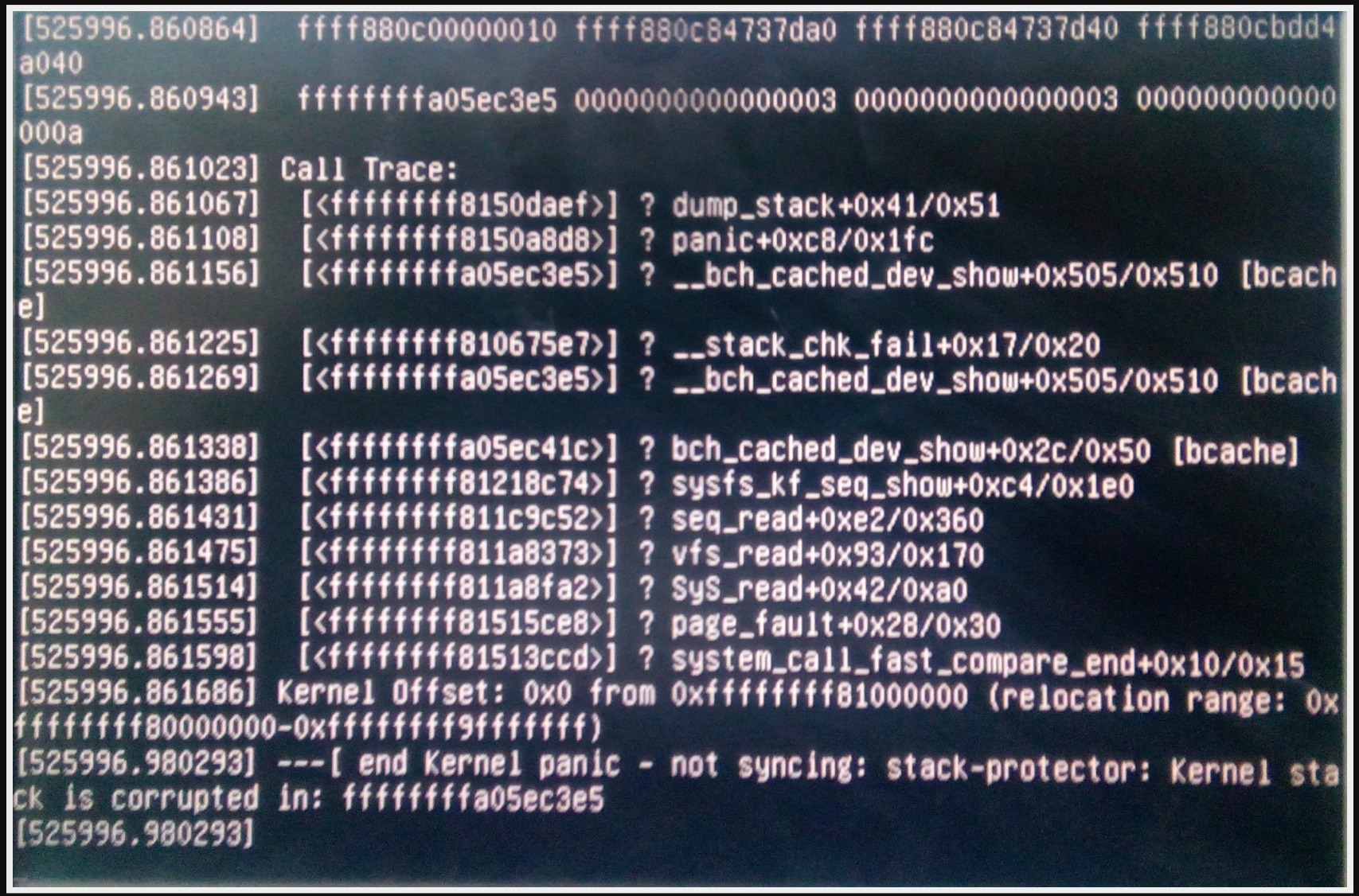
bcache for SSD+HDD

Or: bcache corrupted my data (twice) and here's why I'm still using it...

Danny Robson

A Warning


[525996.860864] ffff880c00000010 ffff880c84737da0 ffff880c84737d40 ffff880cbdd4a040
[525996.860943] ffffffffa05ec3e5 0000000000000003 0000000000000003 000000000000000a
[525996.861023] Call Trace:
[525996.861067] [<ffffffff8150daef>] ? dump_stack+0x41/0x51
[525996.861108] [<ffffffff8150a8d8>] ? panic+0xc8/0x1fc
[525996.861156] [<ffffffffa05ec3e5>] ? __bch_cached_dev_show+0x505/0x510 [bcache]
[525996.861225] [<ffffffff810675e7>] ? __stack_chk_fail+0x17/0x20
[525996.861269] [<ffffffffa05ec3e5>] ? __bch_cached_dev_show+0x505/0x510 [bcache]
[525996.861338] [<ffffffffa05ec41c>] ? bch_cached_dev_show+0x2c/0x50 [bcache]
[525996.861386] [<ffffffff81218c74>] ? sysfs_kf_seq_show+0xc4/0x1e0
[525996.861431] [<ffffffff811c9c52>] ? seq_read+0xe2/0x360
[525996.861475] [<ffffffff811a8373>] ? vfs_read+0x93/0x170
[525996.861514] [<ffffffff811a8fa2>] ? SyS_read+0x42/0xa0
[525996.861555] [<ffffffff81515ce8>] ? page_fault+0x28/0x30
[525996.861598] [<ffffffff81513ccd>] ? system_call_fast_compare_end+0x10/0x15
[525996.861686] Kernel Offset: 0x0 from 0xffffffff81000000 (relocation range: 0xffffffff80000000-0xffffffff9fffffff)
[525996.980293] ---[end Kernel panic - not syncing: stack-protector: Kernel stack is corrupted in: ffffffffa05ec3e5
[525996.980293]

Scenario

- I use a fair amount of storage.
- I need fast random access.
- I don’t want to spend a lot of money.
- I have a fair amount of time on my hands…

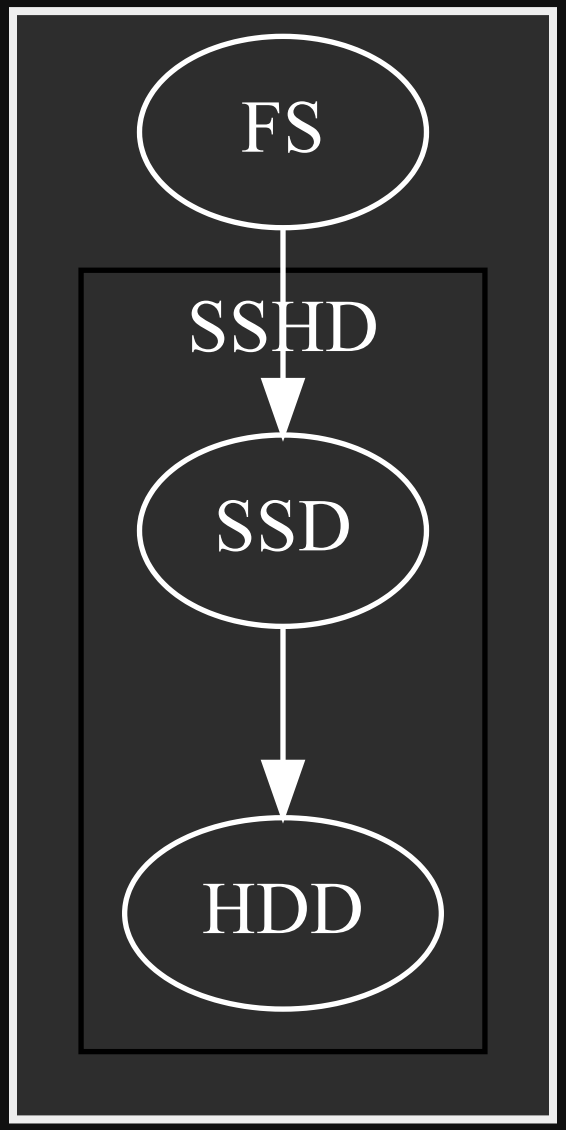
Current storage

	Count	Size	Total	Updates
photos	70k	50MB	1000G	static
dev	5k	4k-100M	100G	dynamic
conf	many	small	20G	dynamic

Possible hardware

	size	contig	rand	cost
HDD	***	**		$
SSD	*	***	***	$$$
SSHD	**	**	*	$$

Combining hardware

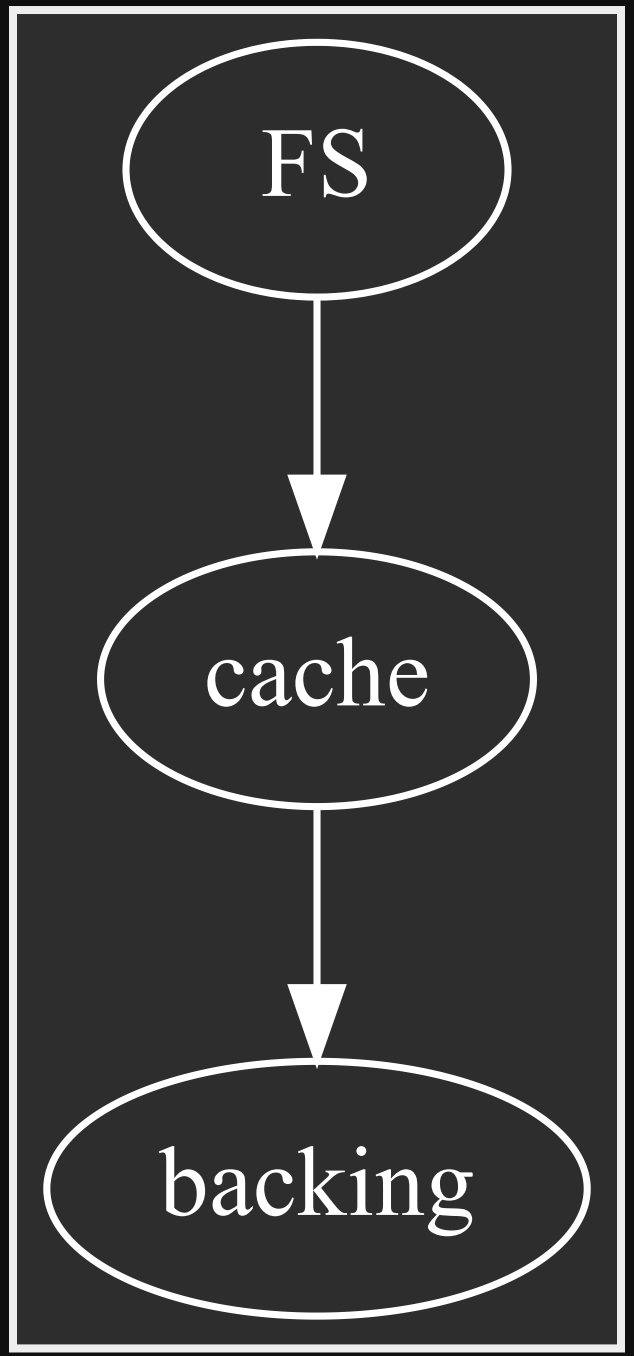


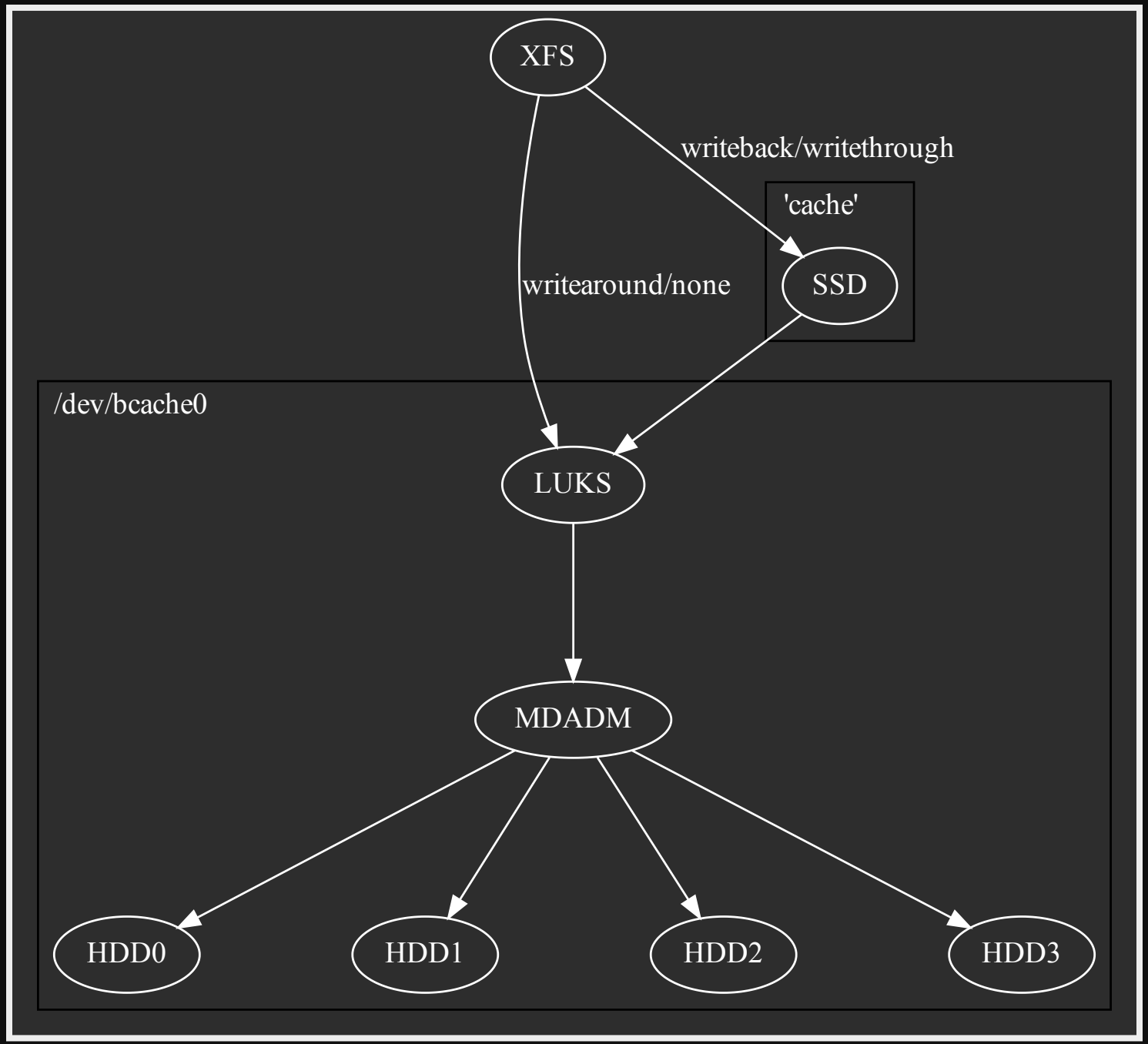
Software caching

- bcache
- flashcache, enhanceio
- Intel’s "Optane"

bcache

- Mainline kernel driver
- Tunable
- "Just another block device"


FS
cache
backing


XFS
writeback/writethrough
'cache'
writearound/none
SSD
/dev/bcache0
LUKS
MDADM
HDD0
HDD1
HDD2
HDD3

Demo

Conventions

`HDD=/dev/sda1` `SSD=/dev/nvme0n1p1`

Create

```
make-bcache -B $HDD
make-bcache -C --block 4k --bucket 2M $SSD
```

```
make-bcache -B $HDD -C --block 4k --bucket 2M $SSD
```

Register

```
echo $HDD > /sys/fs/bcache/register
echo $SSD > /sys/fs/bcache/register
```

But, udev if you can.

Attach

```
CSET_UUID=$(bcache-super-show $HDD | grep cset.uuid | tr '\t' '\n
echo $CSET_UUID > /sys/block/bcache0/bcache/attach
echo "writeback" /sys/block/bcache0/bcache/cache_mode
```

Concerns

- Multiple data corruption bugs over the years
- Middling distro support
- Extra complexity in initramfs
- High write pressure on the SSD. Single point of failure.

Future expansion

- `bcachefs`: upstreaming 'real soon'