftool --noSSLVerify vi://[USERNAME]@[ESXI_HOST]/[MACHINE_NAME]
\\[FILE_SERVER]\[SHARE]\[MACHINE_NAME].ova
```

In-Guest UNMAP, EnableBlockDelete and VMFS-6

Original Article URL: https://www.codyhosterman.com/2017/08/in-guest-unmap-enableblockdelete-and-vmfs-6/

Credit: Cody Hosterman

EnableBlockDelete with VMFS-5

I have a Ubuntu VM with a thin virtual disk on a VMFS-5 volume.

Image not found or type unknown

Image not found or type unknown

Furthermore, I have EnableBlockDelete DISABLED on the ESXi host (set to 0). It is important to note that this is a host-wide setting.

Image not found or type unknown

In my VM, I will put ext4 on the virtual disk then mount it:

Image not found or type unknown

We can see through sg_vpd that UNMAP is supported on this virtual disk:

```
root@Ubuntu16:~# sg_vpd /dev/sdb -p lbpv |grep "Unmap"
  Unmap command supported (LBPU): 1
```

Now I will put some data on the file system. A couple of OVAs.

Image not found or type unknown

```
root@Ubuntu16:/mnt/unmap# df -h /mnt/unmap
Filesystem Size Used Avail Use% Mounted on
/dev/sdb 16G 3.9G 11G 26% /mnt/unmap
```

My file system reports as having 3.9 GB used. My VMDK is 4.4 GB in size.

Image not found or type unknown

There is about 400 MB of capacity that was written when the file system was created, which explains the difference between those.

The underlying array reports 3.7 GB used. Smaller due to data reduction. Since the OVAs are compressed already, there isn’t a ton of data reduction to do.

Image not found or type unknown

Okay, so let’s delete the OVAs.

Image not found or type unknown

We can see the file system is now down to 44 MB used:

```
root@Ubuntu16: /mnt/unmap# df -h /mnt/unmap
Filesystem Size Used Avail Use% Mounted on
/dev/sdb 16G 44M 15G 1% /mnt/unmap
```

But if we look at the VMDK, it is still 4.4 GB:

Image not found or type unknown

And the array is unchanged too.

Image not found or type unknown

So we now have dead space in the VMDK and on the array, because those blocks are no longer in use by the guest. So, in Linux, to reclaim space you can either mount the file system with the discard option so UNMAP is triggered immediately upon file deletion, or you can manually run it with fstrim. I did not mount with the discard option, so I will run fstrim.

Image not found or type unknown

```
root@Ubuntu16:/mnt/unmap# fstrim /mnt/unmap -v
/mnt/unmap: 3.9 GiB (4131360768 bytes) trimmed
```

Now if we look at my VMDK, we will see it has shrunk to 400 MB:

Image not found or type unknown

But my array is still reporting it as used. This is because EnableBlockDelete is not turned on. The UNMAP in the guest only makes the VMDK size accurate by shrinking it down. But the underlying physical device is not told.

Image not found or type unknown

So at this point (since it is VMFS-5) I have to run esxcli storage vmfs unmap to reclaim it.

```
esxcli storage vmfs unmap -l vmfs5
```

Image not found or type unknown

Once complete, we can see the capacity reclaimed on the array:

So this is the default behavior. Let’s enable EnableBlockDelete and repeat the process.

Image not found or type unknown

I will copy the data back to the file system, which will grow the virtual disk again and write data back to the FlashArray. We can see we have 3.9 GB used again on my file system.

```
root@Ubuntu16: /mnt/unmap# df -h /mnt/unmap
Filesystem Size Used Avail Use% Mounted on
/dev/sdb 16G 3.9G 11G 26% /mnt/unmap
```

My virtual disk is back to 4.4 GB:

Image not found or type unknown

My array reduced it to 1.7 GB:

So now to delete the data and run fstrim. My virtual disk shrinks to 400ish MB again:

Image not found or type unknown

My space on my array is reclaimed immediately this time! No need to run esxcli to unmap.

Image not found or type unknown

I ran fstrim at 12:33:00 and the space was reclaimed on the array automatically by 12:33:55.

Great! So now, back to the original question, what about VMFS-6?

EnableBlockDelete and VMFS-6

As you are likely aware, VMFS-6 introduced automatic UNMAP. So you no longer need to ever use esxcli to run UNMAP on the VMFS.

So let’s repeat the test.

I moved my VMDK to my VMFS-6 datastore:

I will not go through every step again, let’s just start from the “we just deleted the files” step, but we have yet to run fstrim. So we have dead space.

VMFS-6: EnableBlockDelete Disabled, Auto-UNMAP Enabled

In this test, I have EnableBlockDelete disabled on my host and auto-UNMAP enabled on the datastore.

Image not found or type unknown

Image not found or type unknown

If I use vsish, I can see no automatic UNMAPs have been issued to this datastore from my host.

Image not found or type unknown

Note “UNMAP IOs” and “Unmapped blocks” are both zero.

So I run fstrim. My VMDK is back down to 400 MB:

Image not found or type unknown

If we look back at the array, we see the space reclaims, but not quite as fast, took a few minutes.

Image not found or type unknown

Since EnableBlockDelete was disabled and auto-unmap was enabled we can see this was auto-unmap. We can further show that by looking back at vsish:

Image not found or type unknown

62 UNMAP I/Os and 3878 blocks reclaimed. So we don’t need to turn on EnableBlockDelete in the case of VMFS-6!

VMFS-6: EnableBlockDelete Enabled, Auto-UNMAP Disabled

In this test, I have EnableBlockDelete enabled on my host…

Image not found or type unknown

…and auto-UNMAP disabled on the datastore:

Image not found or type unknown

Let’s run through the process again. I refreshed my environment so counters are reset etc. Add the VMDK, put data on it, delete the data and run fstrim:

The VMDK shrank back down:

But if we look at the array, nothing happens.

So this shows that EnableBlockDelete is ignored for VMFS-6 volumes. So in this situation we would have to enable automatic UNMAP to reclaim this space, or run the standard esxcli manual UNMAP.

Conclusion

So what does this tell us. A couple things.

- In order to have full end-to-end UNMAP with VMFS-5 volumes, you need to enable EnableBlockDelete.
- For VMFS-6 automatic UNMAP takes care of the VMFS reclamation portion for you.

An interesting thing here is that automatic UNMAP invokes fairly quickly. When you delete a VM or a virtual disk, automatic UNMAP can possibly take 12-24 hours to reclaim the space. But with in-guest UNMAP, as soon as the VMDK shrinks, automatic UNMAP kicks in fairly quickly–in a few minutes. Mimicking the behavior of EnableBlockDelete. Which is great–you don’t lose functionality by moving to VMFS-6.

I will note, that this was done with 6.5 U1. From my understanding there was a bug in 6.5.0 that EnableBlockDelete was actually honored with VMFS-6 and it would issue UNMAP when a VMDK shrank when the setting was enabled. The problem was that UNMAP was issued twice, as the EnableBlockDelete-invoked UNMAP did not prevent the automatic async UNMAP from issuing reclaim. So UNMAP was issued twice.

This behavior was changed in 6.5 P1 and of course in 6.5 U1.