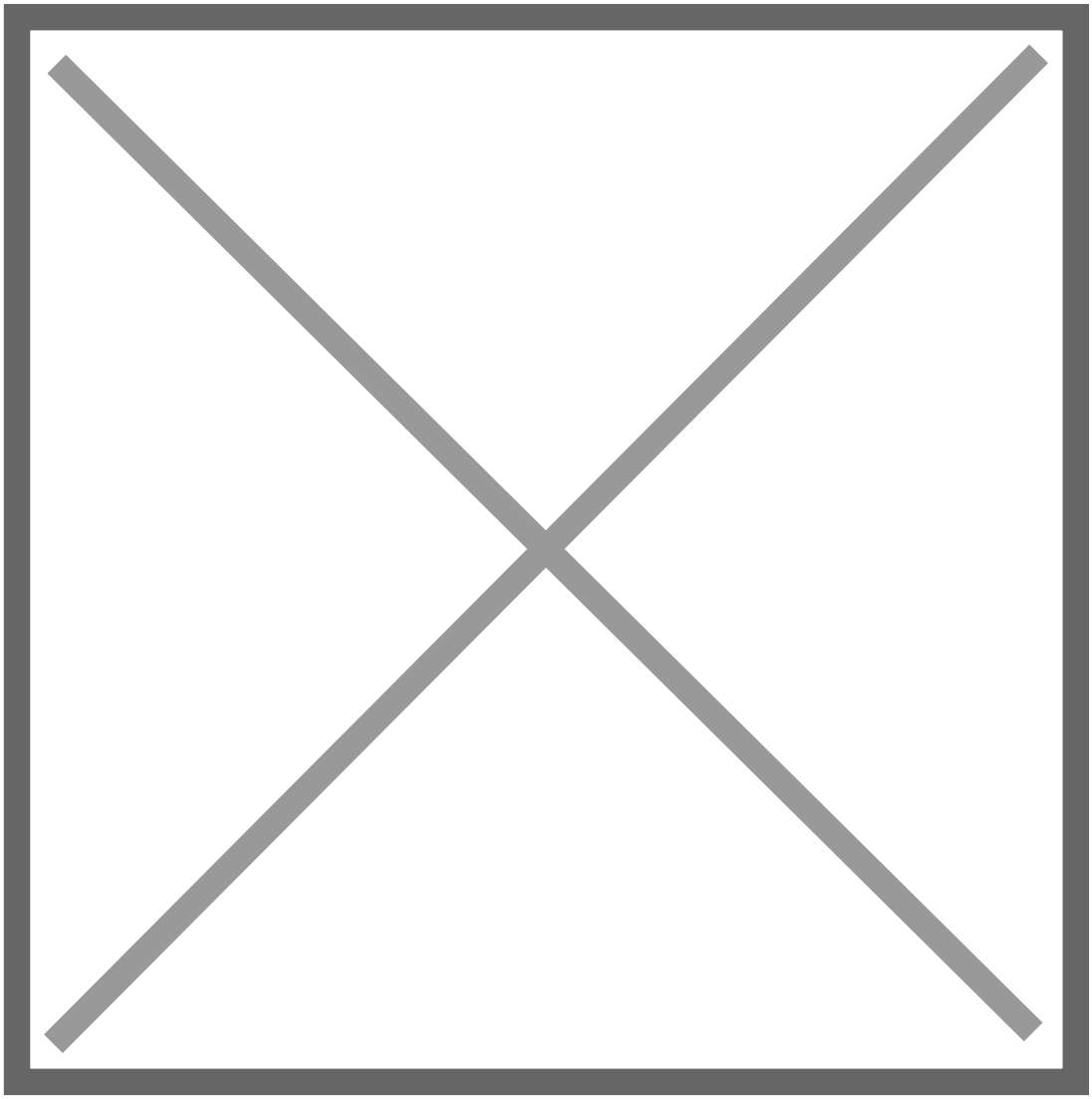
In-Guest UNMAP, EnableBlockDelete and VMFS-6

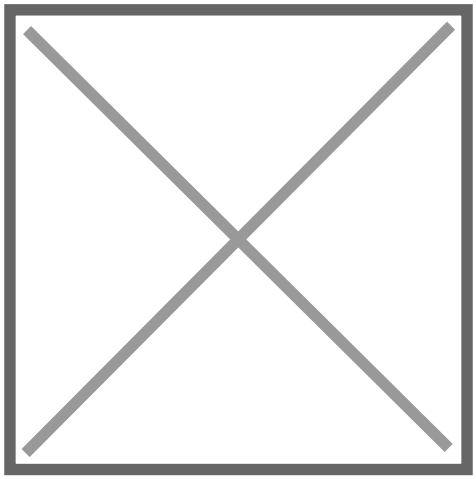
> Original Article URL: https://www.codyhosterman.com/2017/08/in-guest-unmap-enableblockdelete-and-vmfs-6/

> Credit: Cody Hosterman

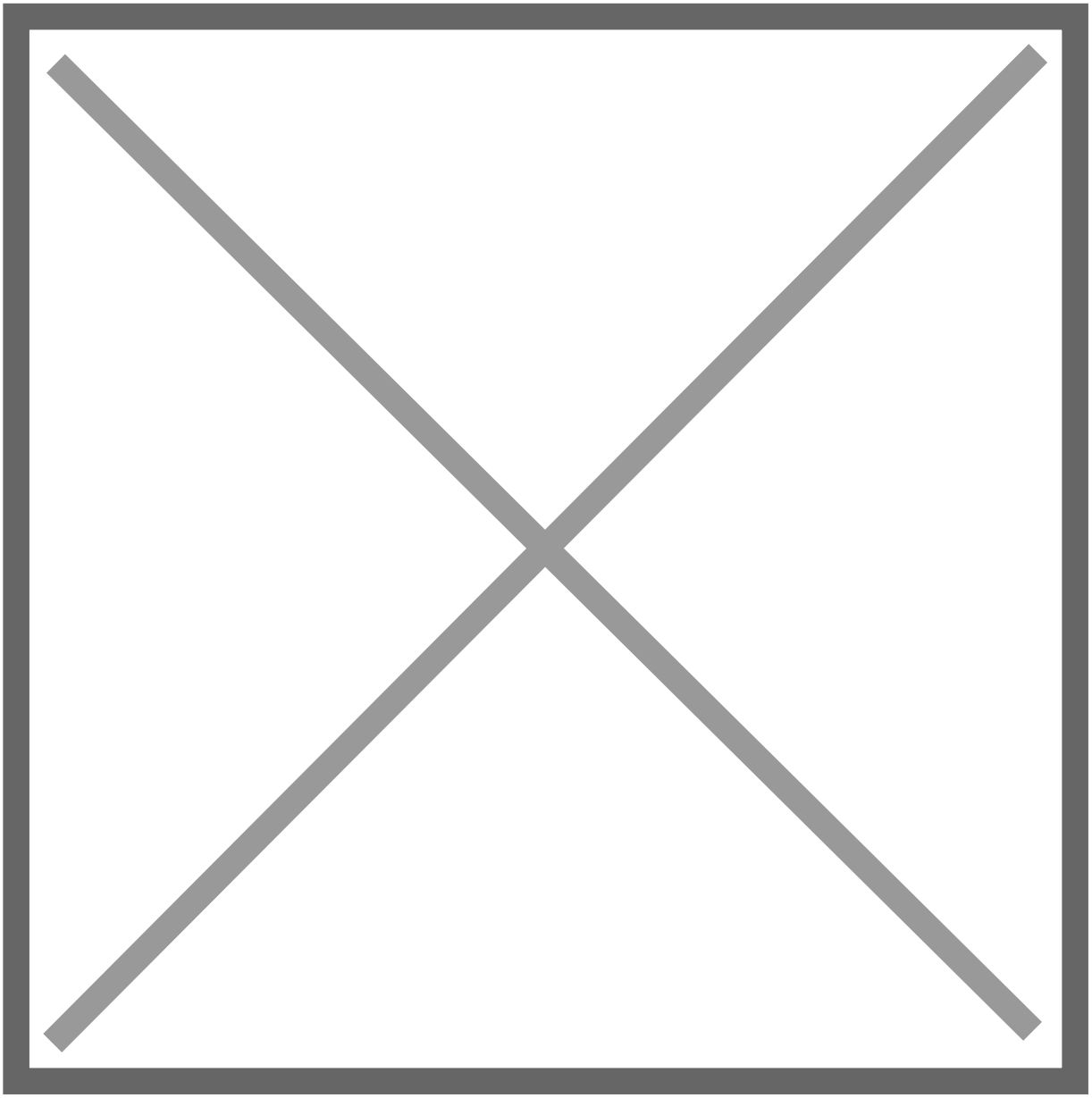
EnableBlockDelete with VMFS-5

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

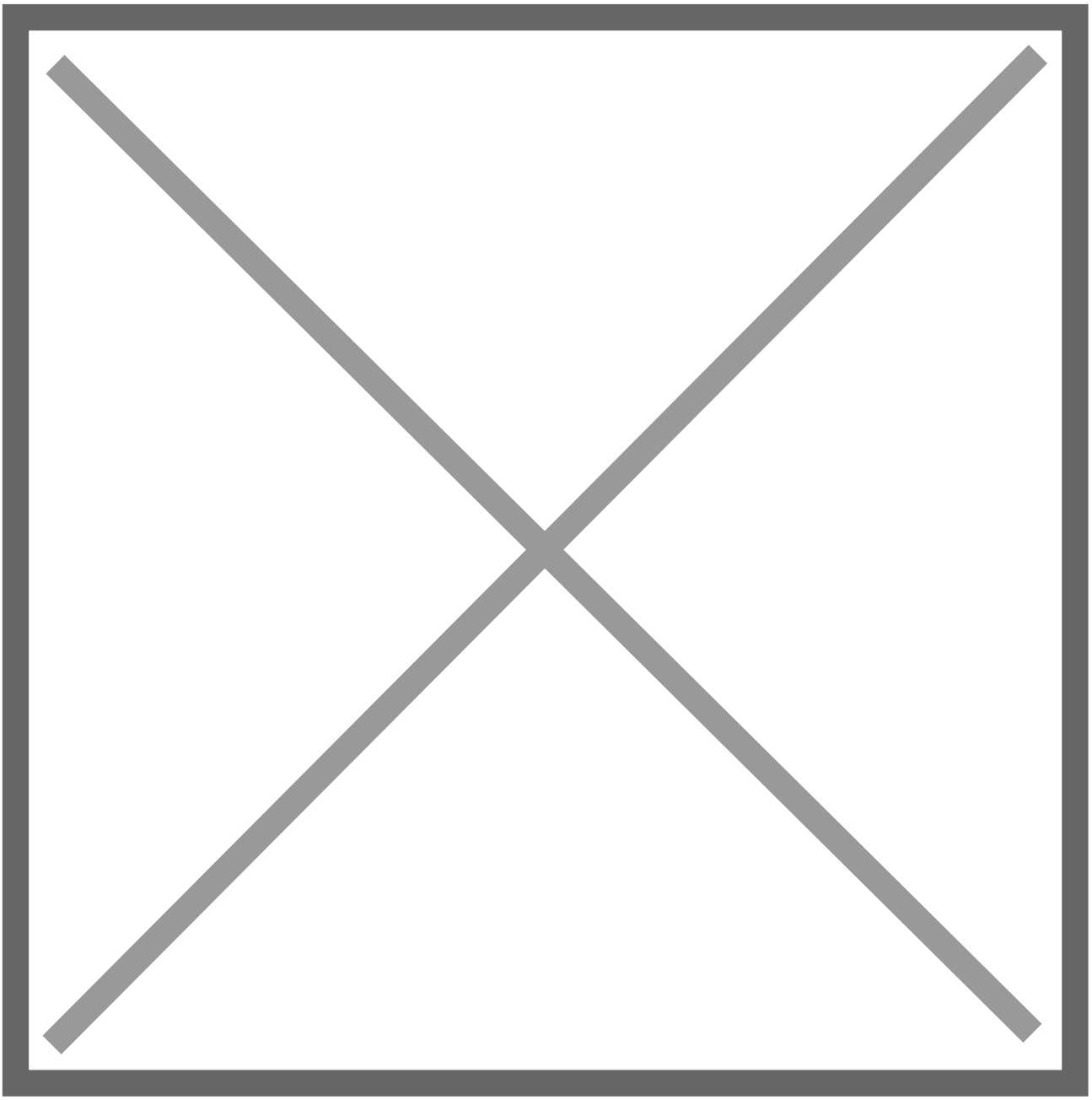




Furthermore, I have EnableBlockDelete DISABLED on the ESXi host (set to 0). It is important to note that this is a host-wide setting.



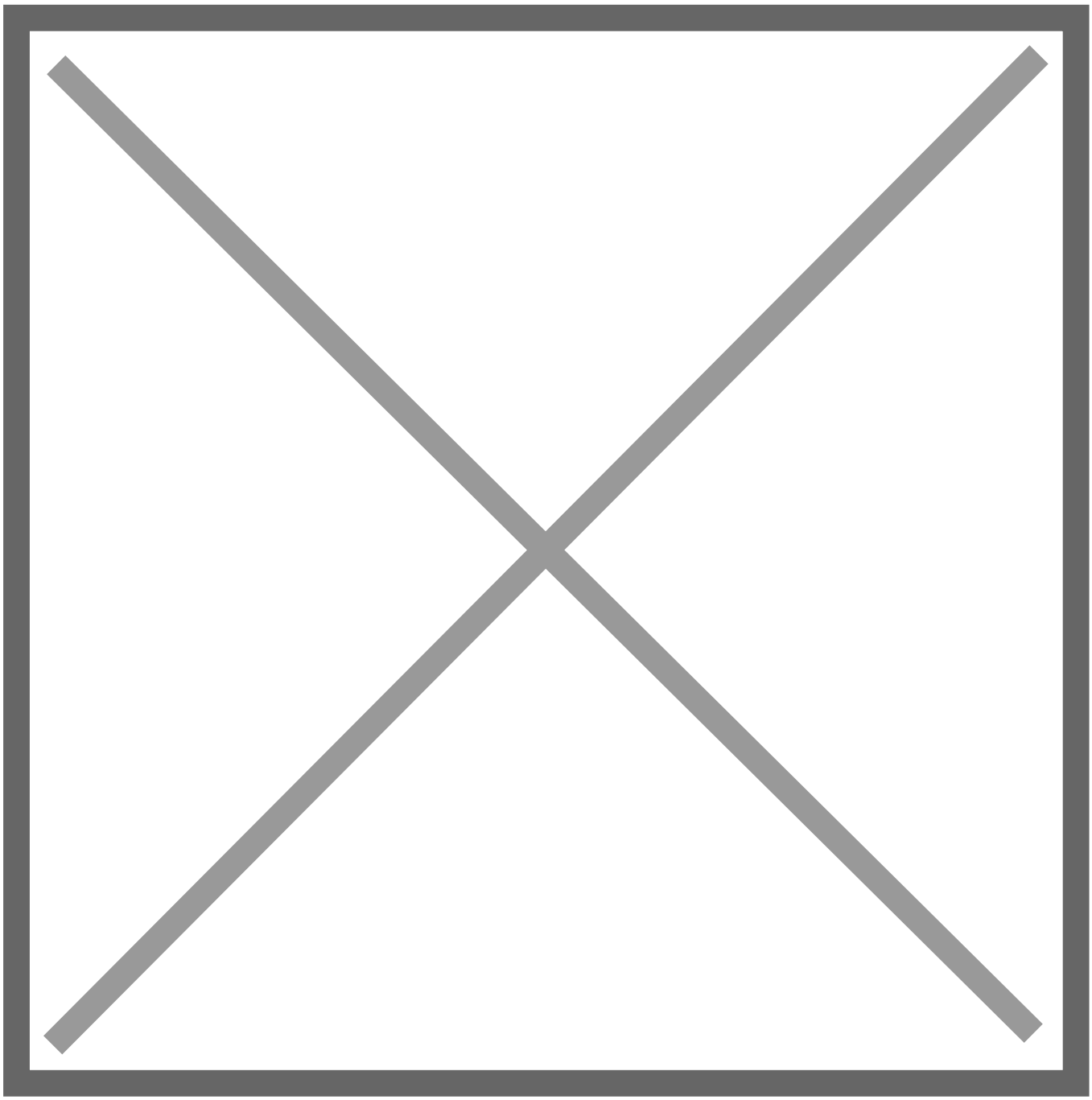
In my VM, I will put ext4 on the virtual disk then mount it:



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

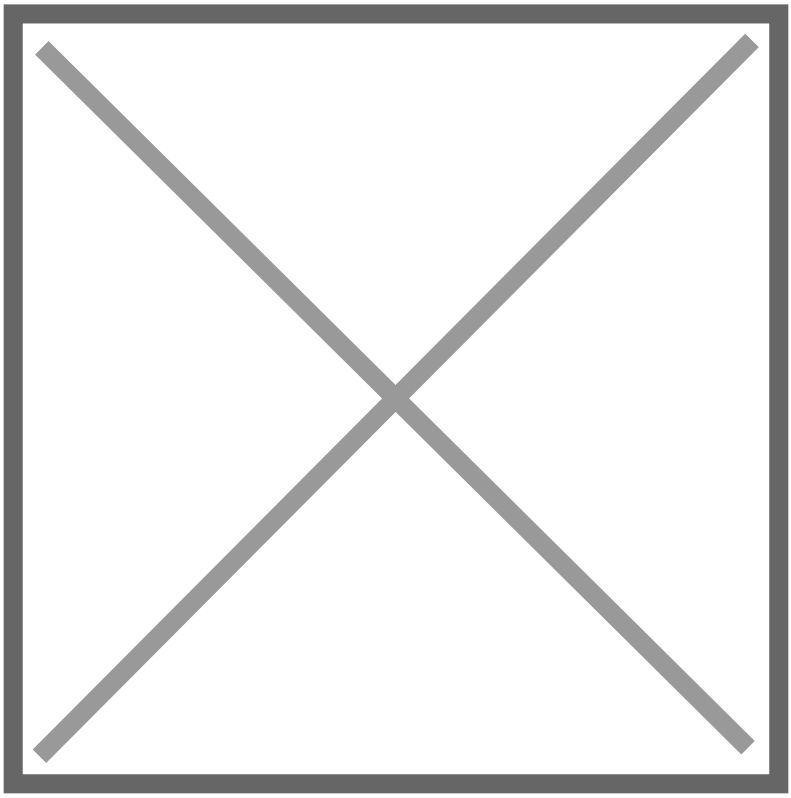
```
root@Ubuntu16:~# sg_vpd /dev/sdb -p lbpv |grep "Unmap"
 Unmap command supported (LBPU): 1
```

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



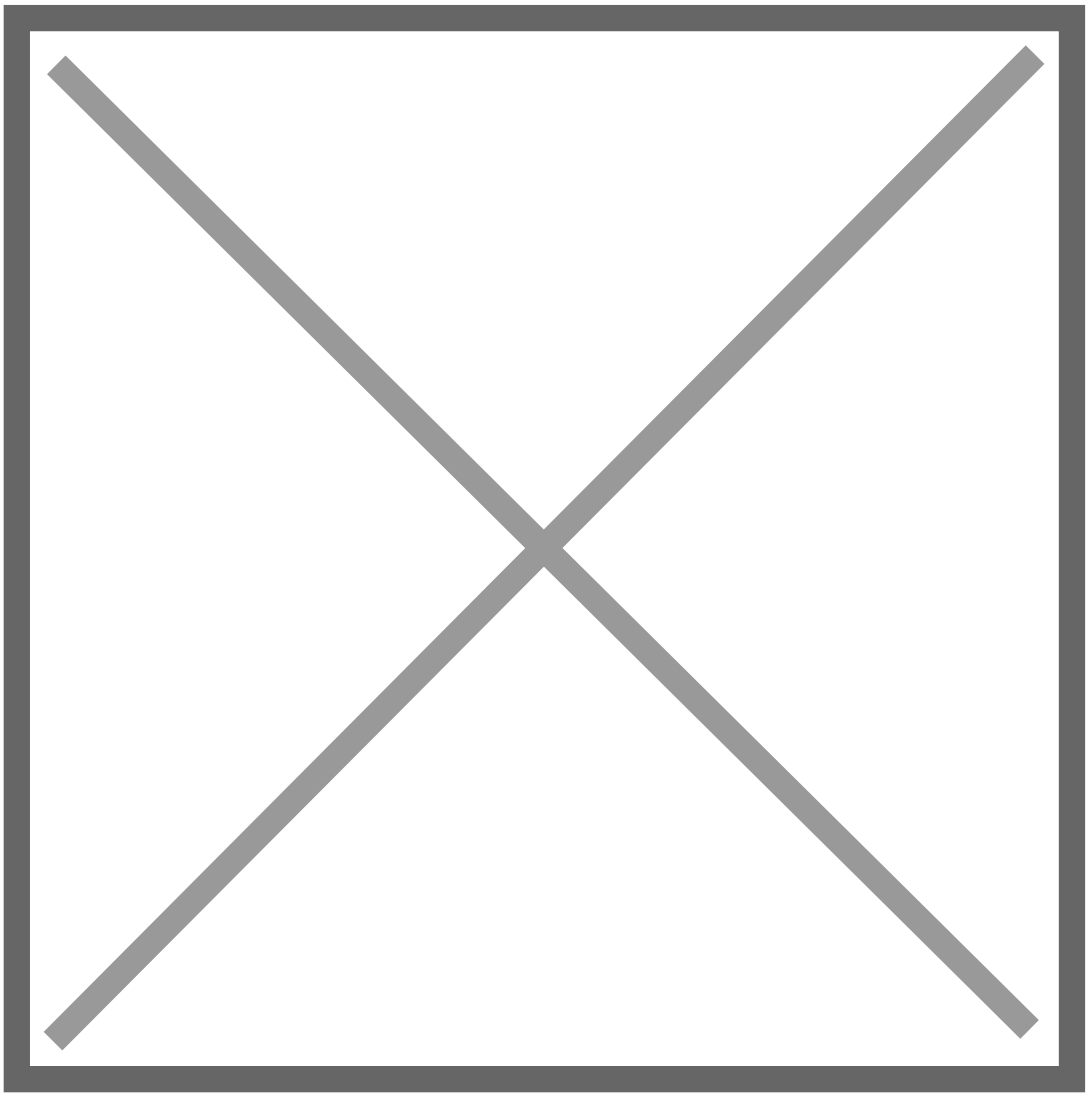
```
root@Ubuntu16:/mnt/unmap# df -h /mnt/unmap
Filesystem Size Used Avail Use% Mounted on
/dev/sdb 16G 3.9G 11G 26% /mnt/unmap
```

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

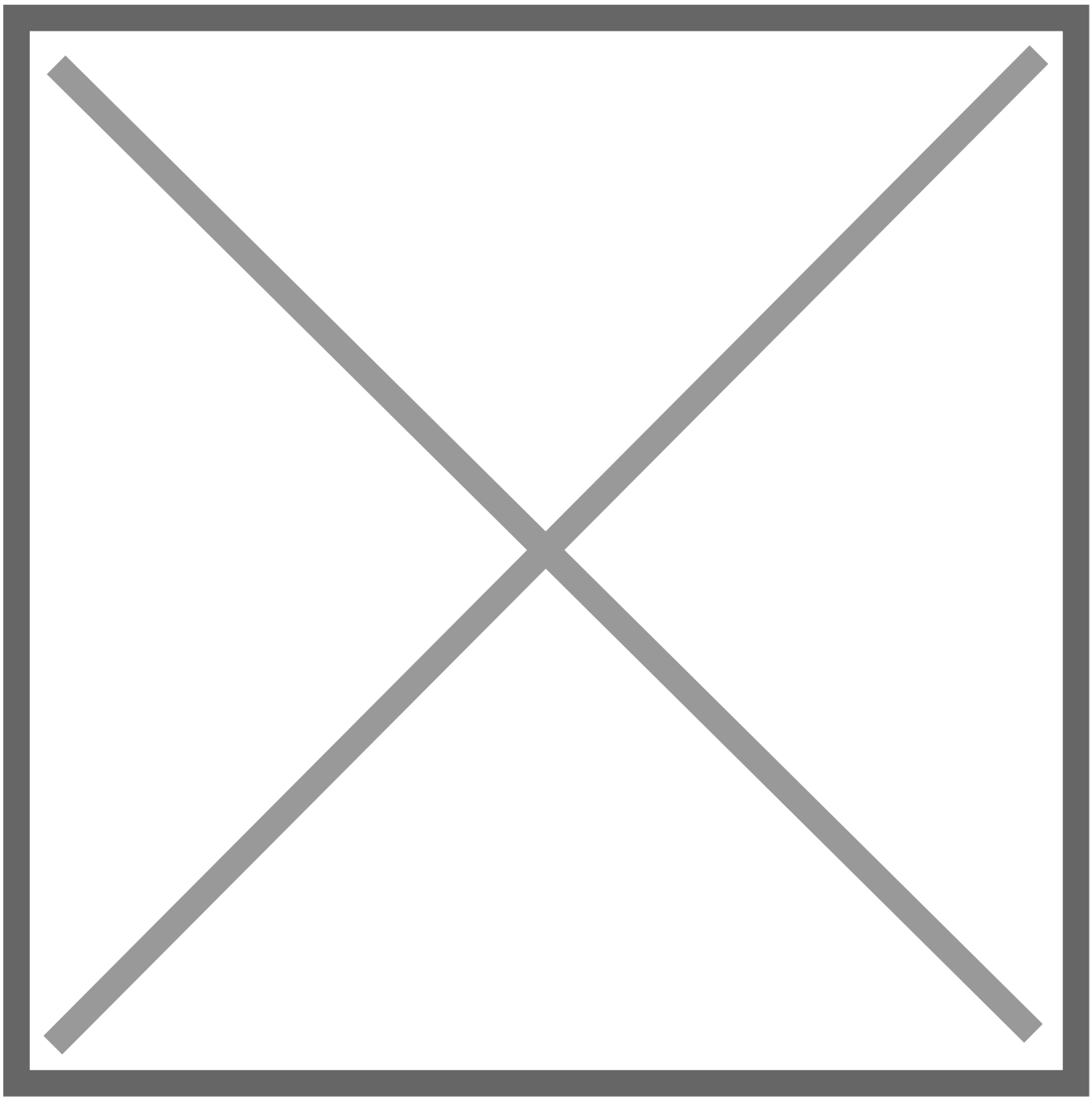


There is about 400 MB of capacity that was written when the file system was created, which explains the difference between those.

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.



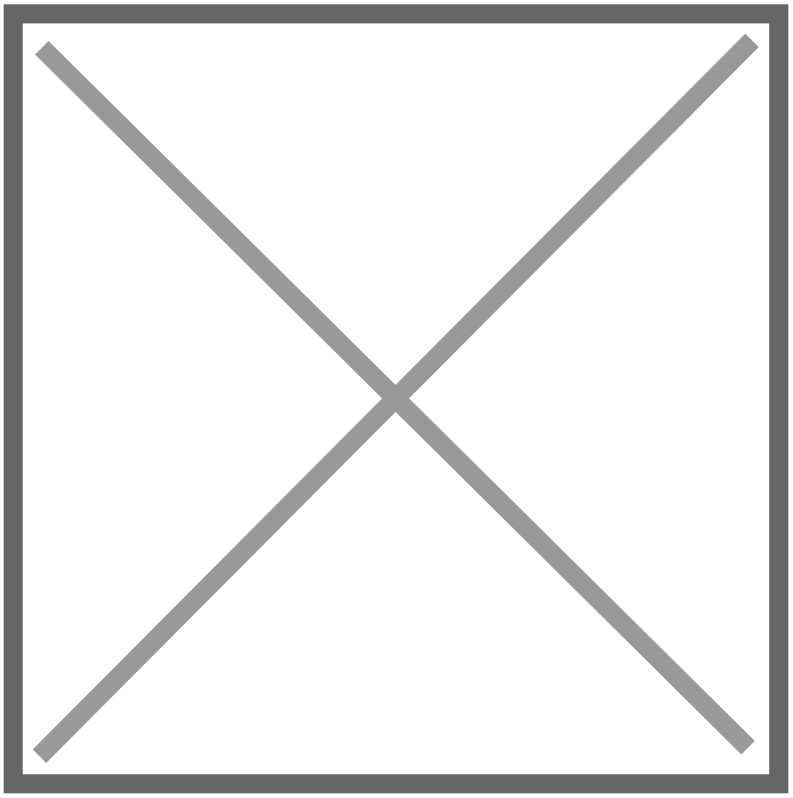
Okay, so let’s delete the OVAs.



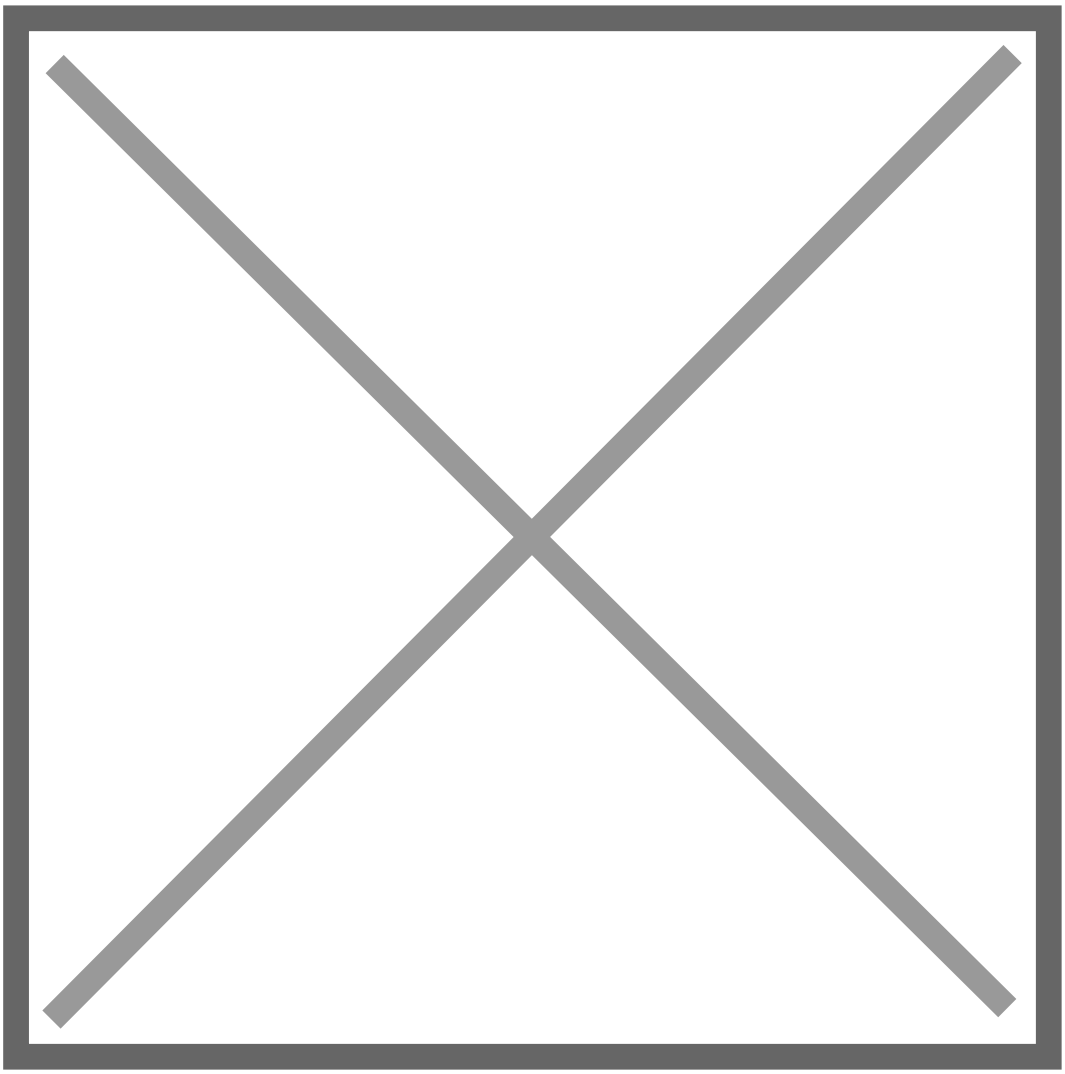
We can see the file system is now down to 44 MB used:

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

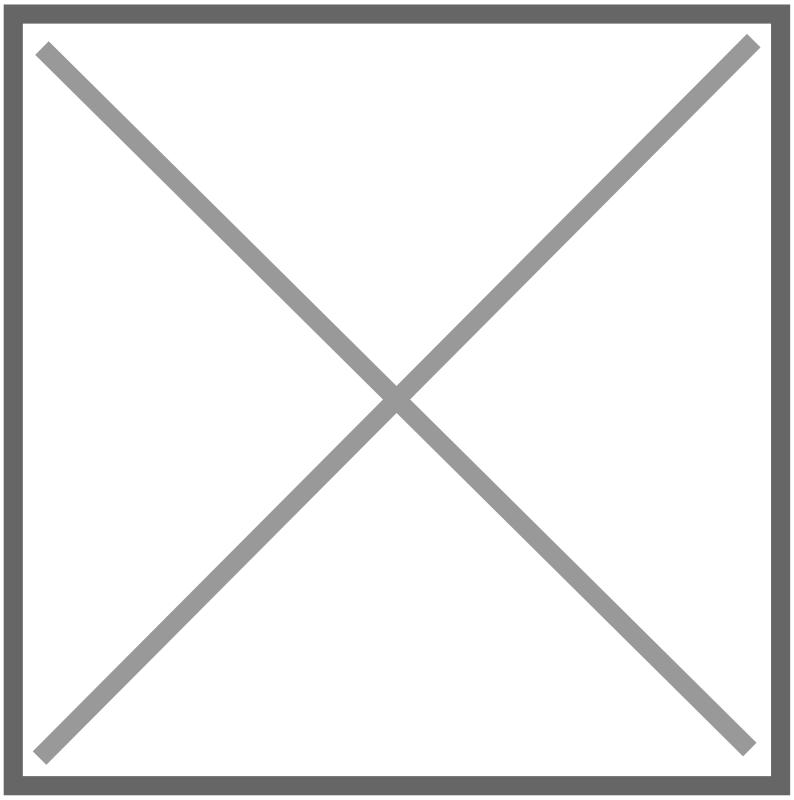
But if we look at the VMDK, it is still 4.4 GB:



And the array is unchanged too.

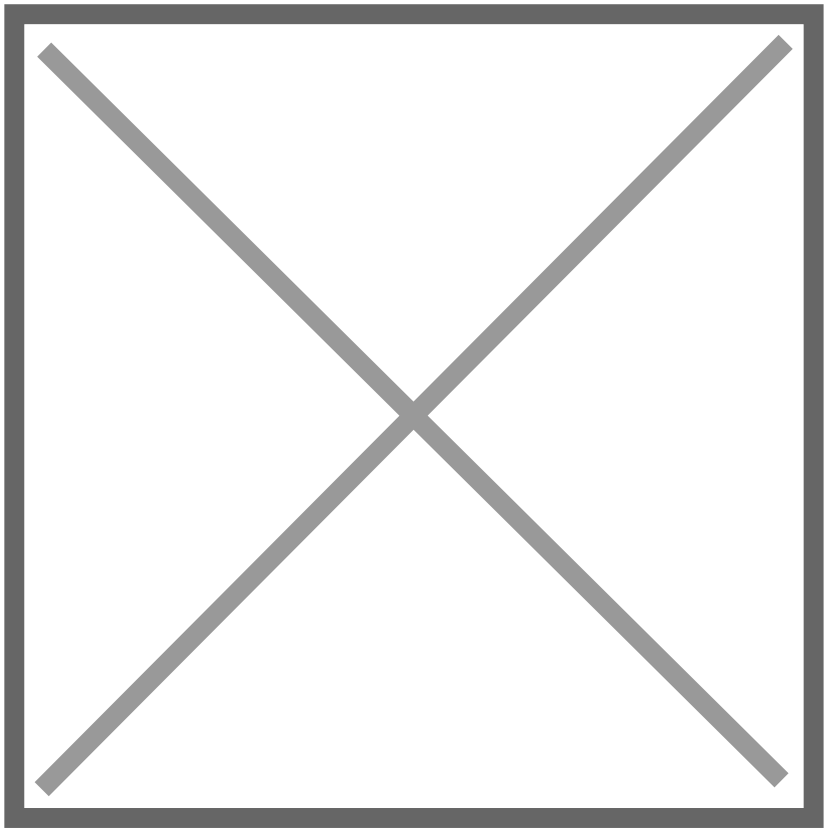


So we now have dead space in the VMDK and on the array, because those blocks are no longer in use by the guest. So, in Linux, to reclaim space you can either mount the file system with the discard option so UNMAP is triggered immediately upon file deletion, or you can manually run it with fstrim. I did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

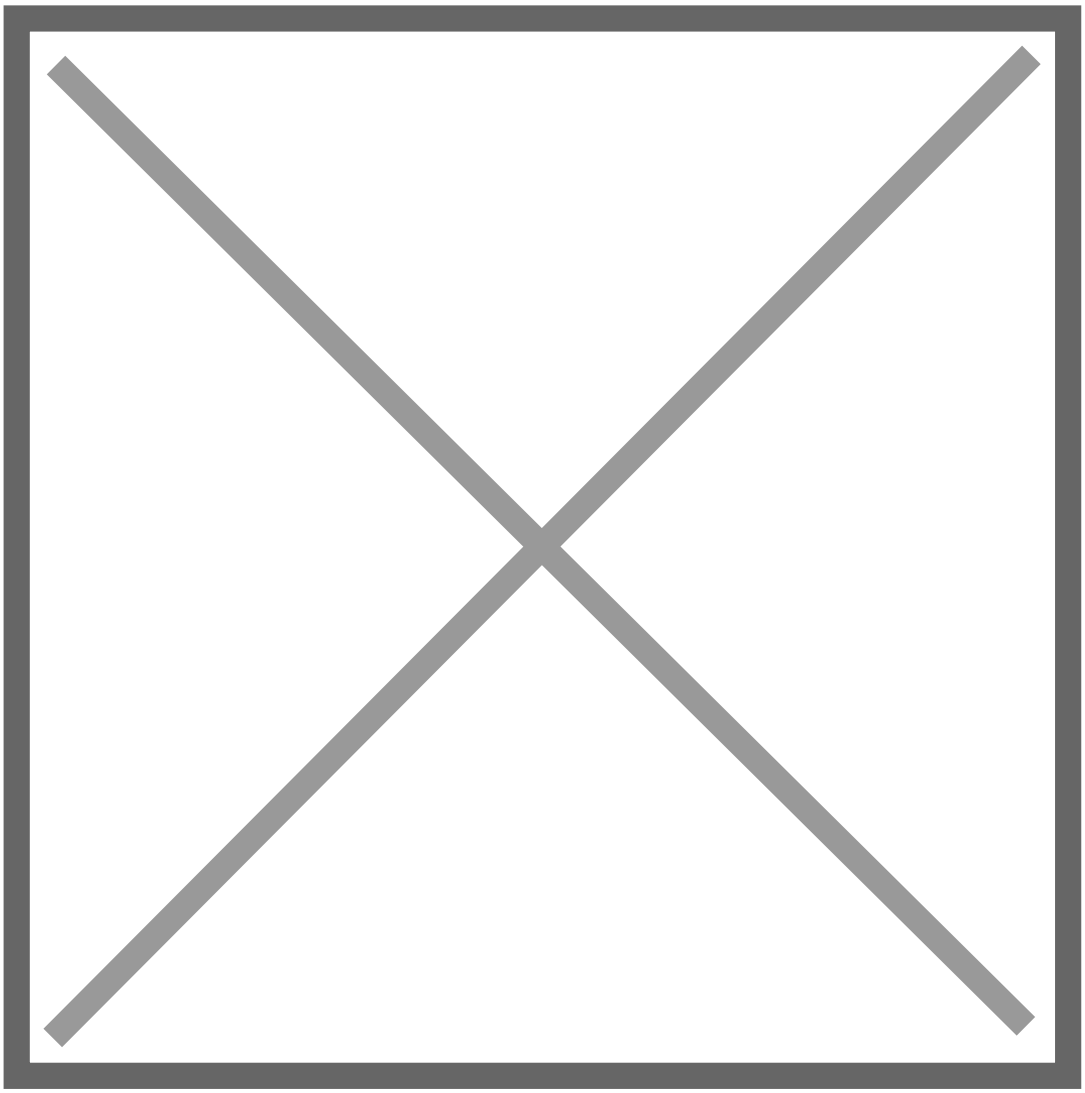


```
root@Ubuntu16:/mnt/unmap# fstrim /mnt/unmap -v
/mnt/unmap: 3.9 GiB (4131360768 bytes) trimmed
```

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

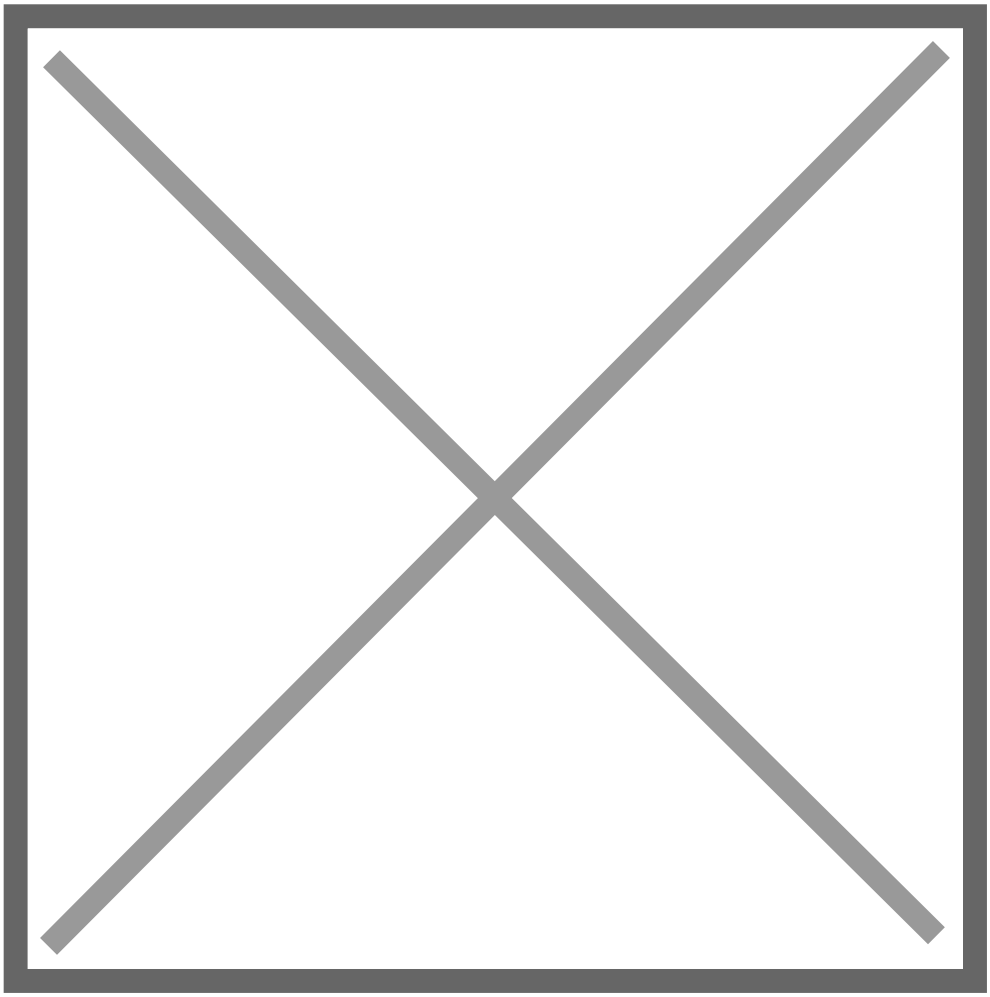


But my array is still reporting it as used. This is because EnableBlockDelete is not turned on. The UNMAP in the guest only makes the VMDK size accurate by shrinking it down. But the underlying physical device is not told.

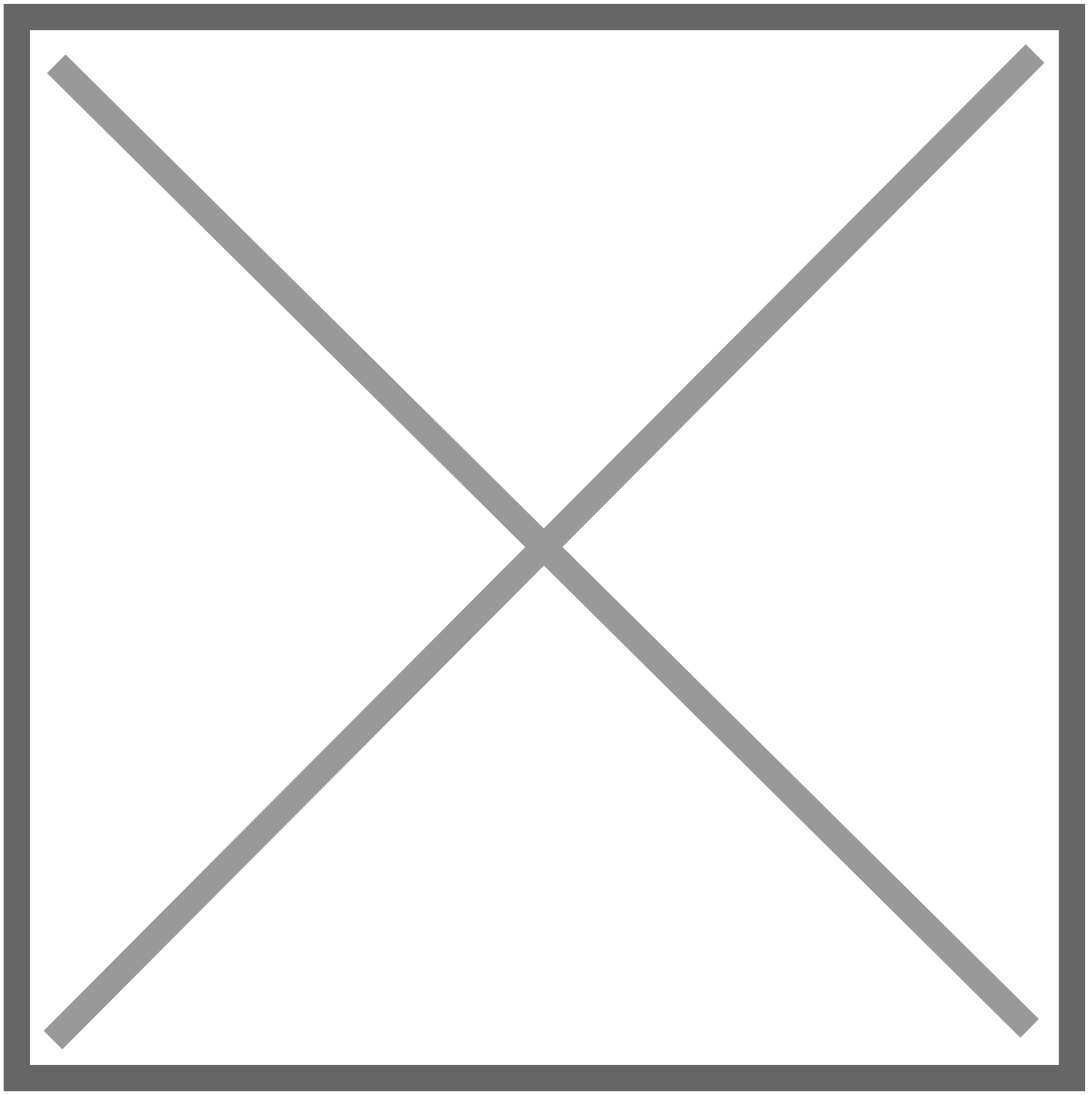


So at this point (since it is VMFS-5) I have to run esxcli storage vmfs unmap to reclaim it.

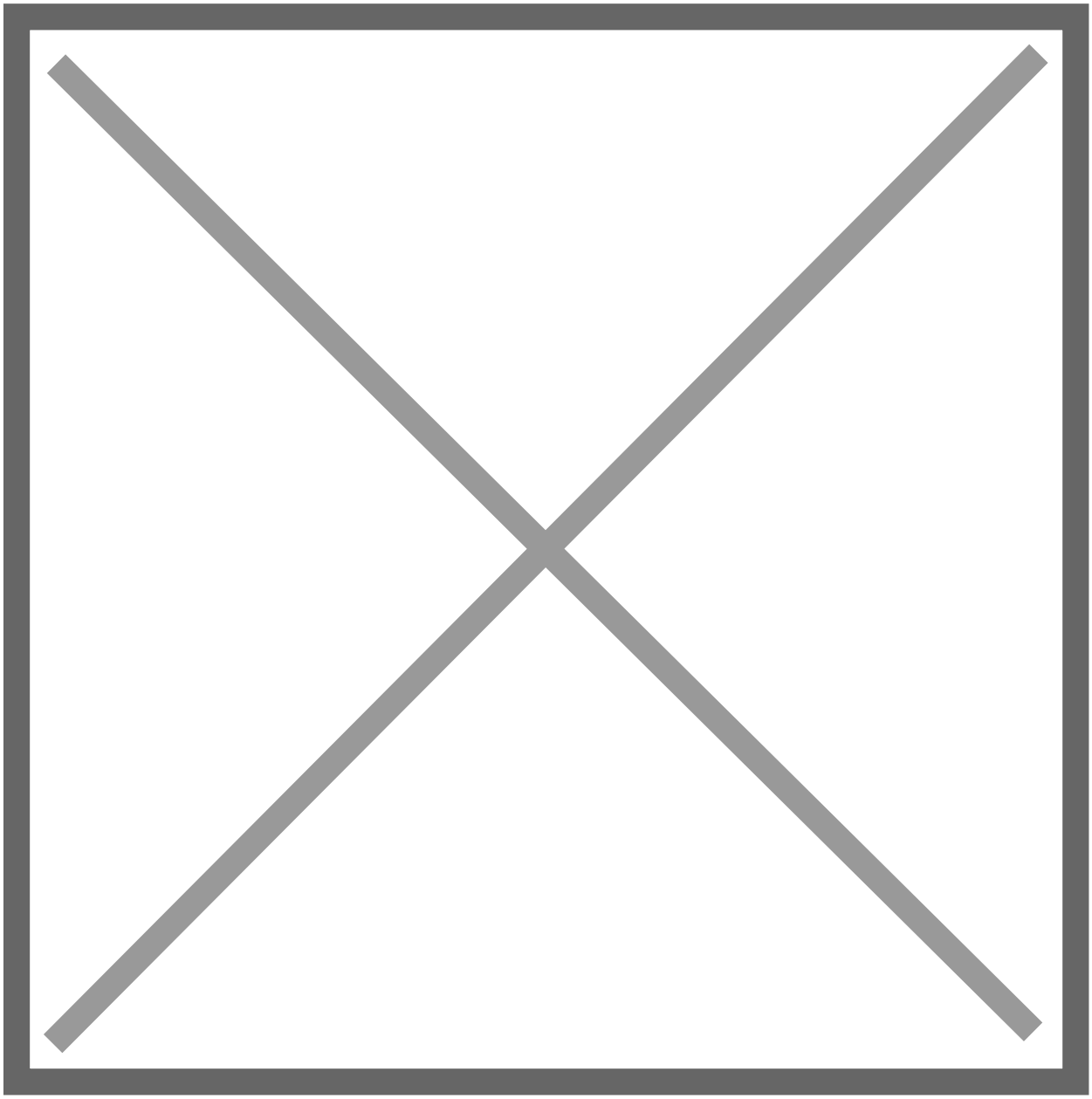
```
esxcli storage vmfs unmap -l vmfs5
```



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



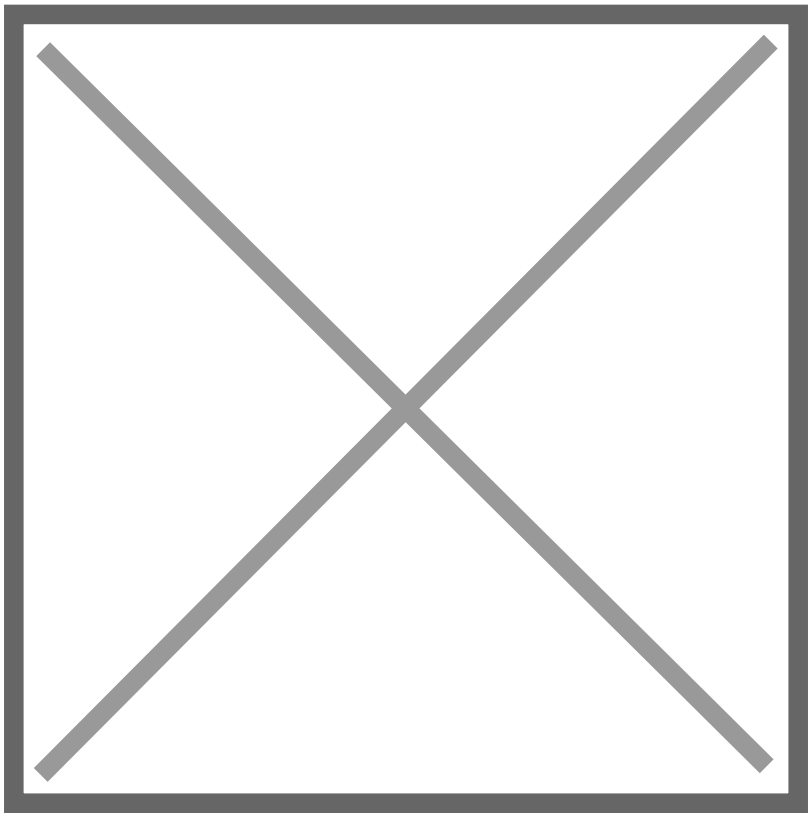
So this is the default behavior. Let’s enable EnableBlockDelete and repeat the process.



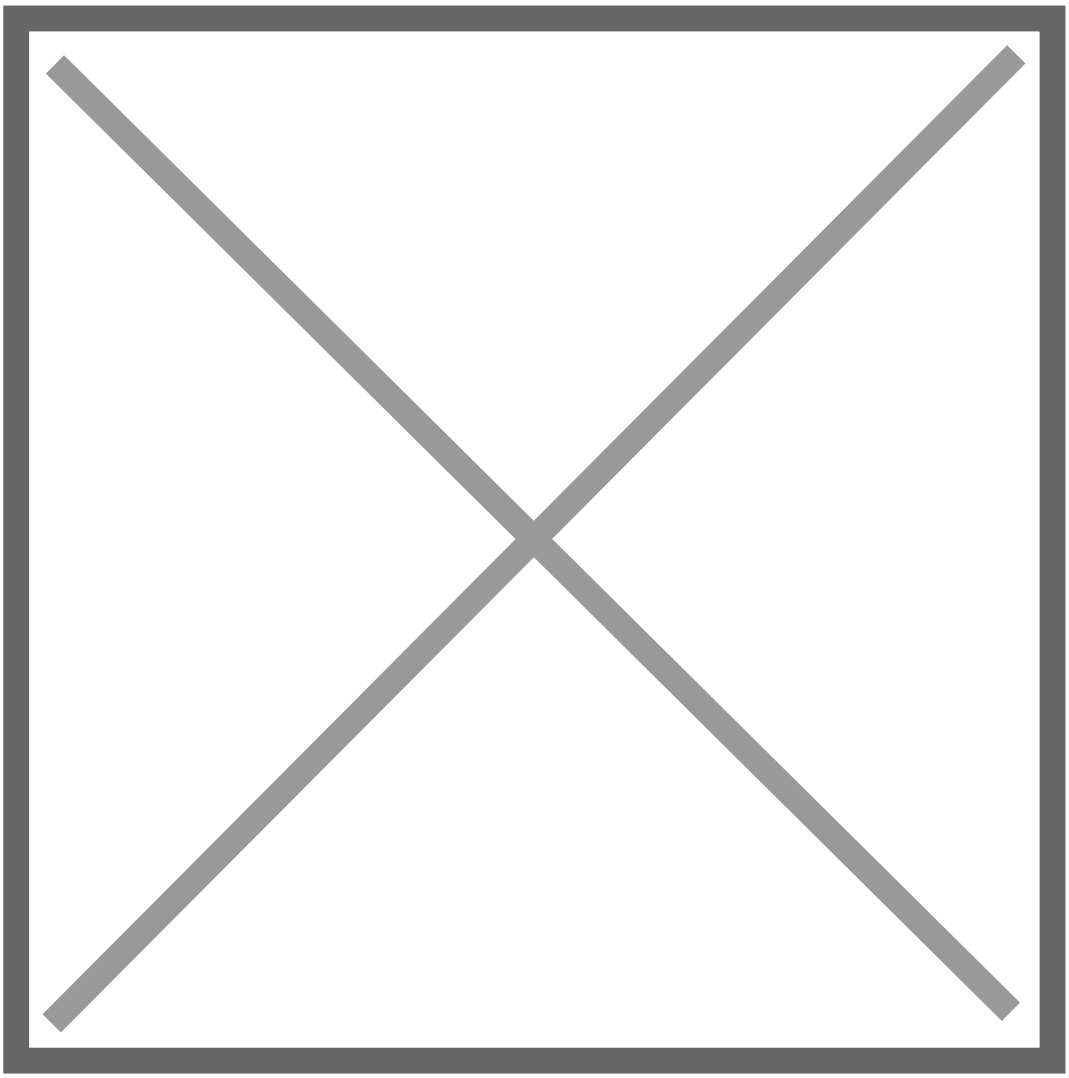
I will copy the data back to the file system, which will grow the virtual disk again and write data back to the FlashArray. We can see we have 3.9 GB used again on my file system.

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

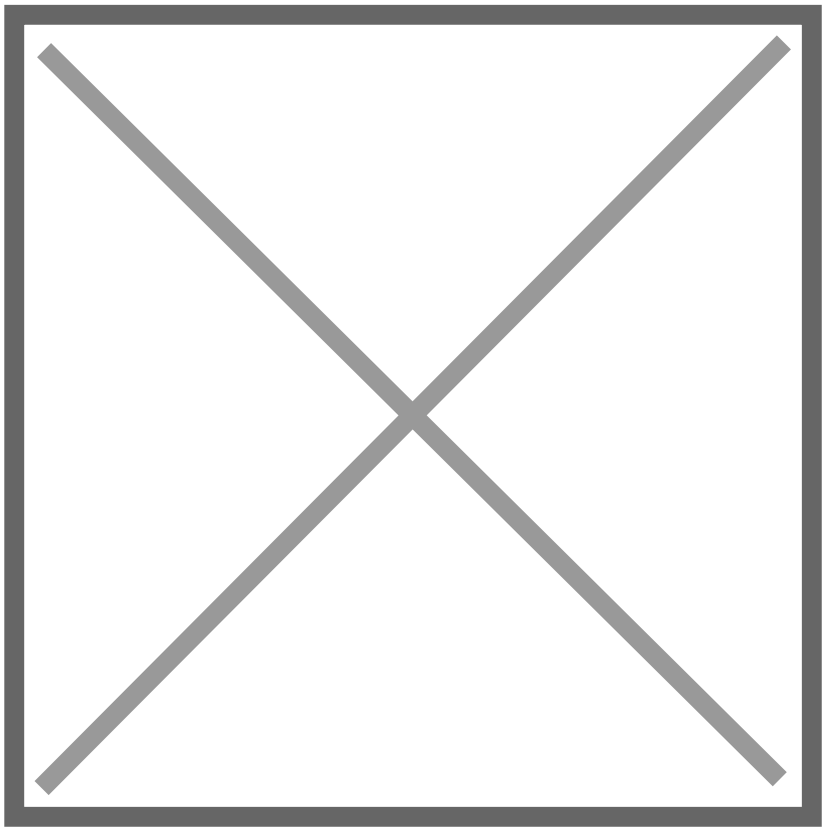
My virtual disk is back to 4.4 GB:



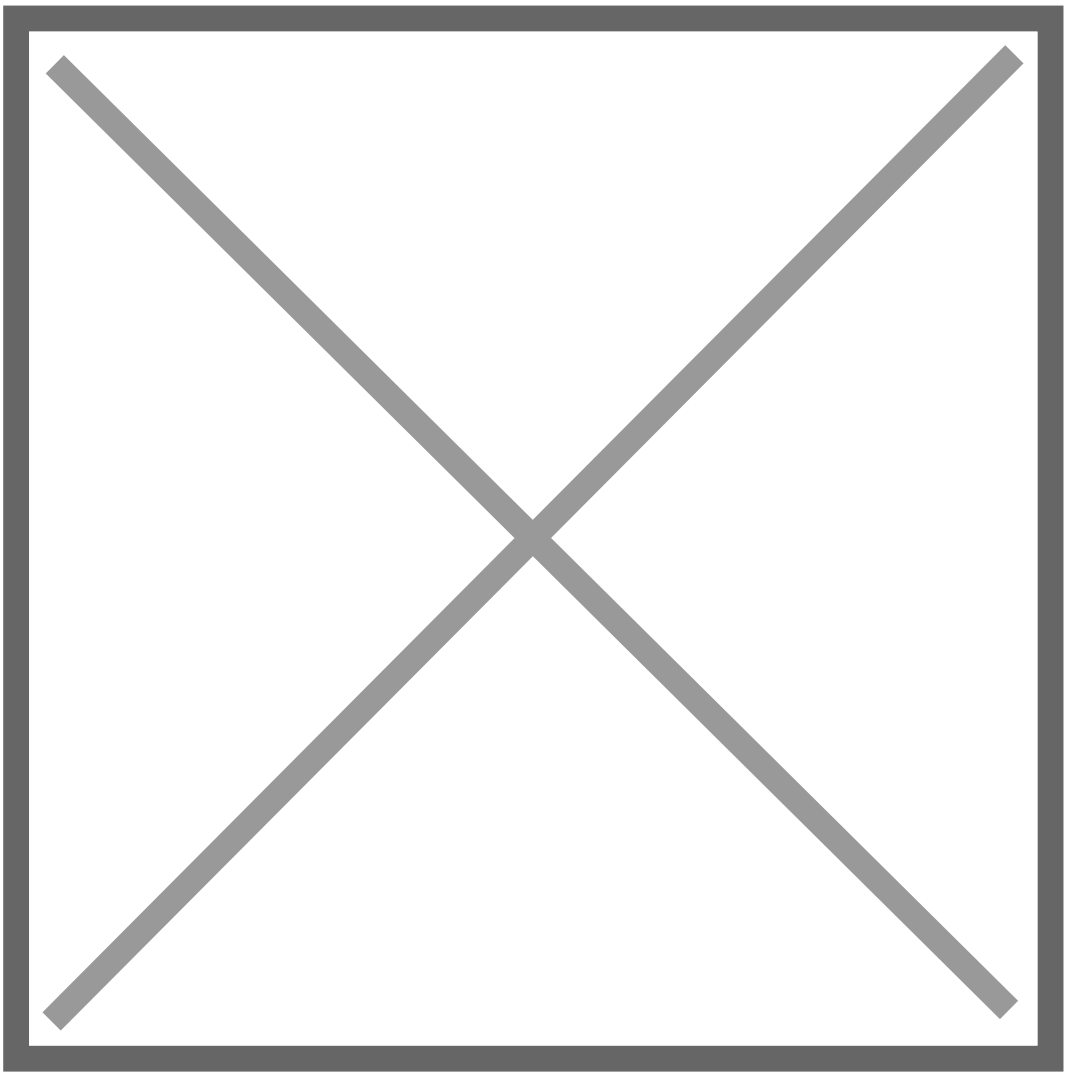
My array reduced it to 1.7 GB:



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



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



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

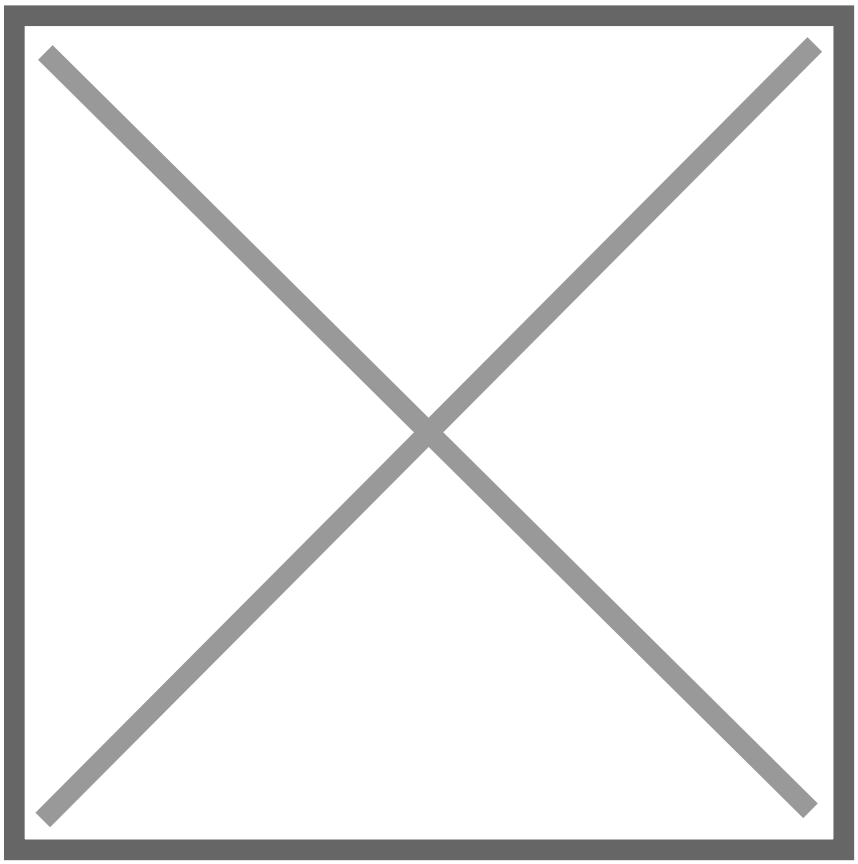
Great! So now, back to the original question, what about VMFS-6?

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.

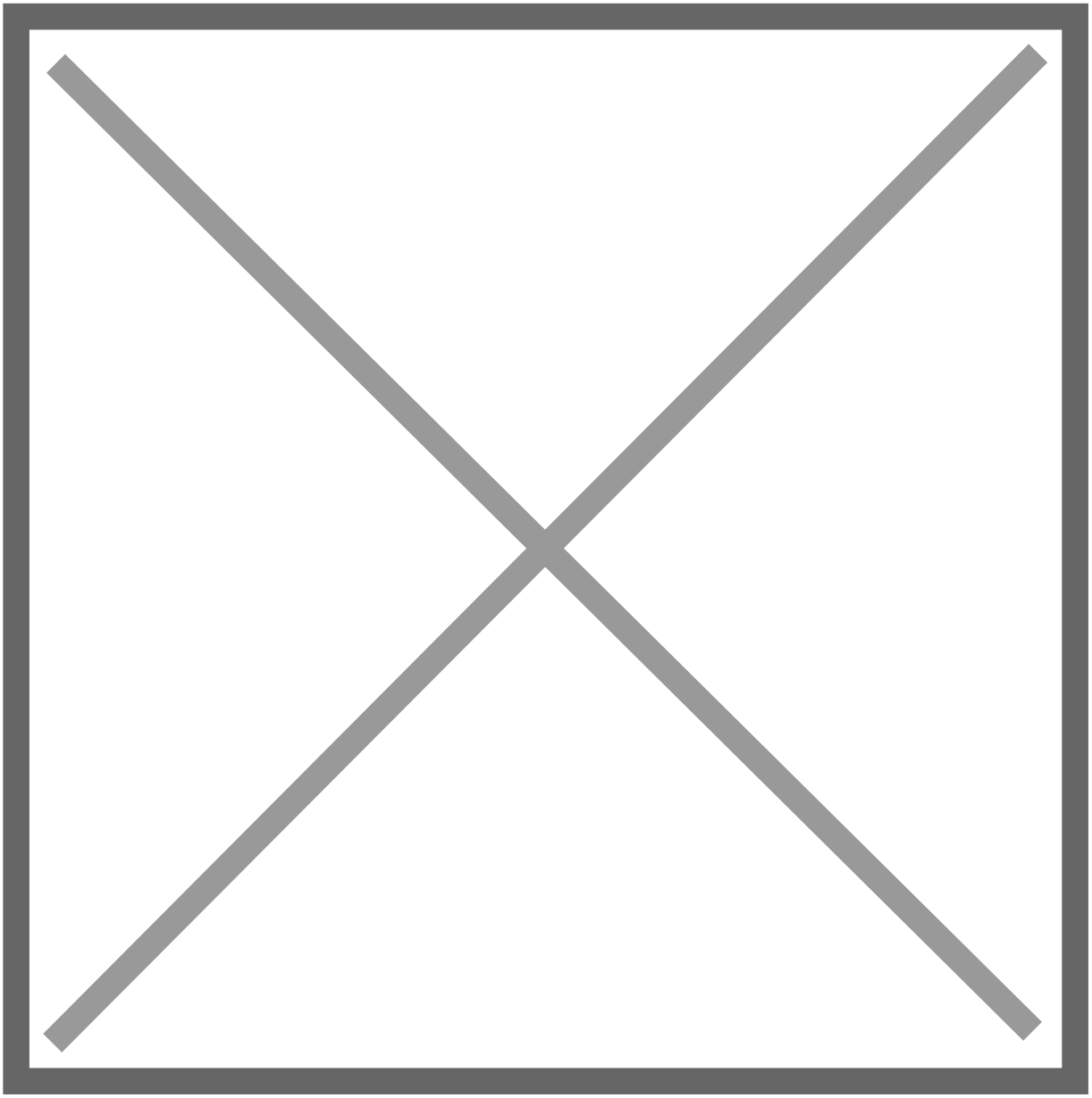
I moved my VMDK to my VMFS-6 datastore:

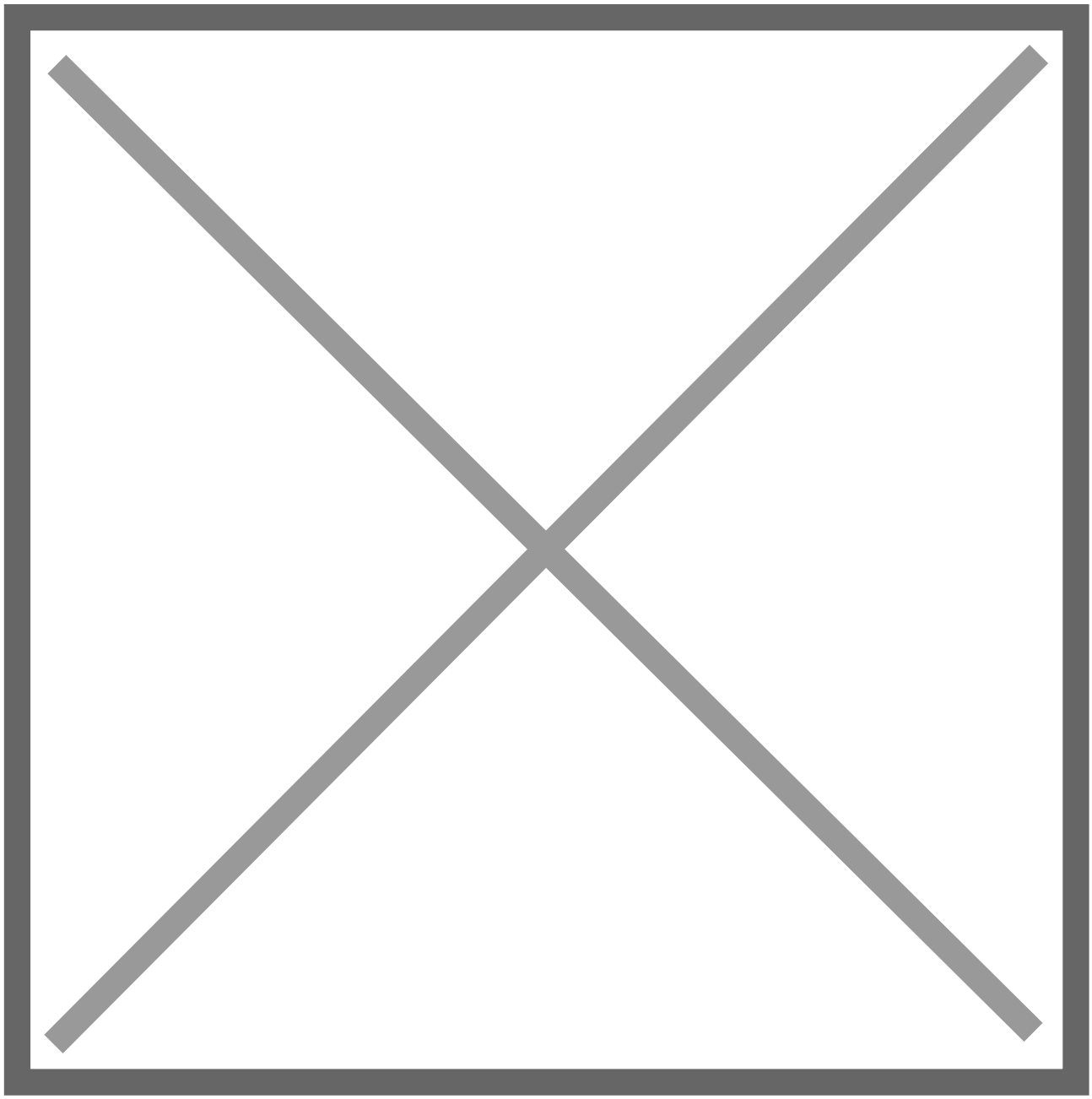


I will not go through every step again, let’s just start from the “we just deleted the files” step, but we have yet to run fstrim. So we have dead space.

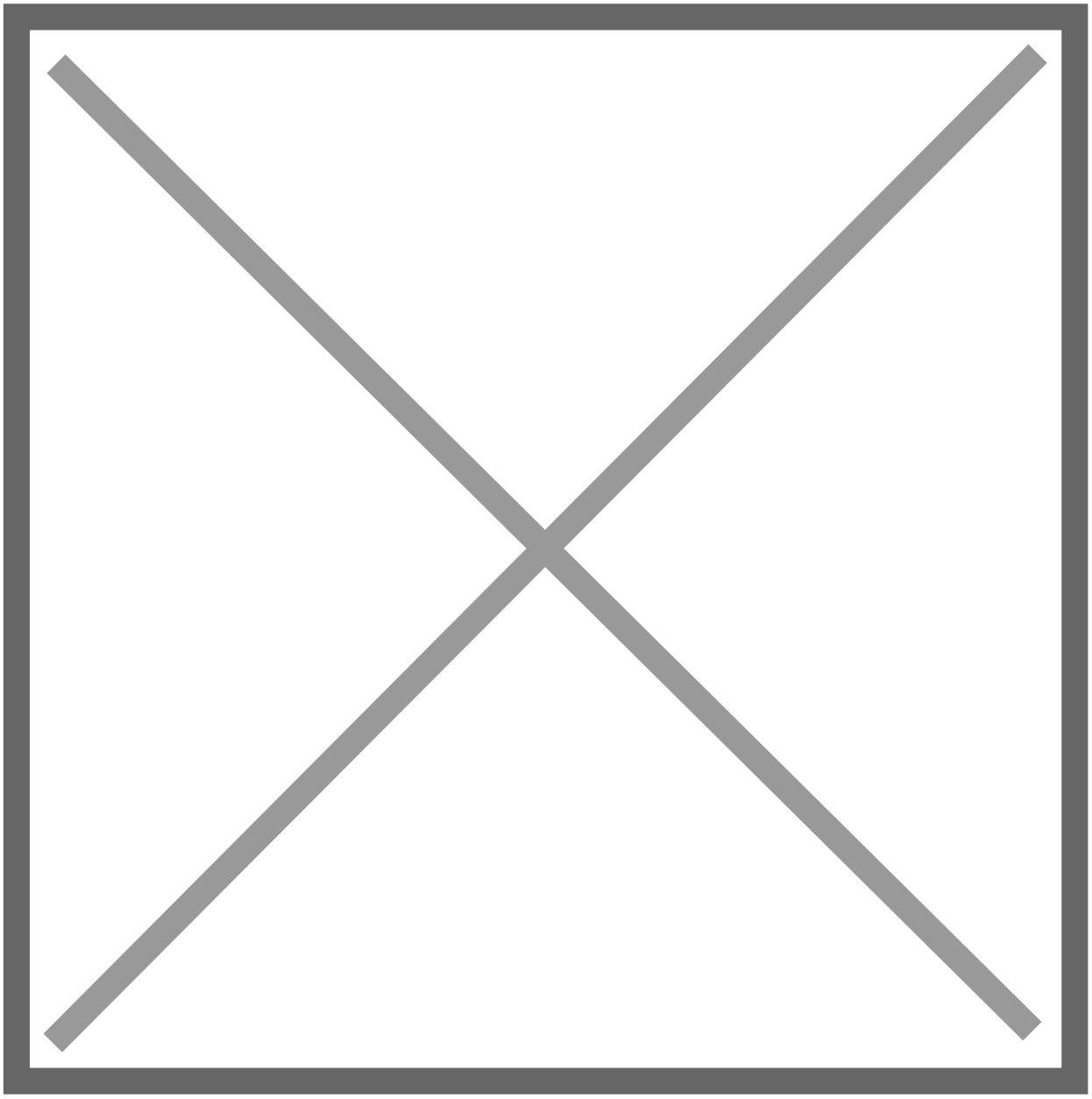
VMFS-6: EnableBlockDelete Disabled, Auto-UNMAP Enabled

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



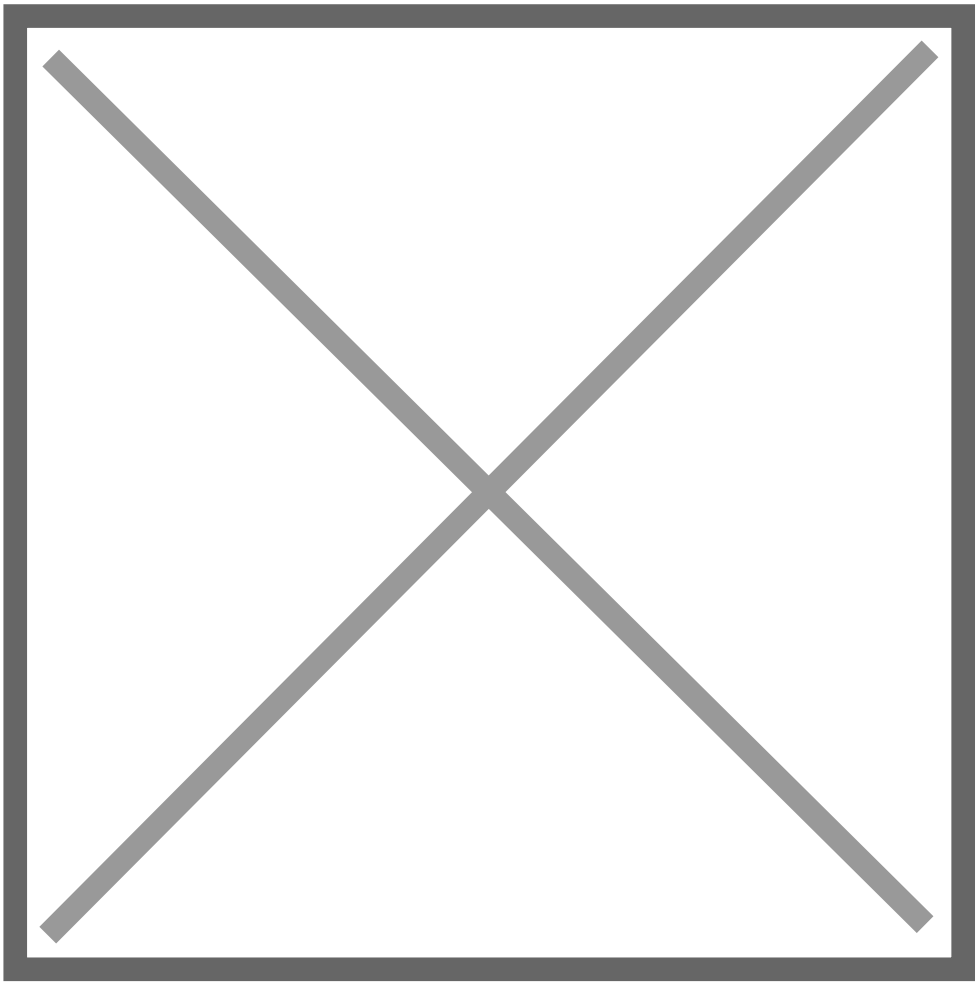


If I use vsish, I can see no automatic UNMAPs have been issued to this datastore from my host.

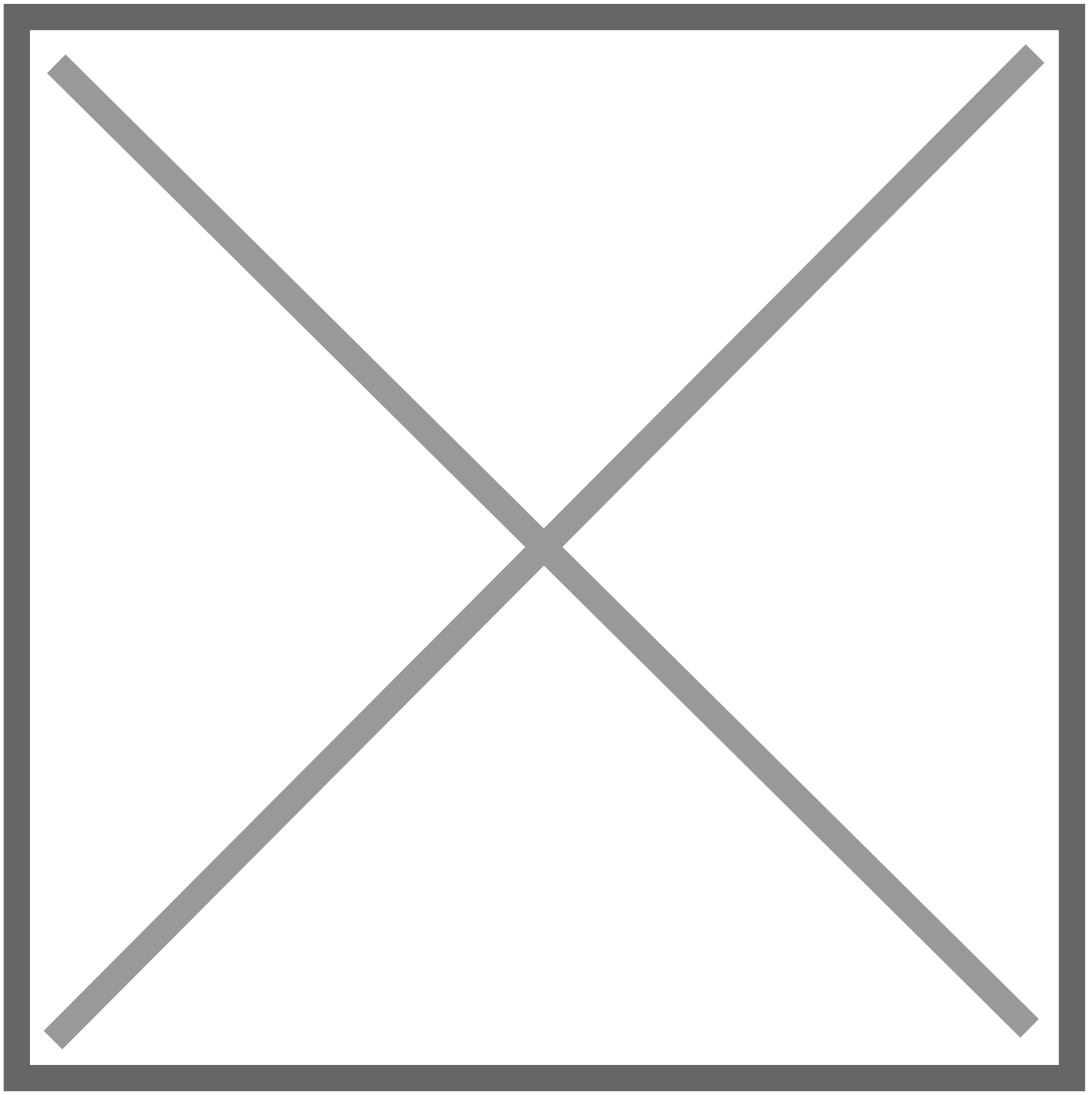


Note “UNMAP IOs” and “Unmapped blocks” are both zero.

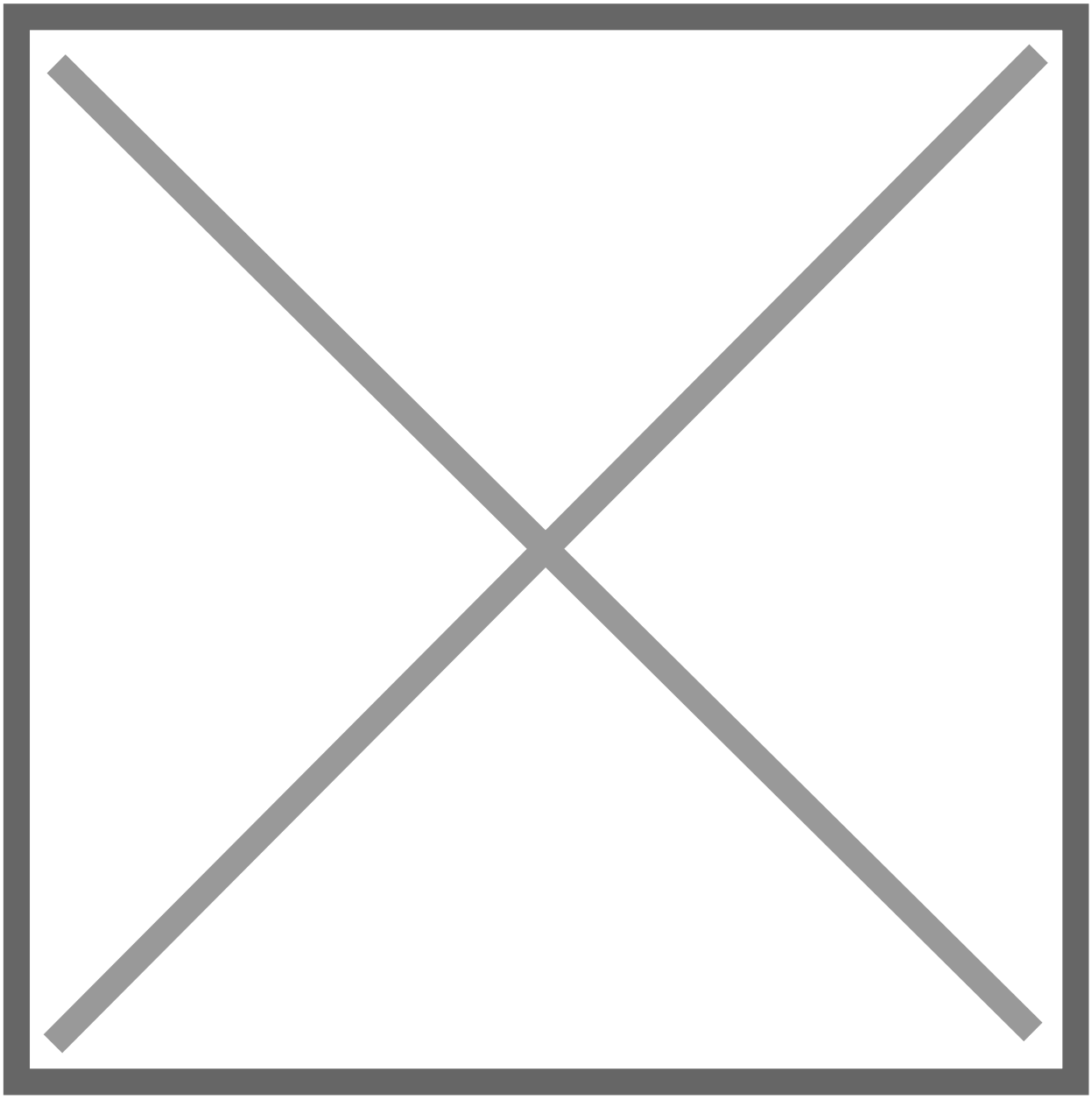
So I run fstrim. My VMDK is back down to 400 MB:



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



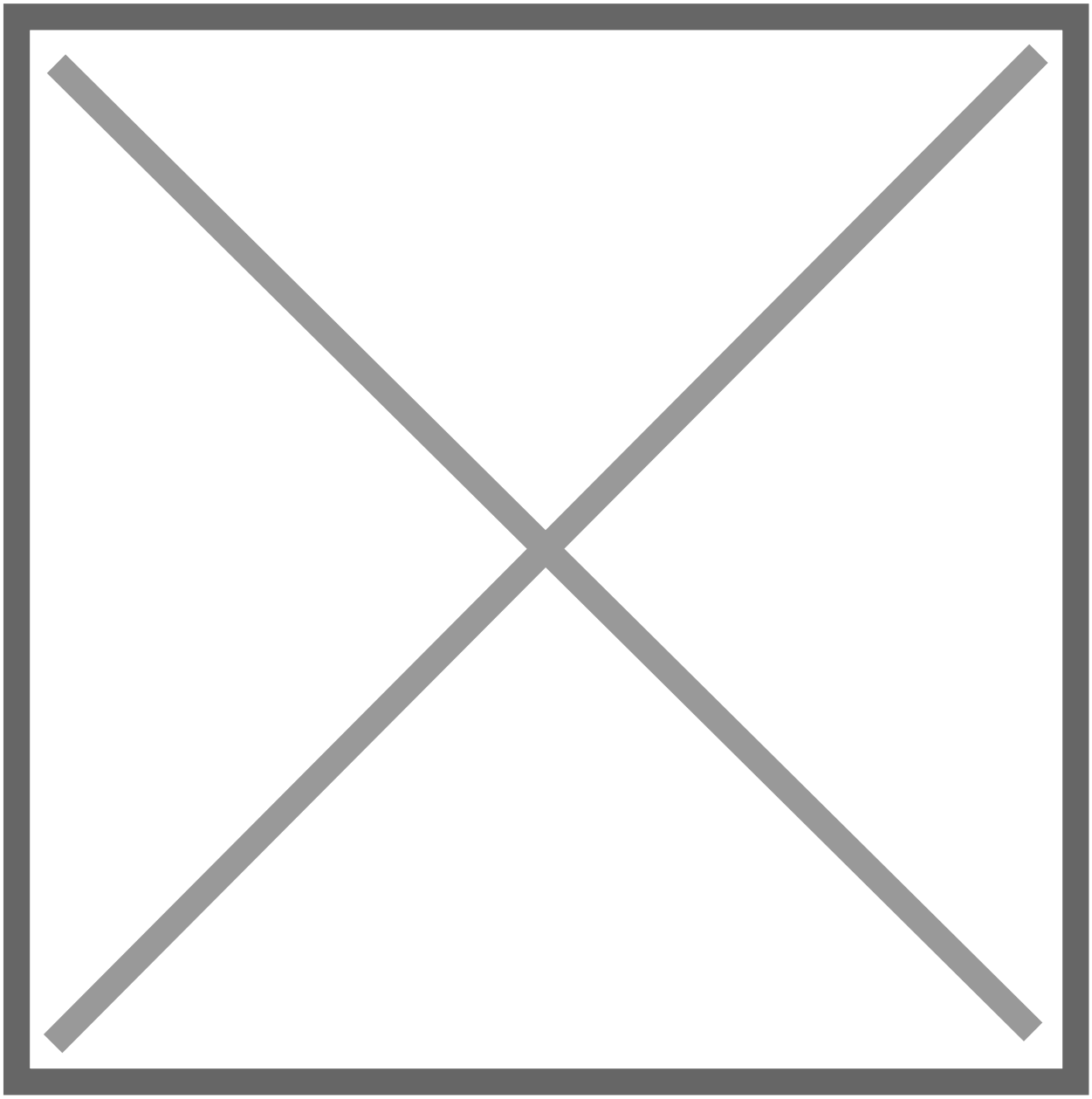
Since EnableBlockDelete was disabled and auto-unmap was enabled we can see this was auto-unmap. We can further show that by looking back at vsish:



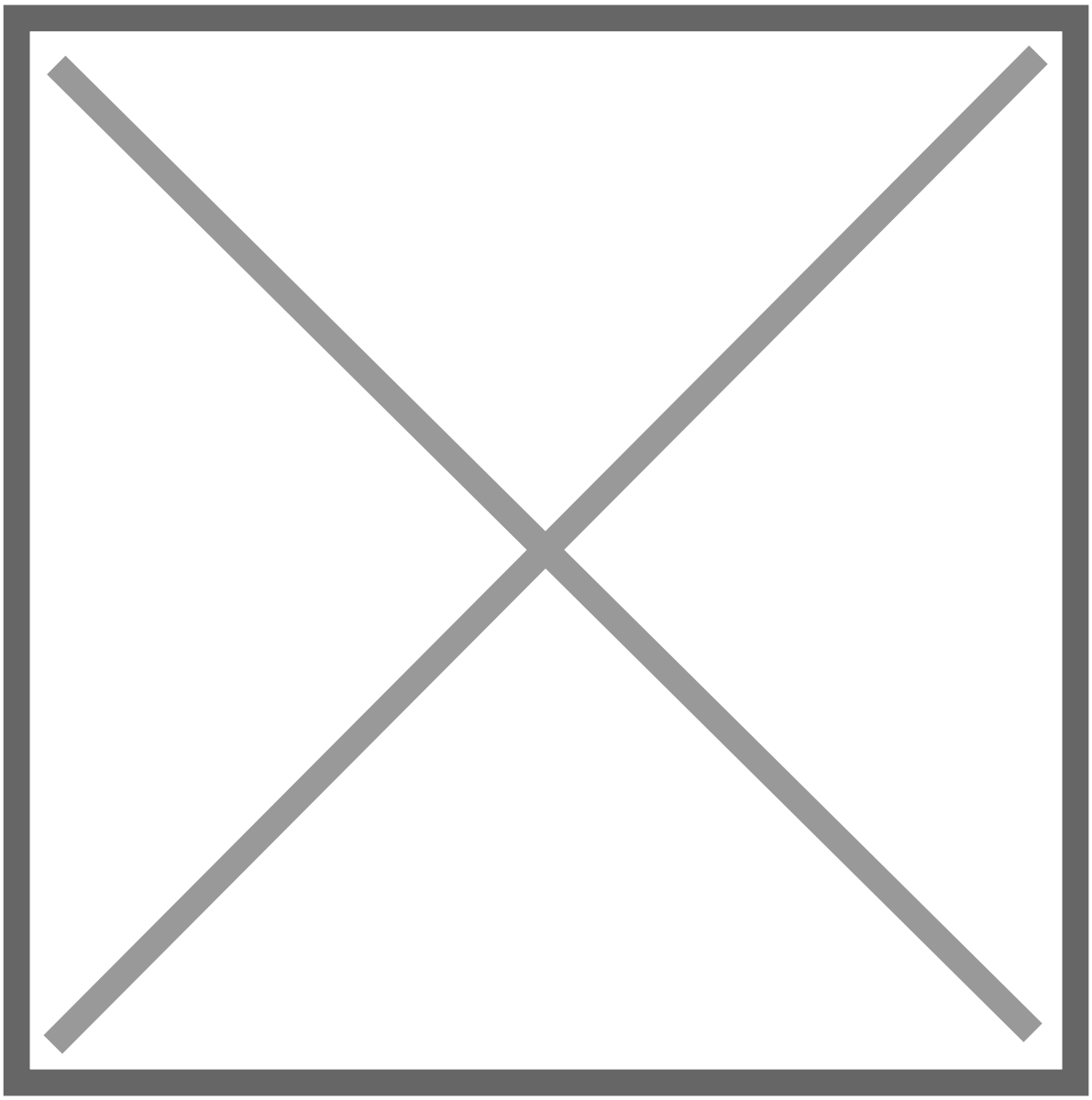
62 UNMAP I/Os and 3878 blocks reclaimed. So we don’t need to turn on EnableBlockDelete in the case of VMFS-6!

VMFS-6: EnableBlockDelete Enabled, Auto-UNMAP Disabled

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

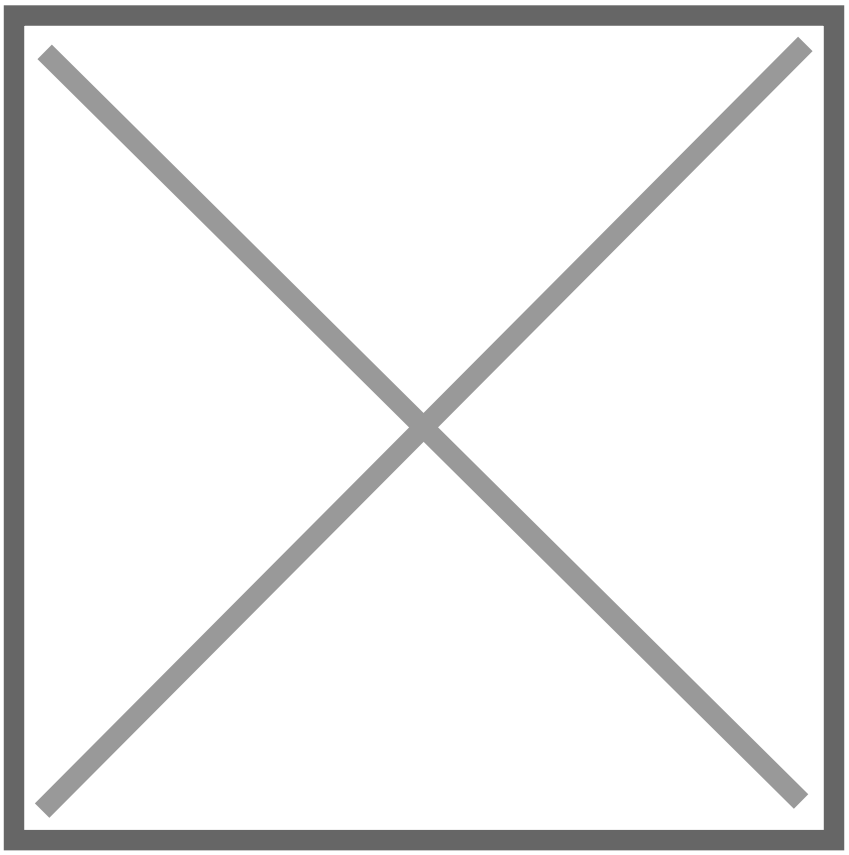


...and auto-UNMAP disabled on the datastore:

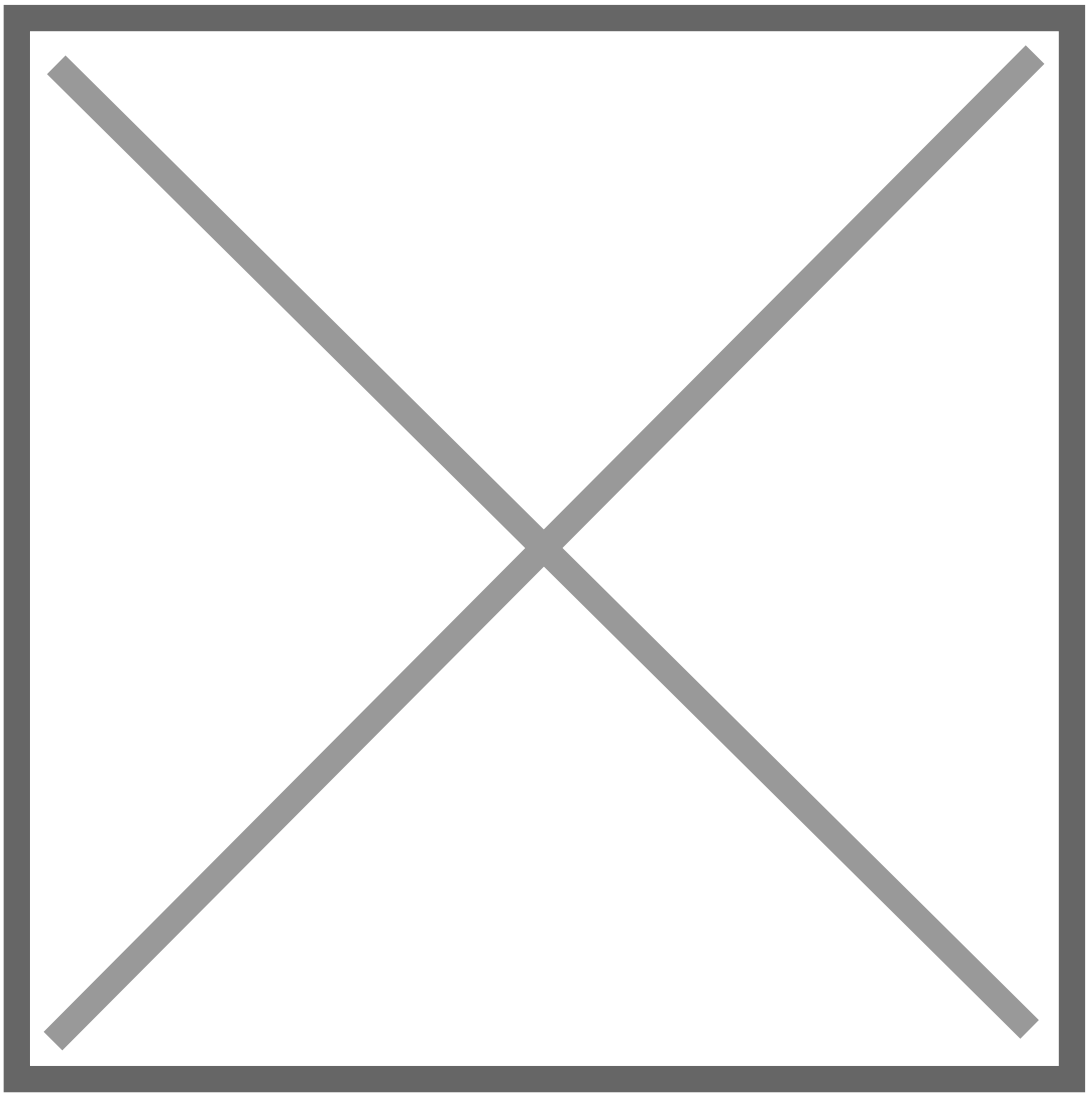


Let’s run through the process again. I refreshed my environment so counters are reset etc. Add the VMDK, put data on it, delete the data and run fstrim:

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.

Revision #2
Created 2023-05-28 11:08:09 UTC by Dino Edwards
Updated 2023-05-28 11:13:05 UTC by Dino Edwards