Hacking the Pachyderm

Scaling Servers and People

Hazel Weakly, Hachyderm

Preston Doster, Hachyderm

@hazelweakly@hachyderm.io
@esk@hachyderm.io
Art courtesy of AI magic
curl -X GET https://agenda

🌿 Hachyderm becoming one of the largest elephant watching sanctuaries

🌿 Challenges we faced in undertaking the Great Elephant Migration

🌿 What’s next
Your RCA is Not My RCA

But Your RCA is Also Useless

Root cause: 2 failing HDDs
Your RCA is Not My RCA

But Your RCA is Also Useless

Perfect hindsight solution:

1. Upload media to object storage from the start
2. Migrate a week earlier

✨ done ✨
Time Loop
Paradox of
Distributed
Systems

Any problem that is prevented before it occurs will spontaneously reincarnate elsewhere...

...but weirder and more fucked up
“Root Causes” are Individual Insights

Systems Resilience requires Systemic Adaptation
Make Change Easy
Making change easy to handle ≠ Making change easy to do
The Great Elephant Migration
The Old Habitat

- Experimental ZFS build
- Debug linux kernel for testing root exploits
- Mastodon serves media off of NFS
- Main host has: Mastod, Nginx, Kubernetes, Postgres, and Redis.
The Timeline

🌿 Nov. 19: Building v2 hachyderm with NixOS
🌿 Nov. 27: NFS is unusable, we attempt to fix
Feat: nixos is the future!

krisnova merged 21 commits into main from nixos-hosts on Nov 22, 2022

hazelweakly commented on Nov 20, 2022

Bye arch!
Hi! Sys Load expert here! This is not funny, computers only do this when they’re in extreme distress.
I accidentally
the largest and most comprehensive
collection of notes on scaling mastodon

Whoops
The Timeline

🌿 Nov. 19: Building v2 hachyderm with NixOS
🌿 Nov. 27: NFS is unusable, we attempt to fix
🌿 Nov. 28: Unable to process queues fast enough
🌿 Nov 28: Run Hachyderm in both EU and US
The Cycle of Toil

🌿 EU wakes up. Balances sidekiq queues manually
🌿 US wakes up. I tell everyone the optimal numbers via calculation
🌿 We’re both wrong
🌿 GOTO :weed-leaf:
teaf: revert nixos is the future! Arch Linux is the future!

hazelweakly commented on Nov 20, 2022

The `yay`'s will continue until stability improves
Breaking the Toil Cycle

One of the highest leverage things done was Nova introducing these two things at the same time:

🌿 All changes needed to be logged via PR
🌿 Scripts which explicitly listed the services to modify
Breaking the Toil Cycle

"I thought you said to make change easy"

- Redistributing friction in key areas balanced the human workflows and streamlined communication.

- Humans need backpressure and circuit breakers too.
The Timeline

Nov. 28: Unable to process queues fast enough
Nov 28: Run Hachyderm in both EU and US
Nov 29: Start moving to object storage + tmpfs
Nov 30: prepare to move the DB to the EU
How 2 make postgres h8 u

“stuff is slow, add more workers” (this adds connections to postgres and slows it down) postgres: *doing more IO* realize queries take too long you run out of connections GOTO PAIN

High number of connections 2354
NFS: Not Fucking Scaleable

The rails app is effectively doing this:

1. Start db transaction
2. Upload image to NFS
3. INSERT or UPDATE statement
4. Commit the transaction
How many IOPS does it take to...
300 IOPS
12 fsync/s
"NFS is too reliable and performant for me, how do I fix that?"

Answer:
run it over the Atlantic ocean and store all the files on a system with two failing SSDs
Why didn’t we notice this earlier?

we_had_no_idea_what_we_were_doing.jpg.mp3

 Metrics are data
 Data is only as good as your understanding
 No(thing) was ever obviously wrong
Why did we take the approach we did?

🌿 We wanted to not be beholden to corporate ulterior motives (see: Twitter)
🌿 We wanted to build a service that was meaningfully owned by the community it served.
We should have burned out
Why didn’t we burnout?

- Intrinsic vs extrinsic motivation
- Aligned incentives throughout the process
- Psychological safety
- We chose our own solution and its downsides
- We continually supported each other
You’ll never be prepared for exponential scale

... But you can be prepared to support each other through the challenges you’ll face
Operating a mastodon instance is great

It's all the reliability and ease of maintenance of a rails application, combined with the user experience of hosting a mail server, and the developer workflow of day drinking on the Linux kernel mailing list.
Soo... what’s next?
Today

🌿 Production service, ~20k MAU

🌿 **Stateful**: people care about their toots!

🌿 **Real-time**: people care about their toots!

🌿 Key constraint: **volunteer** time is limited
Infrastructure values

Brian Cantrill’s Platform as a Reflection of Values - watch it!

Shape our technology choices

Inform how we prioritize our time
What do we **value most** as the Hachyderm Infra team?
Approachability

🌿 Low barrier to entry
🌿 Open to all backgrounds & levels
🌿 Let volunteers use their limited time **effectively**
Safety

🌿 Encourage experimentation & growth

🌿 It’s OK to break prod, so let’s learn from it when we do!

🌿 We’re in it together
Transparency

🌿 No hidden agendas
🌿 Docs, docs, docs
🌿 Clear team backlog
🌿 Public incident reviews - share always!
Key Choices
NixOS

🌿 Base OS & operating environment

🌿 Self-described: Reproducible, Declarative, and Reliable

🌿 Safe and Approachable*

```nix
7 system.stateVersion = "23.05"; # Do not change lightly!
8
9 services.postgresql.enable = true;
```

```nix
7 system.stateVersion = "23.05"; # Do not change lightly!
8
9 services.postgresql.enable = true;
```
Multi-cloud

🌿 From Day 1

🌿 Hetzner, Linode, DigitalOcean, Nova’s Basement

🌿 Paradox: many clouds yield simplicity
lots of stuff we don’t need vs. the stuff we do
Terraform Cloud

🌿 Declarative, Reproducible, and **Reliable**

🌿 **Lowers barrier** to manage our infra

🌿 **Predictable** workflow - PR, plan, apply
PR-driven workflow (familiar, comfortable, approachable)

Know what will happen before merge (safety, confidence)
Write it Down

Something is better than nothing
PR comments, good issue descriptions, Markdown docs in repo

**Anything** the next human can use
Our cloud provider, Hetzner, was rejecting packets

We had an outstanding, unpaid invoice

The auto-payment didn’t post

Change to credit card based billing

We didn’t know to manually process it

We didn’t see the alert emails

We didn’t read the small text

No one was monitoring that e-mail

We didn’t socialize the billing setup

We’re a scrappy startup, and we’re moving fast!

Shared e-mail for critical alerts

Create billing documentation
Q + A