MY WEIRD PROMPTS

Podcast Transcript

EPISODE #206

Beyond the Drive: Mastering Btrfs, ZFS, and Snapshots

Published January 08, 2026 • Runtime: 21:42

https://myweirdprompts.com/episode/btrfs-zfs-storage-pooling/

EPISODE SYNOPSIS

In this episode of My Weird Prompts, Herman and Corn dive deep into the world of advanced file systems like Btrfs, ZFS, and XFS, sparked by a housemate's complex five-disk workstation setup. They demystify the "magic" of Copy-on-Write (CoW) technology, explaining how snapshots provide a near-instant "undo button" for your entire OS without eating up your storage space. Whether you're a data hoarder looking for ultimate integrity or a performance junkie chasing raw speed, this guide breaks down which architecture fits your digital life and why a snapshot is never a replacement for a true backup.

DANIEL'S PROMPT

Daniel

I'd like to discuss file systems that span multiple disks, specifically Btrfs, ZFS, and XFS. Can you explain how these systems work, particularly how they handle features like snapshoting so space-efficiently? Also, what are the primary differences between Btrfs, ZFS, and XFS in simple terms?

TRANSCRIPT

Corn

Hey everyone, welcome back to My Weird Prompts. I am Corn, and I am sitting here in our living room in Jerusalem, looking at a very impressive, and slightly intimidating, tower of computer hardware that belongs to our housemate, Daniel.

Herman

Herman Poppleberry at your service. And yeah, Daniel has really gone down the rabbit hole with this one. He sent us a prompt about file systems that span multiple disks, specifically B-T-R-F-S, Z-F-S, and X-F-S. He is actually running a five-disk array on his workstation right now using B-T-R-F-S, which is pretty bold for a home setup, but also incredibly cool.

Corn

It is fascinating because most people just think of a hard drive as a box where you put files, and the file system is just the way those files are organized. But when you start talking about spanning multiple disks and things like space-efficient snapshotting, you are really talking about the fundamental architecture of how data exists on physical matter. It is much more complex than just a digital filing cabinet.

Herman

It really is. And Daniel's experience with using an artificial intelligence agent to help him install Ubuntu with B-T-R-F-S on a five-disk array is a great example of what is possible in twenty twenty-six on a home workstation. These technologies used to be reserved for high-end enterprise servers in climate-controlled basements. Now, you can set it up on your workstation with a bit of help to navigate the subvolume layouts.

Corn

So, let us dive into the meat of this. When we talk about a file system spanning multiple disks, we are essentially moving away from the traditional model where one disk equals one partition equals one file system. We are talking about pooling resources. Herman, how do these systems actually manage to treat five different physical objects as one cohesive space?

Herman

That is the magic of storage pooling. Traditionally, you had the file system on top and a separate layer called R-A-I-D underneath it. The file system was blind to the disks; it just saw one big virtual drive. But with Z-F-S and B-T-R-F-S, the file system and the volume manager are the same entity. They are "disk-aware."

Corn

And that awareness is key, right? Because if the file system knows where every single block of data is on every single physical disk, it can do things that a traditional system just cannot.

Herman

Exactly. It can optimize for speed, ensure data integrity by comparing copies on different disks, and handle those snapshots Daniel was so impressed by.

Corn

Let us talk about those snapshots because that seems to be the feature that really blew Daniel’s mind. He mentioned that the space involved in creating these recovery points is incredibly modest. To a casual user, that sounds like magic. If I have a one-terabyte drive and I take a snapshot, shouldn't that take up another one terabyte of space? How does it stay so small?

Herman

This is where we get into Copy-on-Write, or C-O-W. Most traditional file systems use an "in-place-update" system. If you change a word in a document, the computer finds the exact spot on the disk, erases it, and writes the new word over it.

Corn

Right, it is like using an eraser on a piece of paper and writing in the same spot.

Herman

Exactly. But B-T-R-F-S and Z-F-S never overwrite data. When you change that word, the file system writes the entire new version of that block to a completely new, empty spot on the disk. Then, it just updates its map—its pointers—to say, "Hey, the current version of this file is now at the new address."

Corn

Okay, so the old data is still there, just sitting on the disk, but the file system is essentially ignoring it for the current version of the file.

Herman

Precisely. And that is why a snapshot is so efficient. When you take a snapshot, the file system basically just freezes that map of pointers. Since nothing is ever overwritten, that old data stays exactly where it was. The snapshot doesn't copy the data; it just protects those old blocks from being recycled.

Corn

So the only extra space a snapshot takes up is the space for the new data you write after the snapshot was taken. If you take a snapshot and then never change a single file, that snapshot takes up almost zero additional space.

Herman

You nailed it. It is just metadata. This is what makes it possible to have hundreds of snapshots without filling up your drive, as long as your rate of change isn't massive.

Corn

That is a huge paradigm shift for data safety. It means you can let an AI agent run scripts on your machine, and if everything goes sideways, you just tell the file system to point back to the map from ten minutes ago. It is like a save point in a video game for your entire computer.

Herman

It really is. In the old days, if you broke your system, you were looking at a long night of re-installing. With B-T-R-F-S or Z-F-S, it is a near-instant rollback.

Corn

Now, Daniel mentioned three big names: B-T-R-F-S, Z-F-S, and X-F-S. Let us start with Z-F-S because you often hear it described as the gold standard. What makes Z-F-S so special?

Herman

Z-F-S stands for the Zettabyte File System. It was originally developed by Sun Microsystems to ensure data was never, ever corrupted. They introduced end-to-end checksumming.

Corn

Can you break down what checksumming actually does?

Herman

Think of it like a digital fingerprint for every piece of data. Every time Z-F-S writes a block, it calculates a mathematical signature and stores it. When you read that data back, Z-F-S recalculates the signature and compares it. If they don't match, Z-F-S knows the data has been corrupted—what we call "bit rot."

Corn

And if it is a multi-disk system, Z-F-S can actually fix it, right?

Herman

Exactly. It says, "Wait, the fingerprint is bad on disk one, but I have a redundant copy on disk two. Let me check that. Fingerprint matches? Great. I will give the user the good data and automatically repair the corrupted block on disk one."

Corn

That is incredible. But Z-F-S has a reputation for being a bit of a resource hog. People say you need a gigabyte of RAM for every terabyte of storage. Is that still true in twenty twenty-six?

Herman

That is a persistent myth! That rule-of-thumb was always mainly about a feature called deduplication. For general use without dedup enabled, Z-F-S has never strictly required a gigabyte of RAM per terabyte of storage, and most home users run it comfortably with more modest memory.

Corn

So dedup is the real culprit there.

Herman

Right. Deduplication can still be very memory-hungry, so it is usually reserved for very specific workloads. There is ongoing work on improving deduplication performance, but it still makes sense for most home users to leave it off unless they really need it.

Corn

You mentioned another hurdle used to be rigidity—you couldn't easily add one disk at a time to expand your pool.

Herman

Exactly.

Corn

"Used to be"? Did that change?

Herman

It did! R-A-I-D-Z expansion is now an official feature in Open-Z-F-S. You can add a single disk to an existing array and the system will redistribute the data, although it is still relatively new and, for very critical systems, people may want to test it carefully before relying on it in production.

Corn

Which brings us to B-T-R-F-S, or "Butter F-S." This is what Daniel is using. Why would someone choose this over the mighty Z-F-S?

Herman

Flexibility. B-T-R-F-S is part of the mainline Linux kernel, so you can use it on most modern distributions without extra patches. It allows you to do things that would make a Z-F-S administrator sweat. You can have a pool of disks with completely different sizes—a five-hundred-gigabyte S-S-D and a four-terabyte hard drive—and B-T-R-F-S can pool them together and, once you add a drive and run a balance, redistribute data across them.

Corn

And you can add or remove disks on the fly, right?

Herman

Yes! You plug in a new drive, tell B-T-R-F-S to add it, and run a "balance" command to reshuffle data. It also handles R-A-I-D differently. B-T-R-F-S R-A-I-D one doesn't mean every disk is a mirror; it just means there are always at least two copies of every piece of data somewhere in the pool. It is much more intelligent about using available space, though the exact capacity and redundancy depend on the RAID profile you choose.

Corn

So if Z-F-S is the enterprise-grade vault, B-T-R-F-S is more like a high-tech, modular shelving system. What about the third one: X-F-S? Is it even in the same category?

Herman

X-F-S is the "heavy-duty truck." It is not a Copy-on-Write file system by default, so it doesn't have native, integrated snapshots like the other two. Its claim to fame is raw performance with massive files and high-concurrency workloads. If you are a movie studio handling eight K video files, X-F-S is often the fastest choice.

Corn

But Daniel asked about snapshots. Can X-F-S do that?

Herman

Sort of. X-F-S added a feature called "reflinks." It allows you to make a copy of a file that shares the same data blocks until one is modified. You can use tools to create "pseudo-snapshots" this way, but it doesn't have the built-in self-healing or the integrated volume management of Z-F-S or B-T-R-F-S.

Corn

So to summarize: X-F-S for raw speed and massive files, Z-F-S for ultimate data integrity and professional-grade storage, and B-T-R-F-S for the enthusiast who wants flexibility.

Herman

Spot on. I would add that X-F-S has long been the default for Red Hat Enterprise Linux installations because it is rock-solid at scale. B-T-R-F-S is much more mature overall now in twenty twenty-six, but for a long time, people were scared of its R-A-I-D five and six implementations because of the "write hole," and current guidance still warns about using those modes.

Corn

Oh, right. Is that still a thing?

Herman

The advice is still to use R-A-I-D one or R-A-I-D ten on B-T-R-F-S for maximum stability. If you want R-A-I-D five or six parity, Z-F-S is still the safer bet because of how it handles its "R-A-I-D-Z" structure.

Corn

Daniel mentioned he has an N-V-M-e and an S-S-D in his pool. He was worried the slowest drive would dictate the speed. Is he right?

Herman

In a simple array, yes. But B-T-R-F-S is smart. You can tell it to store metadata on the fast N-V-M-e and bulk data on the slower S-S-D. He is not necessarily wasting that speed if he configures it correctly.

Corn

This really changes the way you think about hardware. Instead of buying one massive drive, you just keep adding whatever is on sale to the pile.

Herman

It is a very democratic way of handling storage. It is essentially what Z-F-S and B-T-R-F-S bring to your desktop: the intelligence of a data center in your living room.

Corn

I want to go back to the snapshots. Should we be snapshoting before every software update?

Herman

Absolutely! Many distributions like Fedora and open-S-U-S-E do this automatically now. If an update breaks your drivers, you just reboot, select the previous snapshot from the boot menu, and you are back in action. It is an "undo" button for your entire operating system.

Corn

This is where the distinction between a snapshot and a backup becomes really important. Daniel said he should probably call it a snapshot and not a backup. Why?

Herman

This is the golden rule: a snapshot is NOT a backup. A snapshot lives on the same physical disks. If your power supply fries all five of your hard drives, your snapshots die with them. A backup must be on a different device, preferably in a different building.

Corn

Right. A snapshot protects you against software errors. A backup protects you against hardware disasters.

Herman

Exactly. However, these systems make backups easier. They have a feature called "send and receive." Since the file system knows exactly which blocks changed, it can send just those specific blocks over the network. You can back up a multi-terabyte pool in seconds if only a few megabytes changed.

Corn

That is a game changer. So, for Daniel, with his five-disk array and his penchant for tinkering, B-T-R-F-S was a great choice.

Herman

I think so. But he should definitely run a "scrub" occasionally.

Corn

A scrub? What does that mean for a file system?

Herman

It tells the file system to check every single block of data, calculate its checksum, and verify it matches the fingerprint. It is a way of proactively catching bit rot. On a five-disk array, doing that about once a month is a common recommendation.

Corn

It is amazing how much work these systems are doing under the hood. We just see a folder, but underneath, there is this dance of pointers and checksums.

Herman

It is one of the great unsung triumphs of computer science. We can store massive amounts of info and actually trust it won't vanish.

Corn

So, for our listeners who want to try this, what is the first step?

Herman

You can start with just one drive! Both B-T-R-F-S and Z-F-S work on a single disk. You won't get self-healing, but you get checksumming and instant snapshots. Just make sure you have a real backup before you start playing with your partitions!

Corn

That is a very fair point. I think we have covered a lot of ground here. We have looked at the magic of Copy-on-Write, the enterprise power of Z-F-S, and why X-F-S is still the king of speed for big data.

Herman

It is a fascinating world. And as storage gets cheaper, these systems are only going to become more common.

Corn

I hope so. I would love an undo button on my laptop for those times I accidentally delete a whole folder of photos.

Herman

Well, if you use B-T-R-F-S, you already have one. You just need to learn how to press it.

Corn

On that note, I think it is time to wrap up. Daniel, thanks for the prompt. It is always fun to look at the gear in our own house and realize just how much science is packed into those blinking boxes.

Herman

Absolutely. And hey, if you are listening and you have your own weird prompts, we would love to hear from you. Find us at my-weird-prompts dot com.

Corn

And if you have been enjoying the show, please leave a quick review on your podcast app. It genuinely helps other people find us. We have been doing this for over two hundred episodes now, and it is your support that keeps us curious.

Herman

Yeah, a rating makes a huge difference. So thank you to everyone who has already left one.

Corn

We will be back next week with another prompt. Until then, keep questioning, keep tinkering, and maybe take a snapshot of your system just in case.

Herman

Good advice. This has been My Weird Prompts. I am Herman Poppleberry.

Corn

And I am Corn. We will talk to you soon.

Herman

Bye for now!

Corn

So, Herman, be honest. If you were building a rig today, would you go B-T-R-F-S or Z-F-S?

Herman

Oh, man. For my home server? I am a Z-F-S guy. I like that enterprise heritage. But for my laptop? B-T-R-F-S all the way. The flexibility is just too good.

Corn

Fair enough. I think I will stick to my simple setup for now, but I might just ask Daniel to show me those snapshots.

Herman

Just don't let him talk you into buying five more hard drives. Our electricity bill is high enough!

Corn

Good point. See you everyone!

Herman

Take care!

Corn

One last thing for the tech-savvy folks. If you are interested in the deeper mechanics, look up the difference between a Merkle tree and a standard B-tree. That is the real secret sauce behind how Z-F-S handles its data integrity at scale. It is a fascinating rabbit hole.

Herman

Oh, now you are really getting into the weeds. Merkle trees? That is some high-level stuff. But you are right, it is beautiful math.

Corn

It really is. Alright, truly signing off now. Thanks for listening to My Weird Prompts. You can find all our past episodes on Spotify and at our website, my-weird-prompts dot com.

Herman

See ya!