goto; amsterdam

Please

Ask questions through the app

Rate Session

Thank you!

Introduction to SRE at Google

Christof Leng, cleng@google.com
June 2018

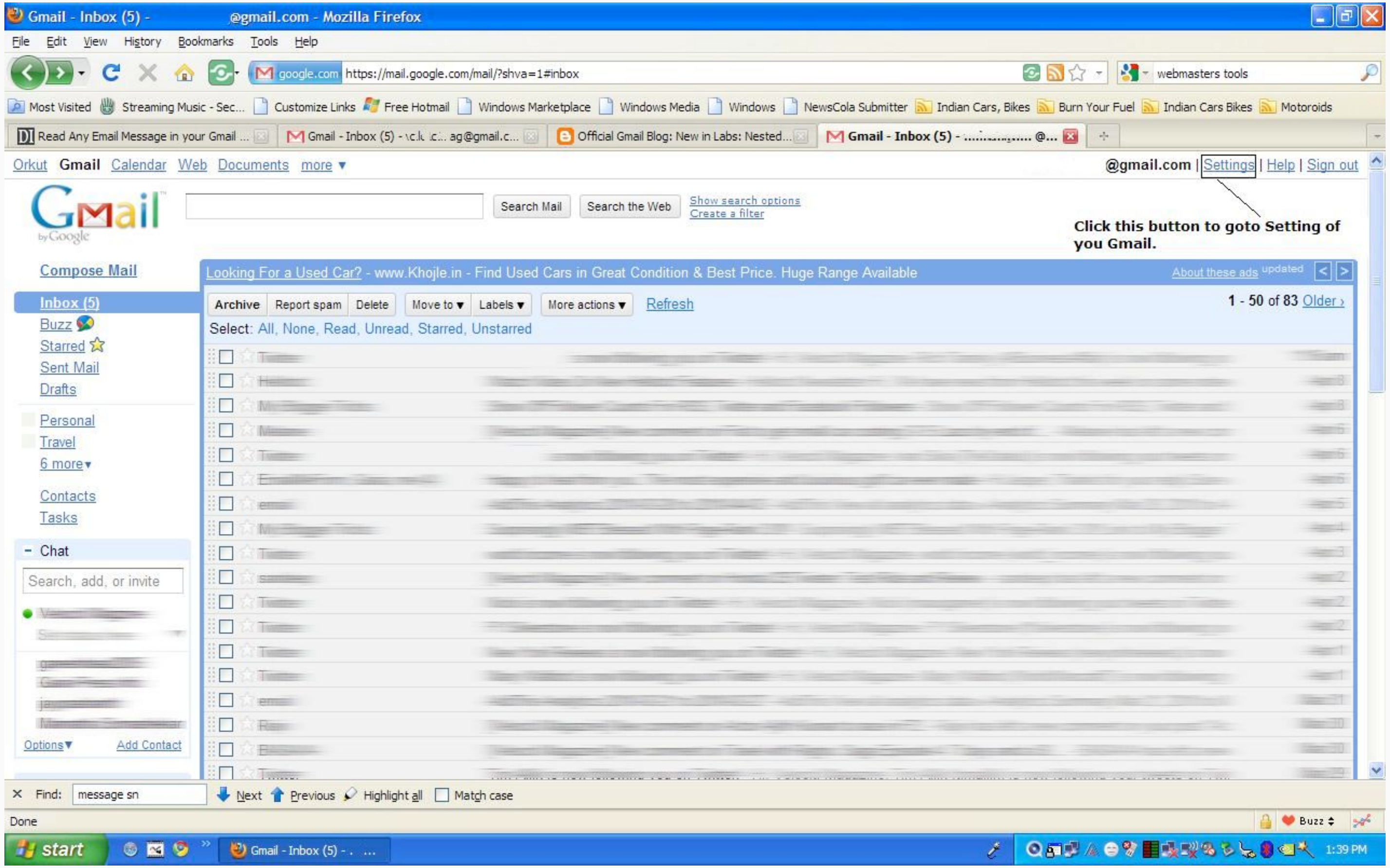
Speaker Introduction

- Christof Leng
- Site Reliability Manager at Google Munich
- Developer Infrastructure SRE
  - Responsible for Google's developer and CI/CD tools
- Researcher, politician, DJ

Why Reliability?

- It's the number one feature

Do you prefer Gmail 2010?



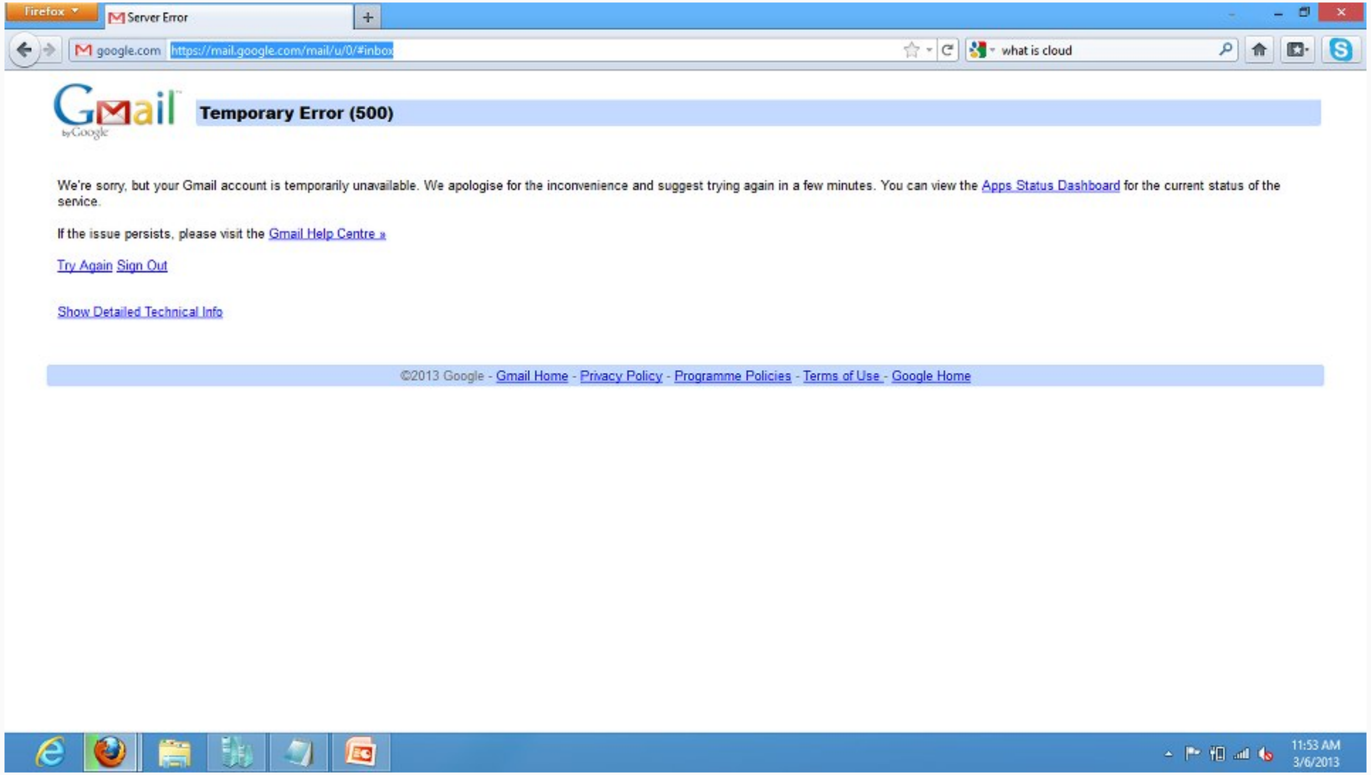
Click this button to goto Setting of you Gmail.

Looking For a Used Car? - www.Khojle.in - Find Used Cars in Great Condition & Best Price. Huge Range Available

About these ads updated

Select: All, None, Read, Unread, Starred, Unstarred

Or Gmail 500?



Temporary Error (500)

We're sorry, but your Gmail account is temporarily unavailable. We apologise for the inconvenience and suggest trying again in a few minutes. You can view the Apps Status Dashboard for the current status of the service.

If the issue persists, please visit the Gmail Help Centre »

Try Again Sign Out

Show Detailed Technical Info

©2013 Google - Gmail Home - Privacy Policy - Programme Policies - Terms of Use - Google Home

Reliability is easy to take for granted

- It’s the absence of errors
- Obviously unstable == too late
- You need to work at reliability all the time
  - Not just when everything’s on fire

SRE Organizational Structure

- The SRE Organization is separate from feature development
- SRE teams are organized around a single service or a collection of related services or technologies

Dev and Ops

- Don't Dev and Ops always fight?
  - Dev wants to...
    - ...roll out features fast
    - ...and see them widely adopted
  - Ops wants...
    - ...stability so they don't get paged

And just to make it harder...

- Information asymmetry is extreme
- Ops doesn’t really know the code base
- The team which knows the least about the code...
  - ...has the strongest incentive to object to it launching

Is conflict inevitable?

No :-)

- SRE doesn’t attempt to assess launch risk,
- or set release policy,
- or avoid all outages

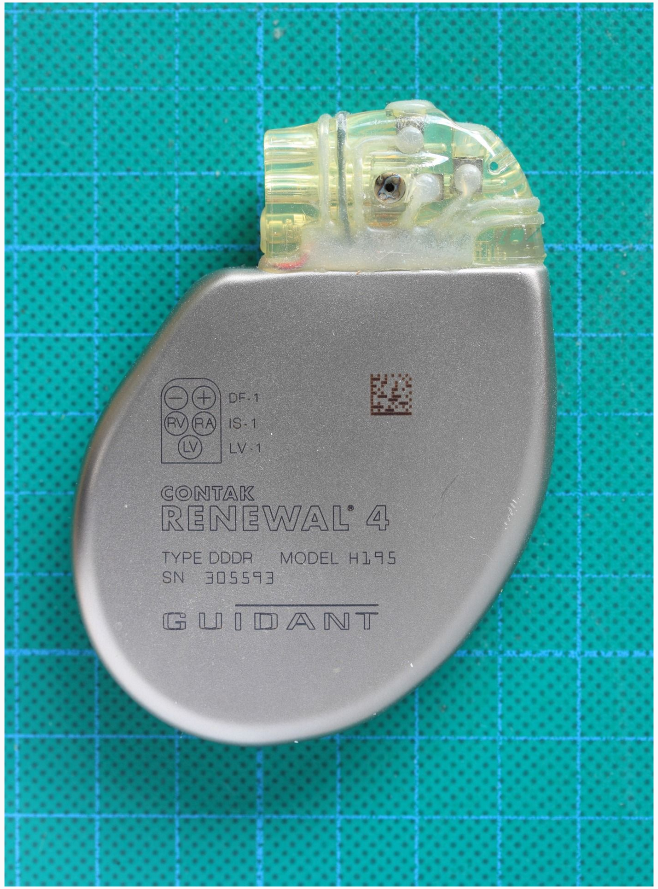
Then what?

- Error budgets!
- But you first need an SLO!

Service Level .*

- Service Level Indicator (**SLI**): a quantitative measure of an attribute of the service. It's a metric that users care about, such as:
    - *availability*
    - *latency*
    - *freshness*
    - *durability*
- Service Level Objective (**SLO**): SLI @ specific target (*99.9% availability* = □)
- Service Level Agreement (**SLA**): SLO + consequences (*99% availability* = ☹)

100% SLO


DF-1
IS-1
LV-1
RV
RA
LV
CONTAK
RENEWAL® 4
TYPE DDDR MODEL H195
SN 305593
GUIDANT

<100% SLO

https://pixabay.com/en/laptop-black-blue-screen-monitor-33521/

https://pixabay.com/en/computer-desktop-workstation-office-158675/

- Google doesn't run at 100% SLO
- Impossible to achieve
- Very expensive

Error Budget

- 1 - SLO
- Example
  - SLO: 99.9%
  - Error budget: 100% - 99.9% = 0.1%
  - Can spend this
  - For a 1 billion query/month service
    - 1 million "errors" to spend

What do you spend your budget on?

- Change is #1 cause of outage
- Launches are big sources of change
- Solution: Spend error budget on launches!
    - … or spend it on service instability :(

The rule

- Error budget > 0, launch away
  - Clearly DEV team is doing a good job
- Error budget < 0, launch freeze
  - Until you earn back enough error budget

Two nice features of Error Budgets

1. Removes major source SRE-DEV conflict
    a. It’s a math problem, not an opinion or power conflict
2. DEV teams self-police because they are not monolithic

Staffing, Work, Ops Overload

- At the core, you can throw people at a badly-functioning system and keep it alive via manual labor
- That job isn't fun
    - Google doesn't ask SREs to do it

But it’s soooo tempting?

- What I see is all there is
- Can’t see operations work = doesn’t exist
- It’s another incentives problem

Fix 1: Common Staffing Pool

- One more SRE = one less developer
- The more operations work...
  - ...the fewer features
- Self-regulating systems win!

Fix 2: SRE hires only coders

- They speak the same language as DEV
- They know what a computer can do
- They get bored easily

Fix 3: 50% cap on Ops work

- If you succeed, traffic increases
- Toil scales with traffic
- Write software to reduce toil
- Leave enough time for serious coding
    - ...or drown,
    - ...or fail

Fix 4: Keep DEV in the rotation

- “What I see is all there is”
- Dev team sees the product in action
- Not all teams do this though

Fix 5: Speaking of Dev and Ops work...

- Excess operations load gets assigned to the dev team
    - tickets, oncall, etc.
- Another self-regulating system :)

Fix 6: SRE Portability

- No requirement to stick with any project
  - No requirement to stick with SRE
- Build it and they will come
  - Bust it, and they will leave
- The threat is rarely executed, but it is powerful

Limiting operational work

1. Single staffing pool
2. Hire coders
3. Ops work < 50%
4. Dev involved in operations
5. Excessive toil → Dev
6. Mobility

Death, taxes, and outages...

- SLO < 100% means that there will be outages
  - This is OK. Not fun, but OK
- Two goals for each outage:
  - Minimize impact
  - Prevent recurrence

Minimize Damage

- Make the outage as short as possible
- No NOC
- Good diagnostic information

A word on practice...

Operational readiness drills aren’t cool.

You know what’s cool?

Wheel of Misfortune!

One of our most popular SRE events.

Prevent recurrence

- Step 1: Handle the event
- Step 2: Write the post-mortem
- Step 3: Reset

Post-mortem philosophy

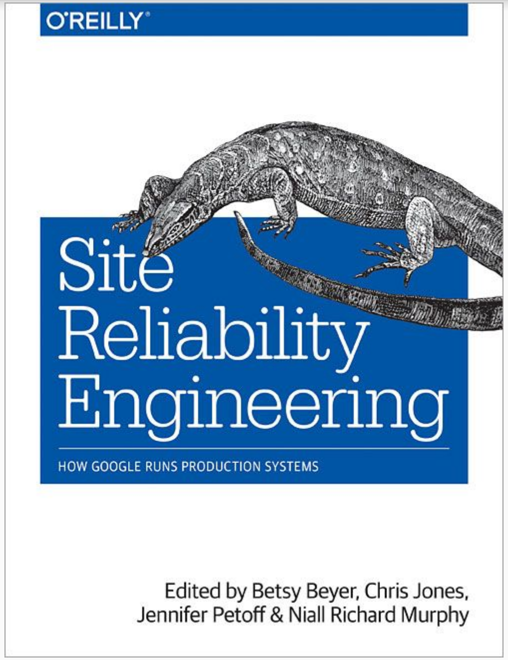
- Post-mortems are blameless
- Assume people are intelligent, well-intentioned
- Focus on process and technology

- Create a timeline
- Get all the facts
- Create bugs for all follow-up work

Google's SRE Website

- https://www.google.com/sre
- More resources
- Articles
- Videos

O'Reilly Book


O'REILLY

Site Reliability Engineering

HOW GOOGLE RUNS PRODUCTION SYSTEMS

Edited by Betsy Beyer, Chris Jones, Jennifer Petoff & Niall Richard Murphy

- Site Reliability Engineering
- How Google Runs Production Systems
- landing.google.com/sre/book.html

Questions on any of these?

- Reliability is the most important feature
- SRE = a dedicated team focused on reliability
  - Software engineering, consulting, on-call
- SLO is the target. Error budget is there to be spent
  - Divert SWE resources to reliability when you run out of error budget
- Limiting operational work
- Incident response and postmortems

goto; amsterdam

Please

Remember to rate this session

Thank you!