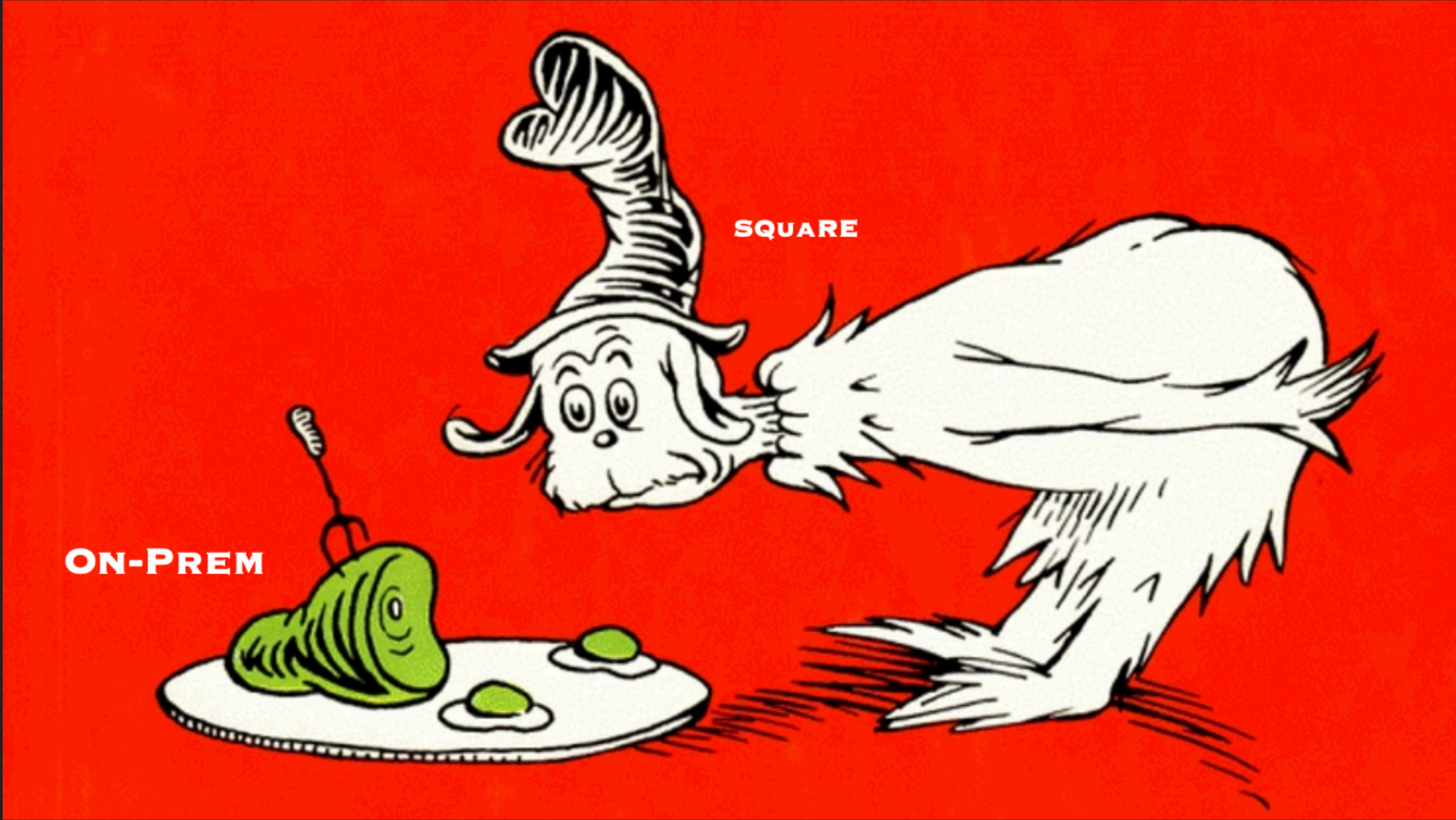
On-prem RSPs: Separation of Concerns

Frossie for SQuaRE

SQuaRE is not sized for on-prem

- 6 developer FTEs + institutional contracts for 2 additional FTEs for:
    - All RSP (inc VO) services
    - Developer platform (phalanx ecosystem)
    - Sasquatch / EFD
    - Times SQuaRE for ops reporting
    - Documentation (technotes, lsst.io, doc sites)
    - Developer aids (bots, GitLFS etc)
    - Operating data.lsst.cloud on Google Cloud for up to 10K users
- On-premises deployments came about often as “temporary” solutions while other software/environments were made available
- … well
- Phalanx has taken over as the defacto services deployment platform
- Now-Sasquatch was meant to be a low-stakes alternative to the “actual” EFD
- => None of this was sized
- ==> No cycles for operating on-premises deployments


SQUARE
ON-PREM

Well how's that working for you Frossie?

- 18/22 phalanx environments are on-prem
- Understandable as phalanx provides service deployment abstraction, developer velocity aids and many services missions are interested in
- But…

- 🟢 UKDF
- 🟢 FRDF
- 🟢 NASA SphereX (internal)
- 🟢 Chandra (experimental)
- 🟡 USDF
- 🔴 Summit

If GCP does it for me, you have to do it for me

- (*) some exclusions apply, but correct to the 0th order

VERA C. RUBIN
OBSERVATORY

To 0th order, on-prem = GCP w/ different economics

- This is not arbitrary
- Commercial cloud capabilities found the right line between generic infrastructure and customer-specific load
- We rely on the capabilities and velocity found in commercial clouds
- Where things cannot be equivalent (eg developer self-serve and scaling cannot be matched obviously) the on-prem provider has to fold in dealing with that in their operational model
- Assuming everything is (as close as possible to) GCP-like creates the level of consistency we need to keep the team developing

K8s

- K8s has been a huge step in abstracting infrastructure: in fact the success of phalanx would not have been possible without it and this talk would be moot

- Things that cause k8s to behave not like we expect (eg USDF vcluster “surprises” such as PV reclaim policy, taints differences etc) cause issues that are hard to detect and understand

Storage

- S3 compliant
- Signed URLs (GCS-like)
- POSIX
- Node-local storage for some services (eg Sasquatch)
- K8s persistant volumes

- Generally ok though performance is still an open question and work on storage has a huge blast radius

Database

- Postgres-as-a-service
  - Postgres server in phalanx is only there for development.
- Indexing, performance tuning, monitoring, backups and other DBA “table-wrangling” tasks are under DF remit
- Only Summit offers us this so far
- We maintain our own schemas for services so we don’t require help indexing those, but the data product holdings probably do
- Qserv teams operates USDF Qserv as part of its development and because it is the hybrid production deployment

Connectivity & Ingress

- Heavily rely on our ability to manipulate k8s ingress resource and nginx ingress configuration
- We assume "normal" internet connectivity (IN2P3 operates without inbound and summit works behind VPN so clearly works that way now but)
- We run cert-manager in-cluster (environment-specific configuration)
- USDF "super-ingress" is annoyingly different from GCP model
- Summit VPN is annoying and we will not have time to deal with that for some time

“Cyber”

- We need well-articulated concerns in good time
- For internal on-prem usage, be aware that we don’t necessarily jump to fix every concern
- Make sure you can burn down and re-deploy
- RSP is remote code execution as a service by design. There is a limit to what we can do in this area
- We rely on our on-prem teams and our management chains to negotiate with Cyber and mostly that works
- Assumption of risk and escalation authority is still wonky with on-premises deployments

Scaling

- Since scaling is harder on-prem, DFs have to consider their own sizing and tune accordingly

- We make sure services *can* horizontally scale given adequate resources

Availability / Reliability

- Hybrid exposed services *have* to be reliable - users don't know/care where the RSP starts and Butler begins, but I also don't know/care where USDF starts and S3DF begins
- Communication about outages (potential and actual)
- Time for recovery
- The invisible 10K users and hybrid make *me* the biggest user of all infrastructure supporting the USDF-side of hybrid
- Tracking the status/progress of work we rely on at USDF can be a challenge

User management

- User wrangling is done by the DF:
  - Authentication source
  - UIDs and GIDs
  - Groups
- This can be an issue (esp when it comes to bot user account creation)
- We need a better off-boarding story everywhere

Alerts & Monitoring

- Tier 1 response needs to be on the on-prem side because since data.lsst.cloud is a lot more aggressively used so most on-prem issues are infrastructure related
- We have good service alerts and improve all the time
- Mobu alerts are a valuable heads up, use them
- Summit is a giant issue here

Backups

- Even if we do application-level backups, the filesystem that the backups are on need to be backed up
- Consider Velero
- Disaster recovery is on-prem’s business
- Stuff “just comes back up” with the exception of Sasquatch

The exceptions

- Services where the primary production environment is on-prem (instead of cloud) do get significant attention from us and we are more hands-on with their deployment and operations

- EFD (Summit and USDF)
- Kafka/Sasquatch at Summit
- Times Square (USDF)

Resources

- Bootcamp videos
- Phalanx topic in c.l.o, esp if nothing is burning
- Feel free to prod on slack
- Comm option include Adam/George/Fabio (team leads meeting
- We have been over-relying on zoom to talk about changes, we plan to get better about written release notes and comms in Ops