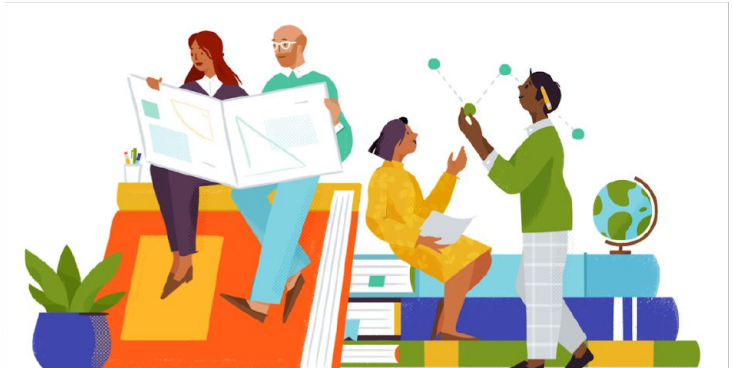
slack

April 25, 2018

Designing and launching the next-generation database system: from whiteboard to production

Guido Iaquinti



$whoami

Guido Iaquinti

- Operations Engineer in Dublin 🇮🇪
- Member of the storage team 💾
- No previous DBA experience

github.com/guidoiaquinti

twitter.com/guidoiaquinti

$whoami

Guido Iaquinti

- Operations Engineer in Dublin 🇮🇪
- ~~Member of the storage team~~ 💾
- No previous DBA experience

github.com/guidoiaquinti

twitter.com/guidoiaquinti

Agenda

1. Slack's current database system ⭐
2. Project Xarding 🔍
3. Next-gen database system 
4. Breakout discovery 📚
5. Conclusions 🎉

Slack's current database system

What is Slack today?

- 9+ million weekly active users
- 4+ million simultaneously connected
- Average 10+ hours/ weekday connected
- $200M+ in annual recurring revenue
- 1000+ employees across 7 offices

What is Slack today?

20+ billion database queries per day

170+ Gbps (database layer network throughput at peak)

2.1 PB of database storage

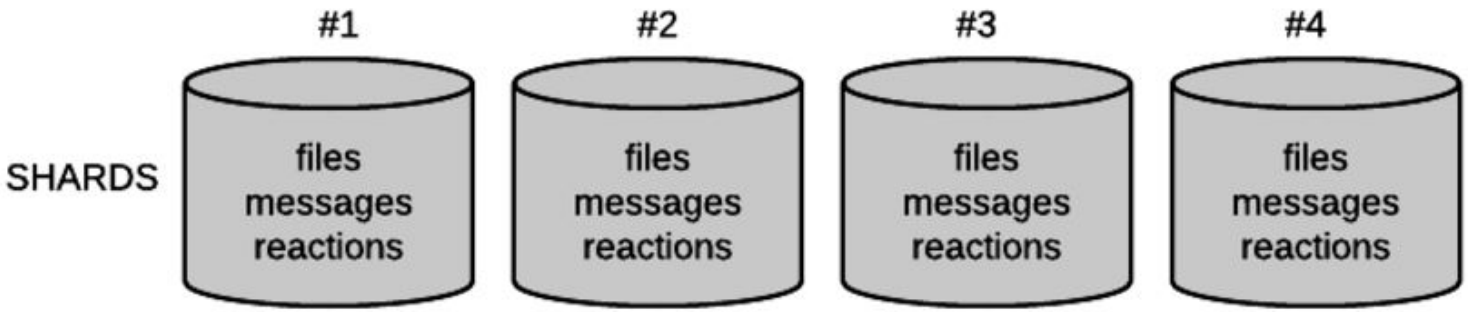
Thousands of database servers

What is Slack today?

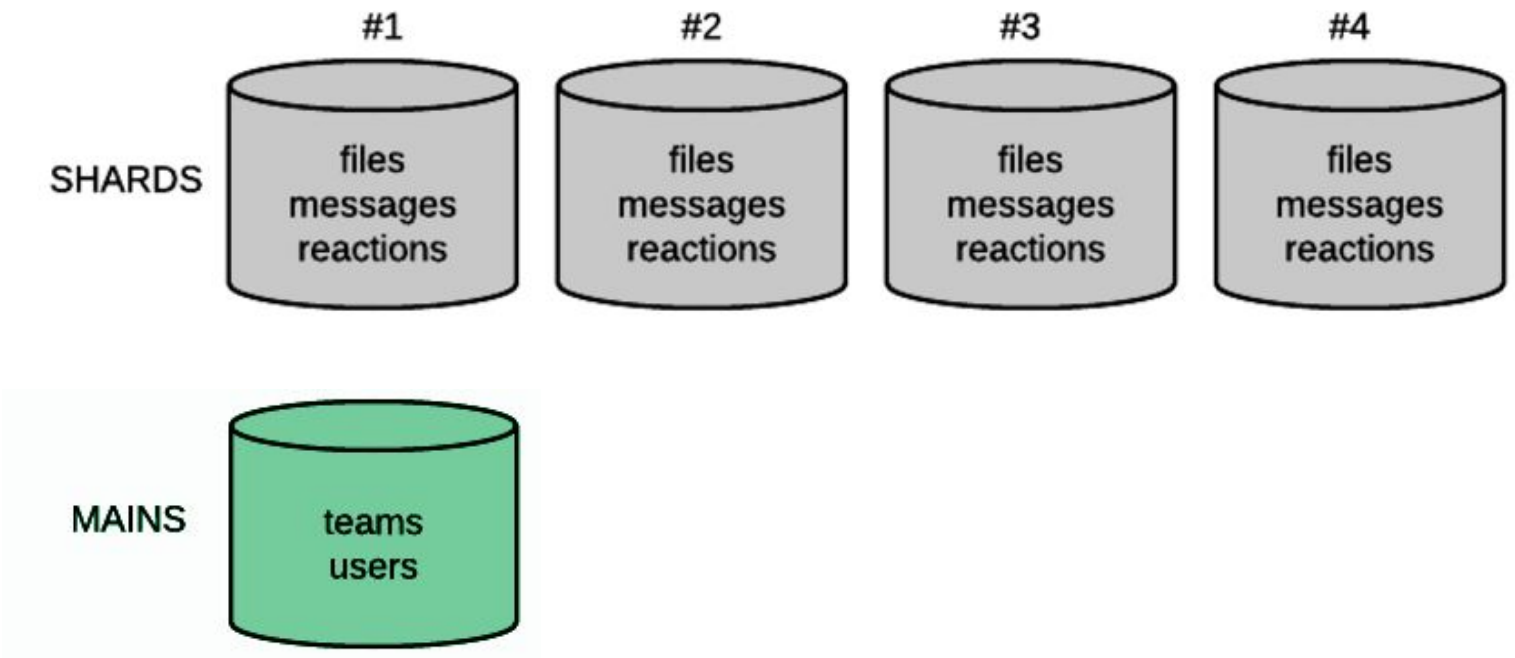
- Evolving from a LAMP stack 
- MySQL as primary storage system: single source of truth
- Custom sharding topology: allow us to scale horizontally—and sometimes vertically 🤷‍♀️

Database clusters

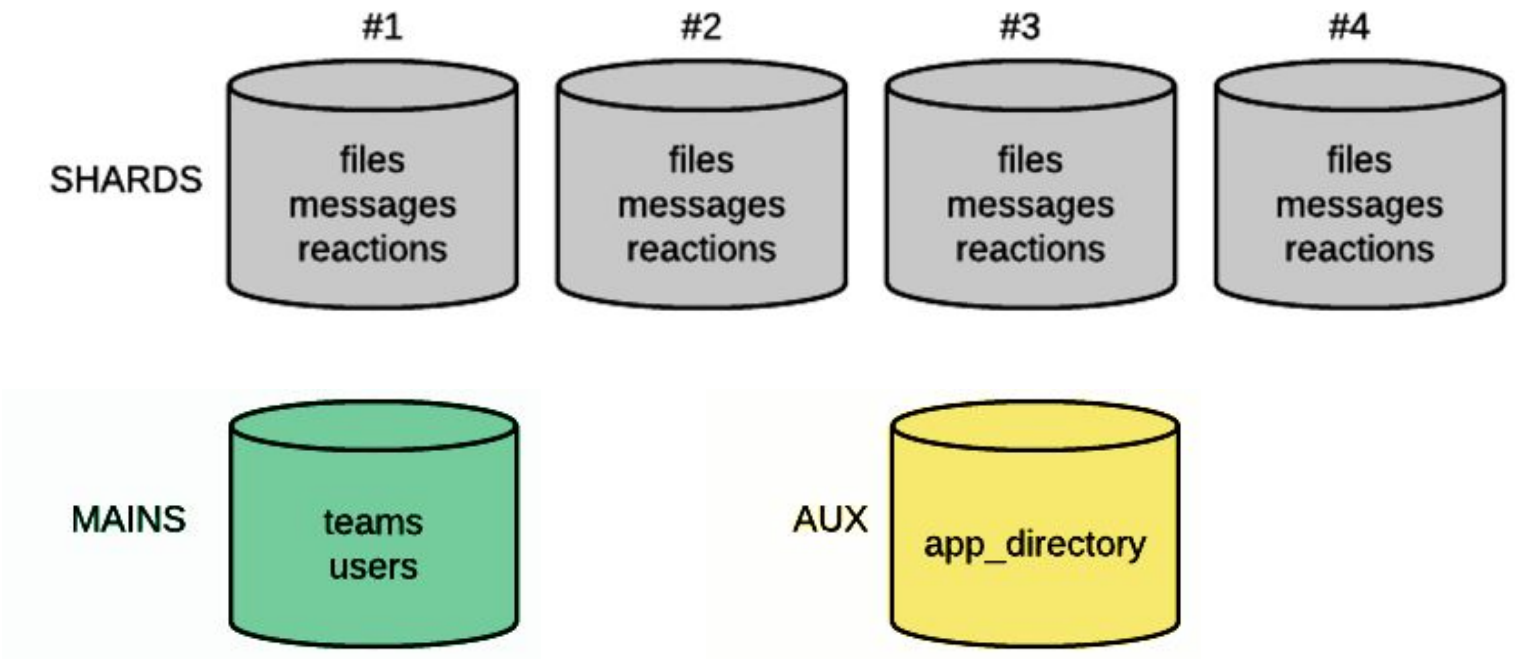
Database clusters



Database clusters



Database clusters



Database infrastructure

- MySQL on AWS EC2 instances
- SSD-based instance storage (no EBS)
- Each cluster is deployed across multiple AZ
- MySQL 5.6 (Percona)

MySQL Master-Master

- **Each cluster is a MySQL pair deployed in Master-Master configuration:** using async replication each master is also a slave of the other… master

- **Designed to prefer availability over consistency**

- **Unique IDs generated by an external service**: we can’t use IDs generated by MySQL, we need to have primary keys globally unique

- **Which master should the application use?** mostly by primary key: odd keys on side A, even keys on side B

Current architecture

Current architecture 👍

- Availability not impacted if a master goes down
- We can horizontally scale by splitting “hot” pairs
- With the asynchronous M-M setup writes are as fast as the node can provide
- “Online” schema changes

Current architecture 👎

- A team can’t grow beyond a single MySQL pair
- Low resource usage: our bottleneck is the SQL replication
- There’s no value to adding read replicas
- Requires Statement Based Replication
- Operational overhead: manual resolution of inconsistent entries

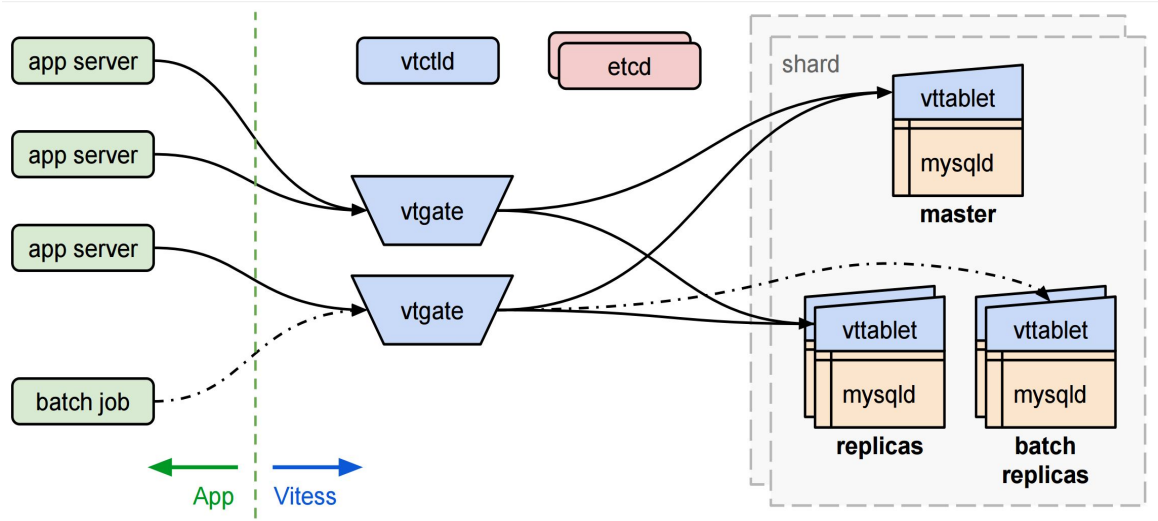
Project Xarding

Requirements

- Sharding must be more granular than by an entire team
- Minimal changes to application code
- Decouple infrastructure and code
- Operator overhead / # servers -> O(1)
- Maintenance is hidden from the end user: no user-visible downtime

Vitess

- Open source project by YouTube (Google)
- Built on top of MySQL replication and InnoDB
- Uses sharding best practices: shared-nothing & consistent hashing



Proposal document 

Xarding: Vitess at Slack Initial Evaluation

Project Members: Mike Demmer, Saroj Yadav
Last Updated: December 21, 2016

(Note: Mike to update with results from initial checkpoint meeting)

Background
The goal of this project is to evaluate using Vitess as an enabling technology for more granular and flexible sharding of our database tables including both finer-grained sharding of per-team data by things such as channel, user, etc, or sharding the single points of failure like mains and aux, aka the “Xarding” project.

Executive Summary
Based on the initial examination, Vitess is a promising technology to enable generic sharding and database management. As compared to an in-house solution (e.g. channel sharding), it would require a more significant initial investment and would have a major long term impact on how we manage our data.

In order to put vitess through enough of an evaluation to decide whether it is actually viable, a further evaluation would take an investment of ~3 engineers (2 development 1 operations) for 2-3 months. Our recommendation is that this investment is worthwhile given the potential benefits.

January 17th, 2017

Mike Demmer 20:43

Hi there.

Ryan told me the good news that you'll be helping out with the Vitess project ✨ 💥
That'll be great from my perspective.

The Vitess team is built

Next-gen database system

Q1 Project Planning

February

- Vitess cluster up and running in DEV

March

- Vitess cluster up and running in PROD
- Develop double read/write experiment
- Planning for larger table migration

April

- Ship double read/write experiment in PROD
- Conclude Vitess go/no-go & plan the rest of 2017

MySQL legacy VS MySQL new

- **Topology**: master-master VS master-slave
- **Replication**: full async VS semi-sync (not strictly required)
- **Binlog**: replication position VS global transaction id

First steps

- **Build**: internal public fork synced upstream. Codebase tested and build by Jenkins, artifacts uploaded to S3
- **Config**: managed by Chef
- **Deploy**: on EC2 instances (no containers)
- **Automate**: work in progress, still trying to figure out what to automate and how Vitess works...
- **Monitoring**: work in progress, mostly reactive

Bleeding edge technology

Bleeding edge technology

Guido Iaquinti DUB 14:32

I've had a chat with Alan Jobart from Google to change this behavior https://github.com/youtube/vitess/issues/2586 and he agreed that the current default doesn't make a lot of sense in most of the setup (edited)

> **Tablet directory structure · Issue #2586 · youtube/vitess · GitHub**
> vitess - Vitess is a database clustering system for horizontal scaling of MySQL.

Bleeding edge technology

Guido Iaquinti DUB 14:32

I've had a chat with Alan Jobart from Google to change this behavior https://github.com/youtube/vitess/issues/2586 and he agreed that the current default doesn't make a lot of sense in most of the setup (edited)

> **Tablet directory structure · Issue #2586 · youtube/vitess · GitHub**
> vitess - Vitess is a database clustering system for horizontal scaling of MySQL.

Mike Demmer 14:14

ARGH!! This is another case where the examples aren't up to date with the code...

This format worked:

```
{
  "sharded": true,
  "vindexes": {
    "hash": {
      "type": "hash"
    }
  },
  "tables": {
    "messages": {
      "column_vindexes": [
        {
          "column": "page",
          "name": "hash"
        }
      ]
    }
  }
}
```

I modeled after:

https://github.com/youtube/vitess/blob/46b2d127697749a9da73015ce0dcfedc28c41c8e/data/test/vtgate/schema_test.json

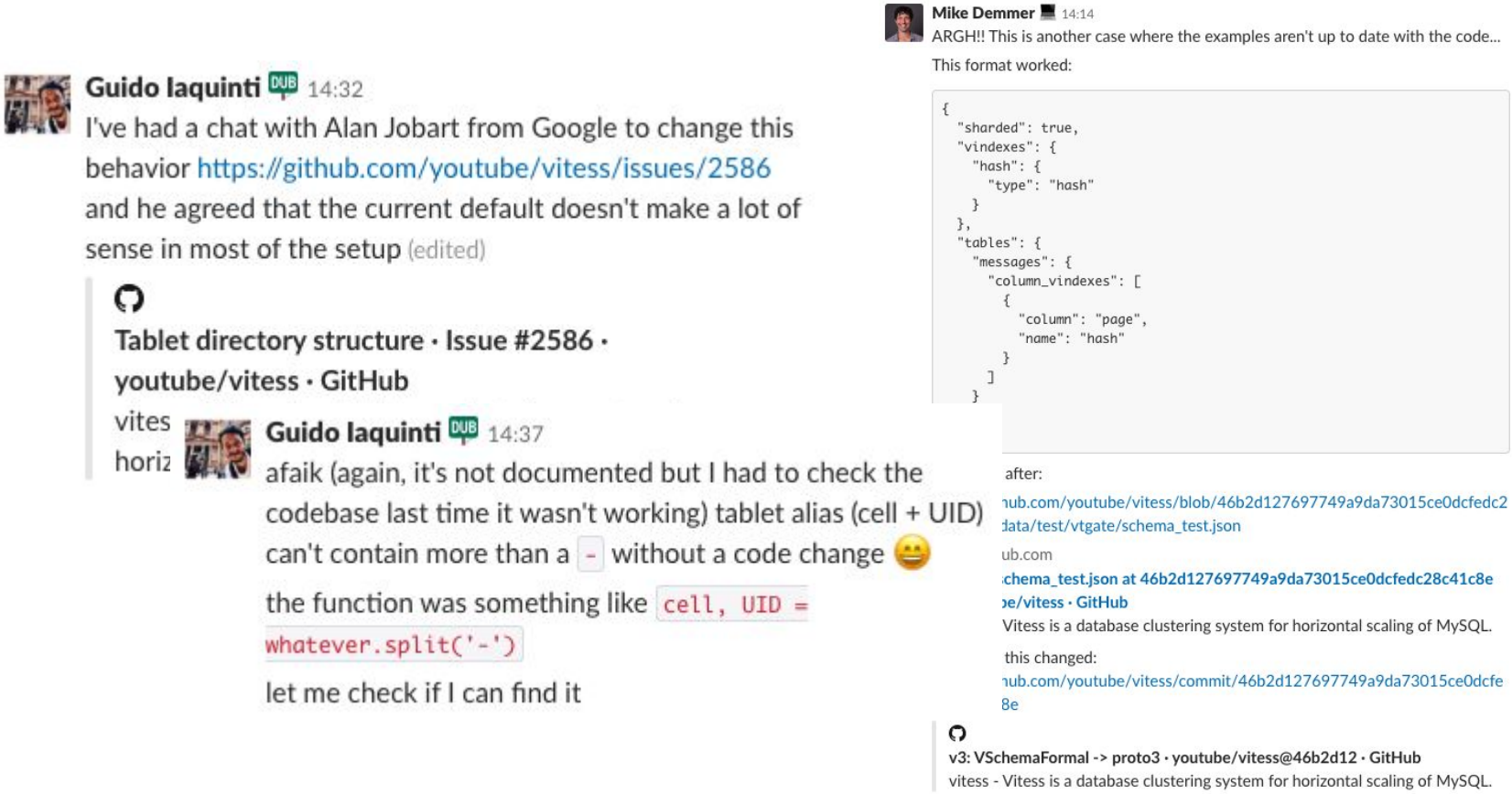
> github.com
> **vitess/schema_test.json at 46b2d127697749a9da73015ce0dcfedc28c41c8e · youtube/vitess · GitHub**
> vitess - Vitess is a database clustering system for horizontal scaling of MySQL.

Looks like this changed:

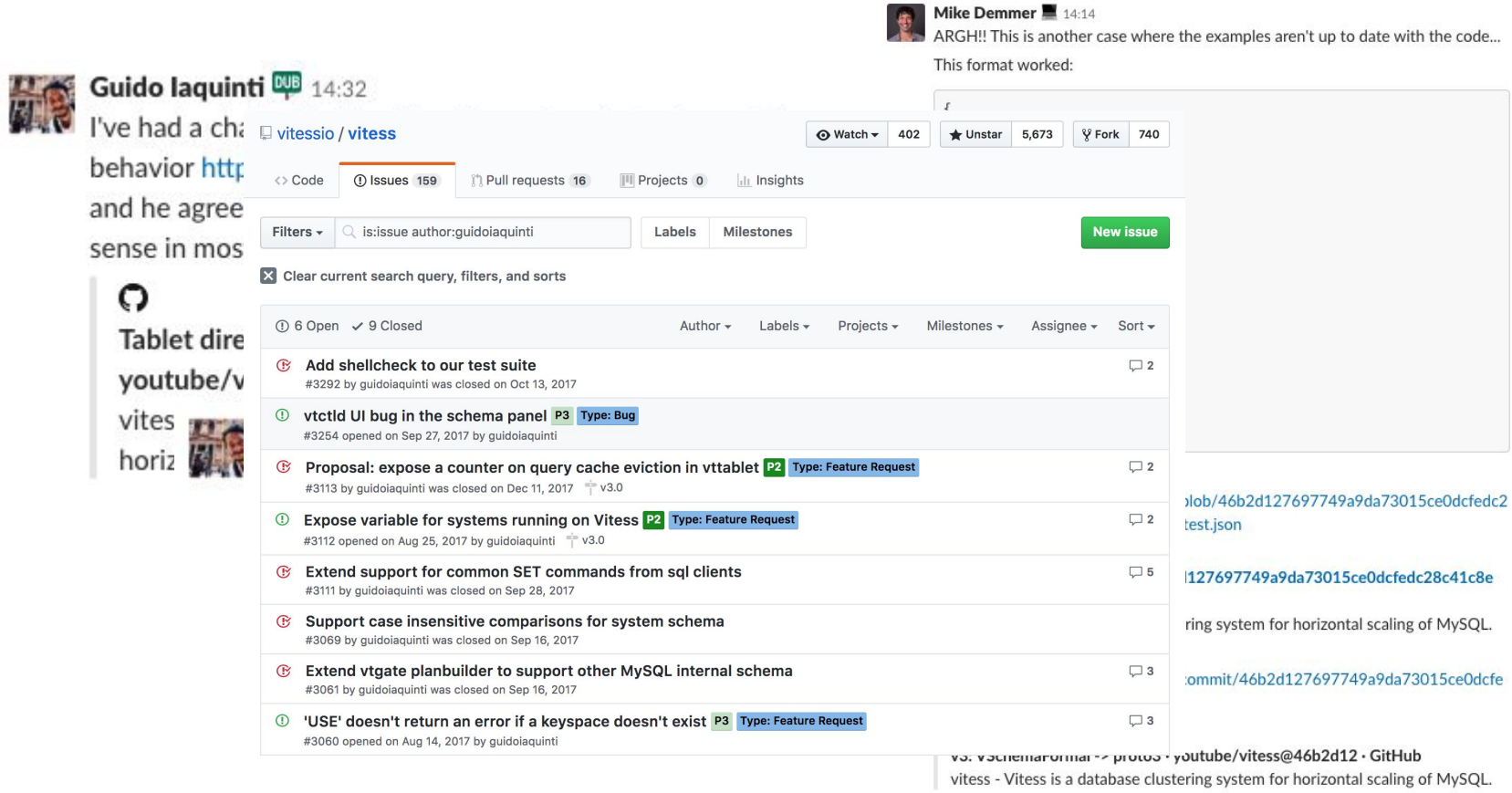
https://github.com/youtube/vitess/commit/46b2d127697749a9da73015ce0dcfedc28c41c8e

> **v3: VSchemaFormal -> proto3 · youtube/vitess@46b2d12 · GitHub**
> vitess - Vitess is a database clustering system for horizontal scaling of MySQL.

Bleeding edge technology



Bleeding edge technology

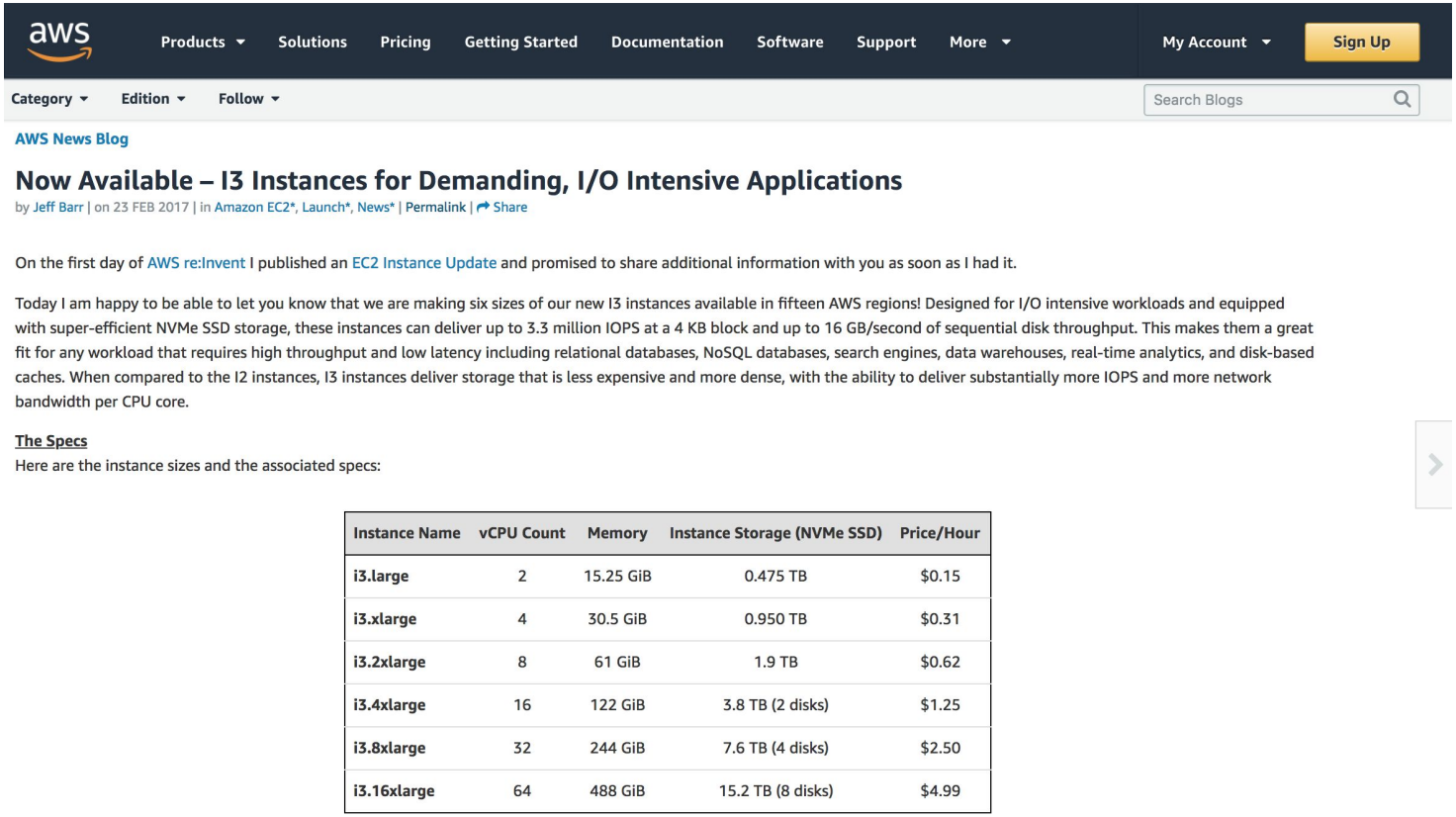


Iterate over the first steps

- **Infrastructure as code**: manage AWS resources via Terraform
- **Service discovery & load balancing:** via AWS ELB
- **Metrics:** custom exporter for expvar -> statsd
- **Logging:** make it working with our ingestion pipeline

Change of plans

Change of plans



aws Products Solutions Pricing Getting Started Documentation Software Support More My Account Sign Up

Category Edition Follow Search Blogs

AWS News Blog

Now Available – I3 Instances for Demanding, I/O Intensive Applications

by Jeff Barr | on 23 FEB 2017 | in Amazon EC2*, Launch*, News* | Permalink | Share

On the first day of AWS re:Invent I published an EC2 Instance Update and promised to share additional information with you as soon as I had it.

Today I am happy to be able to let you know that we are making six sizes of our new I3 instances available in fifteen AWS regions! Designed for I/O intensive workloads and equipped with super-efficient NVMe SSD storage, these instances can deliver up to 3.3 million IOPS at a 4 KB block and up to 16 GB/second of sequential disk throughput. This makes them a great fit for any workload that requires high throughput and low latency including relational databases, NoSQL databases, search engines, data warehouses, real-time analytics, and disk-based caches. When compared to the I2 instances, I3 instances deliver storage that is less expensive and more dense, with the ability to deliver substantially more IOPS and more network bandwidth per CPU core.

The Specs

Here are the instance sizes and the associated specs:

Instance Name	vCPU Count	Memory	Instance Storage (NVMe SSD)	Price/Hour
i3.large	2	15.25 GiB	0.475 TB	$0.15
i3.xlarge	4	30.5 GiB	0.950 TB	$0.31
i3.2xlarge	8	61 GiB	1.9 TB	$0.62
i3.4xlarge	16	122 GiB	3.8 TB (2 disks)	$1.25
i3.8xlarge	32	244 GiB	7.6 TB (4 disks)	$2.50
i3.16xlarge	64	488 GiB	15.2 TB (8 disks)	$4.99

Change of plans

February 24th, 2017

Richard Crowley 03:27

https://tinyspeck.slack.com/archives/ops-random/p1487901811130337

We should immediately make all Vitess tablet instances `i3.large` and never look back.

> **Ryan Park**
>
> EC2 i3 instances are generally available: https://aws.amazon.com/blogs/aws/now-available-i3-instances-for-demanding-io-intensive-applications/
>
> Posted in #ops-random | Feb 24th, 2017

Change of plans

- **Fact:** i3 uses NMVe storage
- **Kernel support:** was added on 3.3 but AWS suggests to use >=4.4
- **OS:** Ubuntu 14.04 ships kernel 3.x (and we don’t like to backport)
- **Decision:** deploy the new system on Ubuntu 16.04

Add i3 support in Slack

Vitess is the first service at Slack to use AWS i3 and Ubuntu 16.04

- required to add support to our provisioning system
- required to add support to our config management system
- validate setup and fix any security regression
- validate setup and fix any performance regression

i3 another bleeding edge component

i3 another bleeding edge component

```
Mar 31 13:29:32 slack-tablet-yahs65s8 kernel: EXT4-fs warning (device nvme0n1p1): ext4_end_bio:329: I/O er
Mar 31 13:29:32 slack-tablet-yahs65s8 kernel: Buffer I/O error on device nvme0n1p1, logical block 2861847
Mar 31 13:29:32 slack-tablet-yahs65s8 kernel: Buffer I/O error on device nvme0n1p1, logical block 2861848
Mar 31 13:29:32 slack-tablet-yahs65s8 kernel: Buffer I/O error on device nvme0n1p1, logical block 2861849
Mar 31 13:29:32 slack-tablet-yahs65s8 kernel: Buffer I/O error on device nvme0n1p1, logical block 2861850
Mar 31 13:29:32 slack-tablet-yahs65s8 kernel: Buffer I/O error on device nvme0n1p1, logical block 2861851
Mar 31 13:29:32 slack-tablet-yahs65s8 kernel: Buffer I/O error on device nvme0n1p1, logical block 2861852
Mar 31 13:29:32 slack-tablet-yahs65s8 kernel: Buffer I/O error on device nvme0n1p1, logical block 2861853
Mar 31 13:29:32 slack-tablet-yahs65s8 kernel: Buffer I/O error on device nvme0n1p1, logical block 2861854
Mar 31 13:29:32 slack-tablet-yahs65s8 kernel: Buffer I/O error on device nvme0n1p1, logical block 2861855
Mar 31 13:29:32 slack-tablet-yahs65s8 kernel: Buffer I/O error on device nvme0n1p1, logical block 2861856
```

(storage edition)

i3 another bleeding edge component

Ubuntu
linux package

Overview Code **Bugs** Blueprints Translations Answers

Amazon I3 Instance Buffer I/O error on dev nvme0n1

Bug #1668129 reported by Pete Cheslock on 2017-02-27

This bug affects 26 people 142

Affects	Status	Importance	Assigned to	Milestone
linux (Ubuntu)	Won't Fix	Critical	Dan Streetman	
Xenial	Won't Fix	Critical	Dan Streetman	
linux-aws (Ubuntu)	Fix Released	Critical	Dan Streetman	
Xenial	Fix Released	Critical	Dan Streetman	

Also affects project Also affects distribution/package Nominate for series

Bug Description

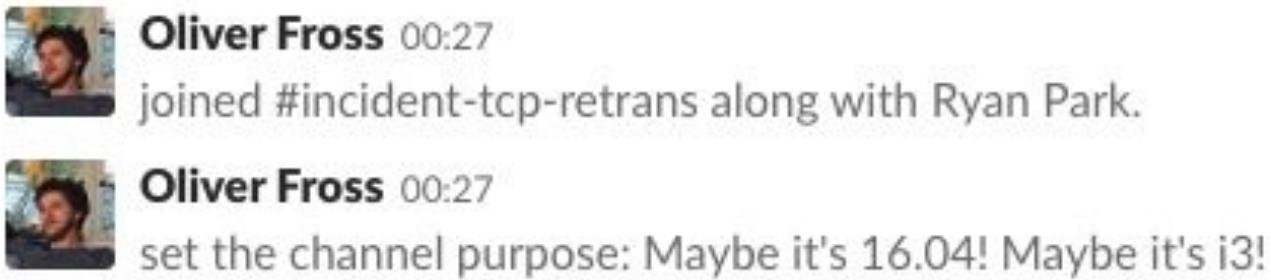
```
On the AWS i3 instance class - when putting the new NVME storage disks
under high IO load - seeing data corruption and errors in dmesg

[ 662.884390] blk_update_request: I/O error, dev nvme0n1, sector 120063912
[ 662.887824] Buffer I/O error on dev nvme0n1, logical block 14971093,
lost async page write
[ 662.891254] Buffer I/O error on dev nvme0n1, logical block 14971094,
lost async page write
[ 662.895591] Buffer I/O error on dev nvme0n1, logical block 14971095,
lost async page write
[ 662.899873] Buffer I/O error on dev nvme0n1, logical block 14971096,
lost async page write
[ 662.904179] Buffer I/O error on dev nvme0n1, logical block 14971097,
lost async page write
```

(storage edition)

i3 another bleeding edge component

i3 another bleeding edge component



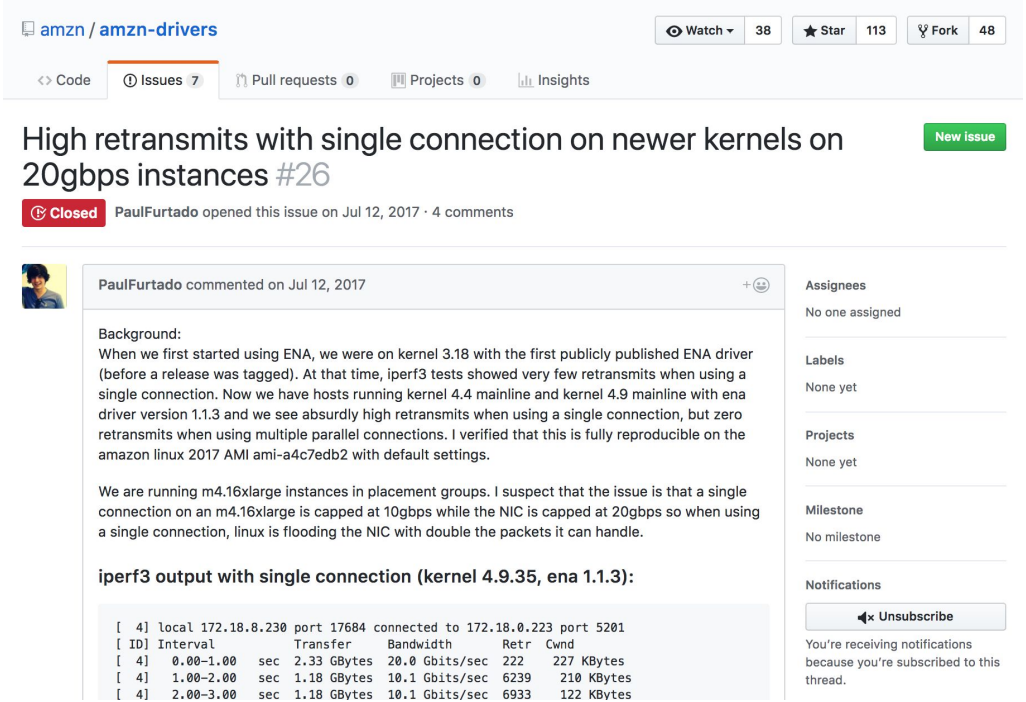
(network edition)

i3 another bleeding edge component

- **Theory 1:** is it 16.04 vs 14.04?
- **Theory 2:** is it i3/r4 specific?
- **Theory 3:** is it the ENA interface driver?
- **Theory 4:** look at rto!
- **Theory 5:** tcp_mem is smaller on 16.04 than 14.04

(network edition)

i3 another bleeding edge component



amzn / amzn-drivers

Watch 38 | Star 113 | Fork 48

Code | Issues 7 | Pull requests 0 | Projects 0 | Insights

High retransmits with single connection on newer kernels on 20gbps instances #26

New issue

Closed **PaulFurtado** opened this issue on Jul 12, 2017 · 4 comments

PaulFurtado commented on Jul 12, 2017

Background:
When we first started using ENA, we were on kernel 3.18 with the first publicly published ENA driver (before a release was tagged). At that time, iperf3 tests showed very few retransmits when using a single connection. Now we have hosts running kernel 4.4 mainline and kernel 4.9 mainline with ena driver version 1.1.3 and we see absurdly high retransmits when using a single connection, but zero retransmits when using multiple parallel connections. I verified that this is fully reproducible on the amazon linux 2017 AMI ami-a4c7edb2 with default settings.

We are running m4.16xlarge instances in placement groups. I suspect that the issue is that a single connection on an m4.16xlarge is capped at 10gbps while the NIC is capped at 20gbps so when using a single connection, linux is flooding the NIC with double the packets it can handle.

iperf3 output with single connection (kernel 4.9.35, ena 1.1.3):

```
[  4] local 172.18.8.230 port 17684 connected to 172.18.0.223 port 5201
[ ID] Interval           Transfer     Bandwidth       Retr  Cwnd
[  4]   0.00-1.00   sec  2.33 GBytes  20.0 Gbits/sec  222    227 KBytes
[  4]   1.00-2.00   sec  1.18 GBytes  10.1 Gbits/sec  6239    210 KBytes
[  4]   2.00-3.00   sec  1.18 GBytes  10.1 Gbits/sec  6933    122 KBytes
```

Assignees
No one assigned

Labels
None yet

Projects
None yet

Milestone
No milestone

Notifications
Unsubscribe
You're receiving notifications because you're subscribed to this thread.

(network edition)

i3 another bleeding edge component

Guido Iaquinti DUB 14:59

regarding the ENA bug:

> It's been escalated to basically the entire Networking team along with one of the EC2 teams.

(network edition)

i3 another bleeding edge component

Guido Iaquinti DUB 16:30

I burnt the few lonely neurons that I had left in my 🧠 to fix few blockers on the way to get our new AMI with ENA drivers and with the i3 bug fixed:

```
filename:       /lib/modules/4.4.0-1013-
aws/kernel/drivers/net/ethernet/amazon/ena/ena.ko
version:        1.1.2
license:        GPL
description:    Elastic Network Adapter (ENA)
author:         Amazon.com, Inc. or its affiliates
srcversion:     53573A34DAA90B1BD1A0994
alias:          pci:v00001D0Fd0000EC21sv*sd*bc*sc*i*
alias:          pci:v00001D0Fd0000EC20sv*sd*bc*sc*i*
alias:          pci:v00001D0Fd00001EC2sv*sd*bc*sc*i*
alias:          pci:v00001D0Fd00000EC2sv*sd*bc*sc*i*
depends:
intree:         Y
vermagic:       4.4.0-1013-aws SMP mod_unload modversions
parm:           debug:Debug level (0=none,...,16=all) (int)
```

(network edition)

😇 vs 😈 engineer

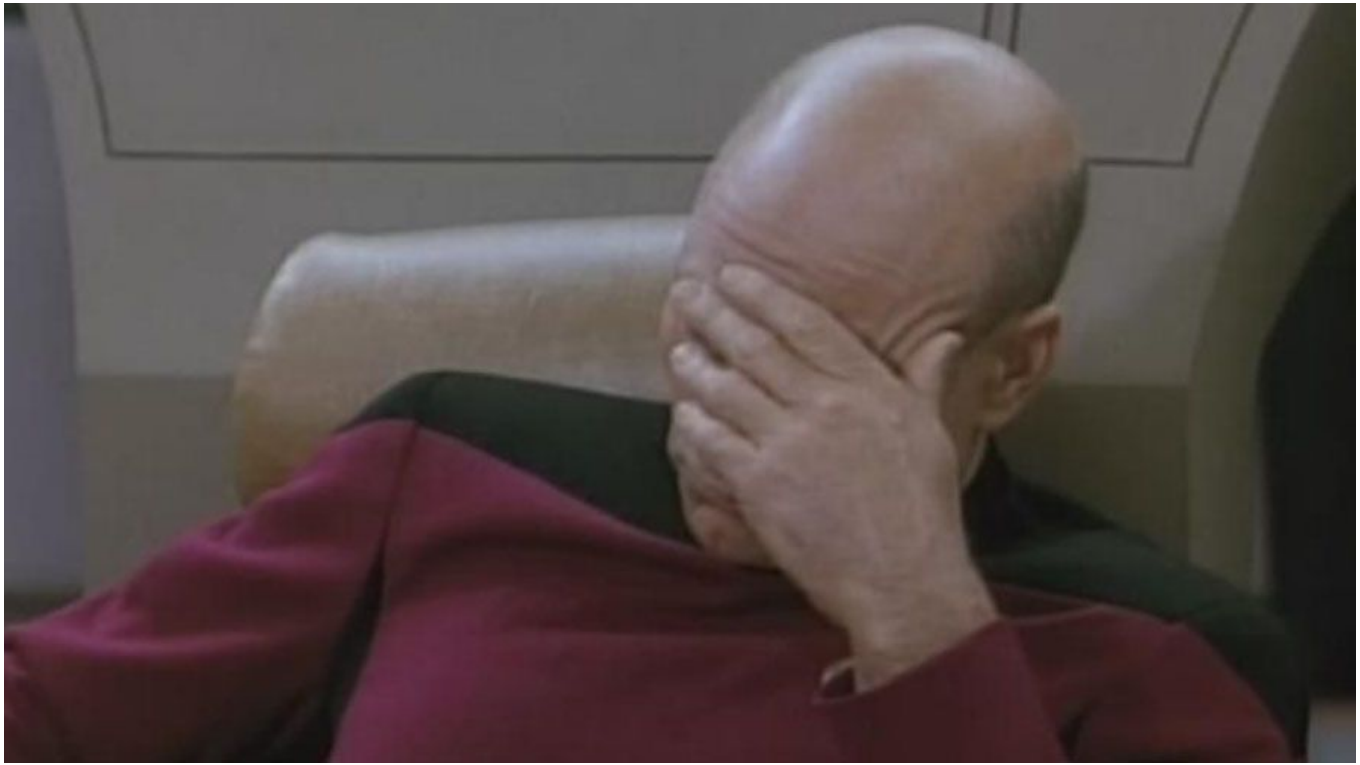
 engineer

- MySQL 5.6 -> MySQL 5.7
- Non strict mode -> strict mode
- SBR (Statement Based Replication) -> RBR (Row Based Replication)
- PHP mysqli driver -> HHVM async MySQL driver

Upgrade 

Changelog

- **AWS instance:** i2 -> i3
- **OS:** Ubuntu 14.04 -> Ubuntu 16.04
- **MySQL version:** 5.6 -> 5.7
- **MySQL Topology**: master-master VS master-slave
- **MySQL Replication**: full async VS semi-sync
- **MySQL Binlog**: replication position VS global transaction id
- **MySQL Strict mode:** OFF -> ON
- **MySQL Binlog format:** SBR -> RBR
- **App driver:** PHP mysqli driver -> HHVM async MySQL driver
- **App logic:** r/w on master -> read on replicas
- **Metric collection:** statsd -> Prometheus



End of Q1 (Feb-Apr)

- **Clusters up and running:** in DEV and PROD
- **Service discovery & load balancing:** via AWS ELB
- **Manual processes designed, documented, and tested for:**
  - schema changes
  - shard split
  - master failover/election
- **Backup & restore:** automated, tested and documented

End of Q1 (Feb-Apr)

- **Clusters up and running:** in DEV and PROD
- **Service discovery & load balancing:** via AWS ELB
- **Manual processes designed, documented, and tested for:**
  - schema changes
  - shard split
  - master failover/election

- **Backup & restore:** automated, tested and documented

The Vitess team is growing





Q2 Project Planning (May-Jul)

Q2 Project Planning (May-Jul)

🔐 Security

- Complete full security review
- Ensure all database accounts and grants are managed automatically
- Ensure all credentials are distributed via Vault

🗿 Durability

- Simulate and verify application and customer impact for the loss of each component and service dependency
- Ensure backups are stored in a locked-down backup account and replicated cross-region

Q2 Project Planning (May-Jul)

📈 Availability

- Ensure 100% of master failover/recovery are automatically handled
- Simulate and verify the ability auto recover from the loss of AZs
- Ensure we get alerts for any error conditions that could affect availability

⚒️ Operational tooling

- Data warehouse ingestion
- Develop procedure and runbooks for troubleshooting hotspots, badly behaving clients or servers

mysql-grants

🔐 mysql-grants

Internal CLI tool to manage MySQL accounts & permissions

Input

- config file with user and policy definitions
- credentials file

Output

- SQL to execute
- directly execute SQL against `--target` if the `--execute` flag is passed

🔐 mysql-grants

- Uses “AWS IAM” concepts in MySQL
  - Users
  - Policies
  - Privileges
- Allow whitelist/blacklist policies
- Allow global, database or table scope privileges
- Exposes API bindings

mysql-grants

Example config

```
# Users for monitoring & metrics export
slack_monitoring_a@localhost:
  max_user_connections: 3
  policies:
    monitoring:
      remove: False
    metric_export:
      remove: False
```

```
# MySQL Monitoring
monitoring:
  type: whitelist
  privileges:
    SELECT:
      "slack_monitoring.*":
        remove: False
      "performance_schema.*":
        remove: False
    UPDATE:
      "slack_monitoring.*":
        remove: False
    INSERT:
      "slack_monitoring.*":
        remove: False
    CREATE:
      "slack_monitoring.*":
        remove: False
    DELETE:
      "slack_monitoring.*":
        remove: False
```

mysql-grants

Example whitelist/blacklist policy

```yaml
# MySQL Monitoring
monitoring:
  type: whitelist
  privileges:
    SELECT:
      "slack_monitoring.*":
        remove: False
      "performance_schema.*":
        remove: False
    UPDATE:
      "slack_monitoring.*":
        remove: False
    INSERT:
      "slack_monitoring.*":
        remove: False
    CREATE:
      "slack_monitoring.*":
        remove: False
    DELETE:
      "slack_monitoring.*":
        remove: False
```

```yaml
# Read access on non sensitive data
read_only_no_sensitive:
  type: blacklist
  privileges:
    SELECT:
      - channels.name
      - channels.purpose
      - channels.topic
```

mysql-grants

Example API usage

```
from mysql_grants.user import User
from mysql_grants.privilege import Privilege
from mysql_grants.policy import Policy

# Create user
user = User(username='guido', host='127.0.0.1')

# Create policy
policy = Policy(name='operator')
select_all = Privilege(privilege='SELECT', scope='*.*')
insert_all = Privilege(privilege='INSERT', scope='*.*')
policy.attach_privilege(select_all)
policy.attach_privilege(insert_all)

# Add policy to user
user.attach_policy(policy)

user.sql() # return a list of SQL statements to manage the user
           #
           # ["CREATE USER IF NOT EXISTS 'guido'@'127.0.0.1';",
           #  "GRANT SELECT ON *.* TO 'guido'@'127.0.0.1';",
           #  "GRANT INSERT ON *.* TO 'guido'@'127.0.0.1';"]
```

⚒ Operational tools

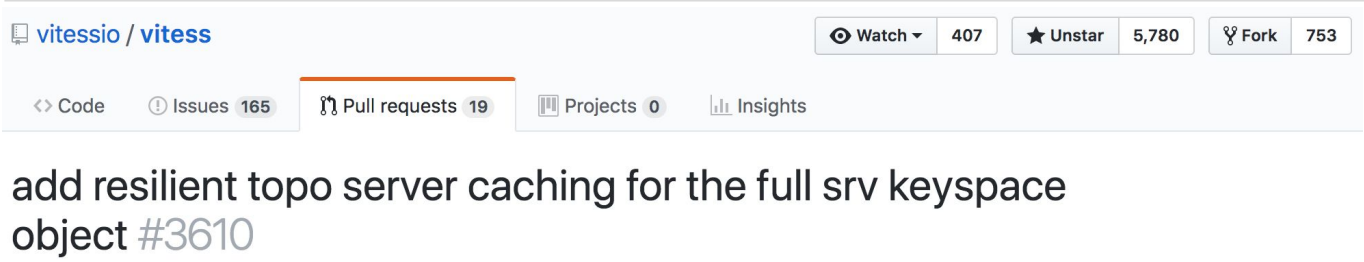
- Add Vitess support to SlackOps
- Monitoring + Visibility
  - custom exporter for Prometheus
  - add support for stacktraces in query comment
  - extend our MySQL visibility tools to support Vitess
- Alerting
- vtexplain

📈 #disasterpiece-theater 🗿

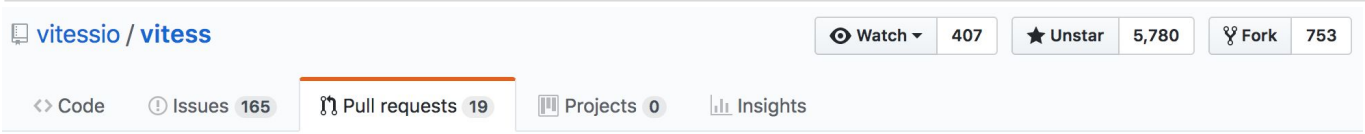
📈 #disasterpiece-theater 🗿

- **stateless components** (vtctld+vtgate): in ASG
- **stateful components** (vttablet+mysqld): non ASG (yet) but each shard is deployed with 4 semi-sync replicas in different AZs
- **metadata/topology storage:** based on Consul

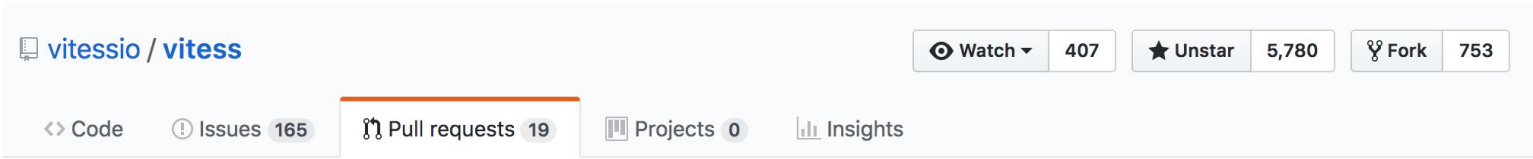
📈 #disasterpiece-theater 🗿



📈 #disasterpiece-theater 🗿

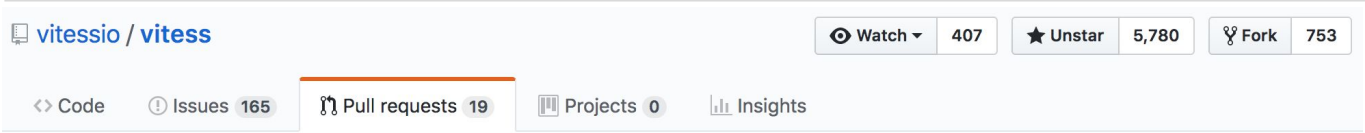


add resilient topo server caching for the full srv keyspace object #3610

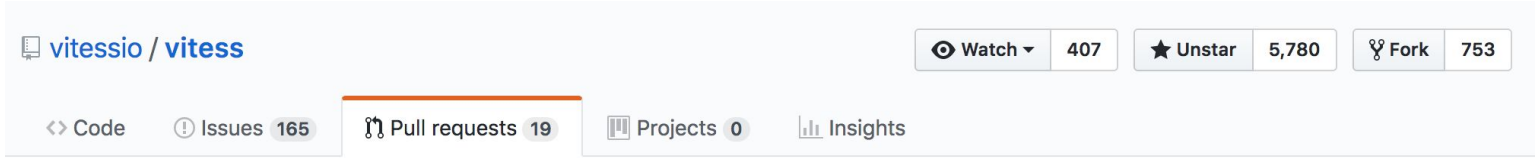


make the resilient topo cache even more resilient and informative #3641

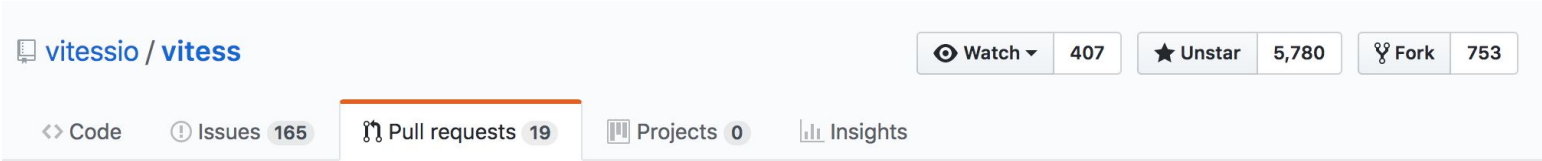
📈 #disasterpiece-theater 🗿



add resilient topo server caching for the full srv keyspace object #3610



make the resilient topo cache even more resilient and informative #3641



add a GetTopoServer method to srvtopo.Server #3740

📈 #disasterpiece-theater 🗿

Richard Crowley 🔑 14:57

The first Disasterpiece Theater exercise is **February 22 from 1:00pm to 3:30pm in Primrose at 500 Howard** (DM me for an invitation), featuring Rafael and some unplanned Vitess tablet reparenting. We'll be killing an important `mysqld` or maybe terminating an important EC2 instance to verify that recovery is automatic, safe, and unnoticeable. **If you're an engineer who would be involved in incident response during a Vitess- or webapp-related outage, you should come practice with us.** (edited)

✌ 2

📈 #disasterpiece-theater 🗿

📈 #disasterpiece-theater 🗿

Richard Crowley 🔑 22:22

Now, with that out of the way, we're going to do another disaster exercise, partitioning Vitess from Consul. This is starting **in dev** shortly. Follow along in #disasterpiece-theater.

🍿 1 😀 1

Richard Crowley 🔑 22:38

And that went as anticipated, so we'll be stopping the Consul agent on one `vtgate` instance **in prod**. This will be the last #disasterpiece-theater exercise for today.

Richard Crowley 🔑 22:43

#disasterpiece-theater is at an end for today. Thank you all for playing along.

🎉 2

📈 #disasterpiece-theater 🗿

Milo 22:43

who won?

📈 #disasterpiece-theater 🗿

📈 #disasterpiece-theater 🗿

End of Q2 (May-July)

- **Vitess is live:** serving one feature
- **Foundations:** we successfully built the foundations to support Vitess as the first-class data store

Now it’s time to make the setup more “*user friendly*” and increase internal adoption!

The Vitess team is growing (again!)







Q3 (Aug-Oct)

Adoption

- Documentation & best practices
- #triage-vitess
- Weekly office hours

Knowledge sharing

- Internal brownbags
- Presentations

Better tooling

- Automated deploy for Vitess components
- Schema changes

Breakout discovery

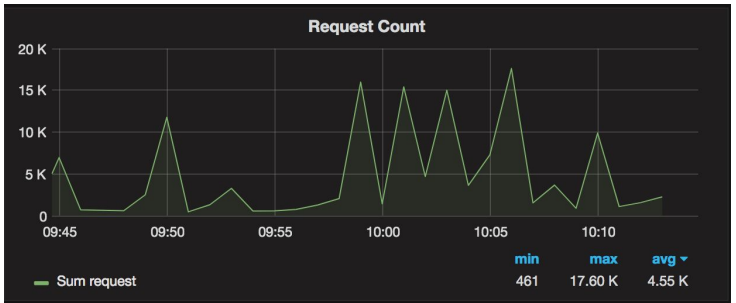
Breakout discovery (AWS)

Breakout discovery (AWS)

Elastic Load Balancer

Breakout discovery (AWS)

Elastic Load Balancer is not very “elastic” for short lived connections



Guido Iaquinti DUB 15:26

all the connection failures that I see are due to ELB latency

Guido Iaquinti DUB 17:22

let's burn with fire the ELB as 1st thing next week?

Mike Demmer 17:21

Yep.

I agree

Breakout discovery (AWS)

Service discovery

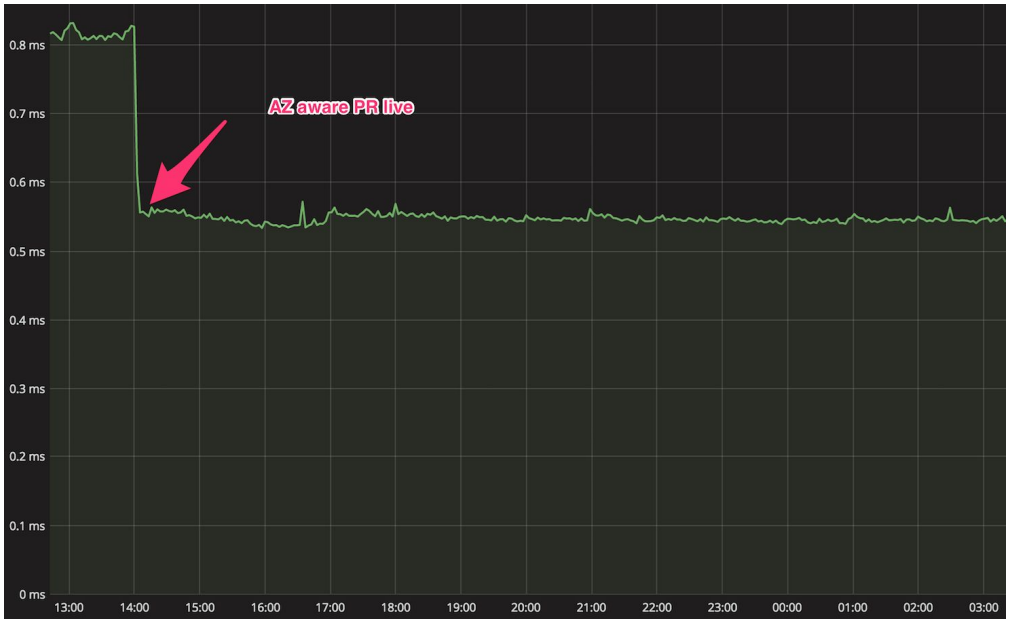
- A new vtgate node is provisioned
- The node register itself to Consul and to the vtgate service
- Each node subscribed to the vtgate service get notified by the change and the service list is refreshed on disk
- PHP cache the new service information in APC

Breakout discovery (AWS)

AZ affinity

Breakout discovery (AWS)

AZ affinity: performance improvements and cost savings using inter-AZ connections



Breakout discovery (MySQL)

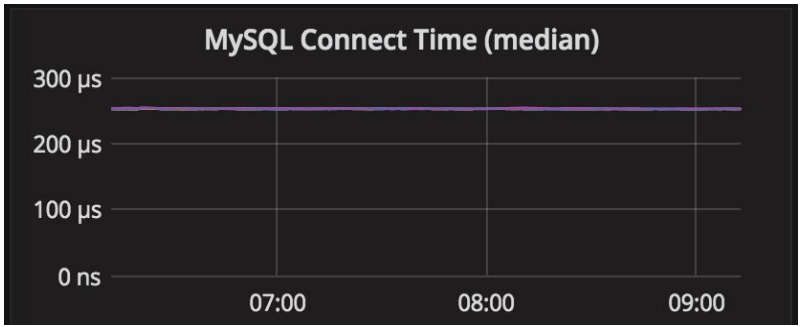
GTID and errant transactions

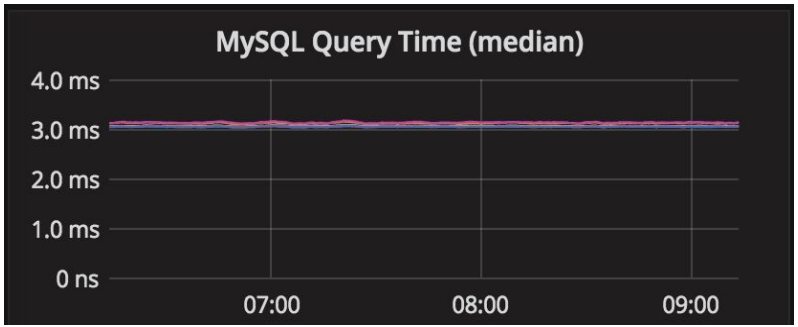
- Breaks failovers automation
- Add a monitoring check for that
- `SET GLOBAL SQL_SLAVE_SKIP_COUNTER = n` is not your friend (anymore)

Breakout discovery (Vitess)

Breakout discovery (Vitess)

Stable performances: connect time ~250µs, query time 3ms (p50)

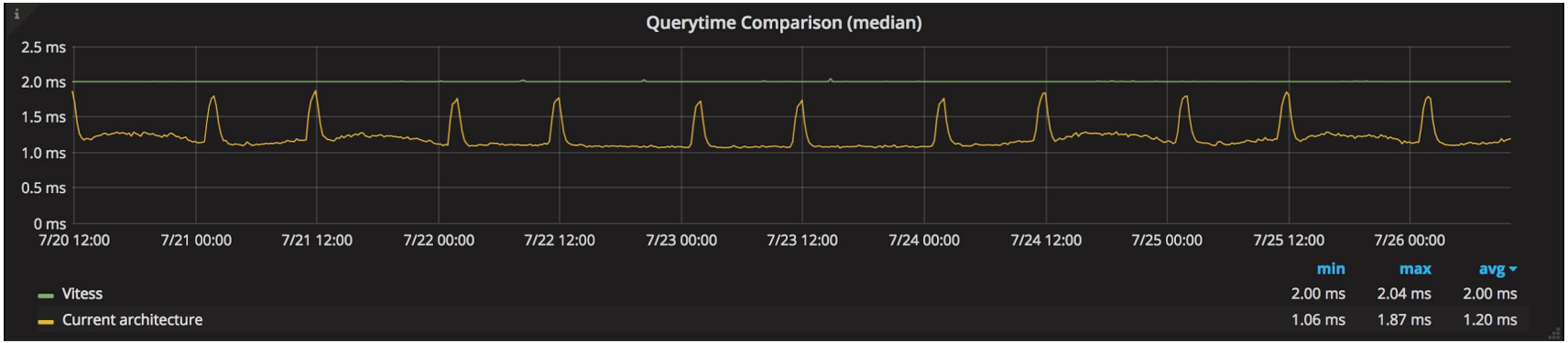




Why? If the keyspace is properly sharded and there aren't hotspots we should only see network latency: 4 hops on write ops (+2 disk flushes), 2 hops on read ops

Breakout discovery (Vitess)

It's slower (but more stable) than our legacy architecture



Why? Network latency: each read operation requires an additional network hop while write operations requires 2 more (due to the semi-sync replication)

Breakout discovery (Vitess)

- Vitess is a not (yet) apt-get install vitess (but we are working on it!)
- Vitess performances (with semi-sync enabled) depends a lot on the network quality
- Vitess has an awesome open source community, we are here to help!
- Vitess is growing fast and getting traction: it’s now an official Cloud Native Computing Foundation 

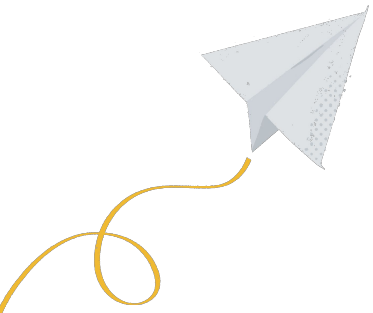
project!

Conclusion

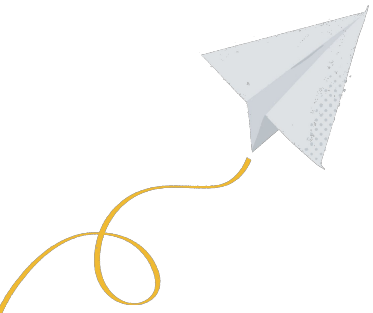
Prepare for the unexpected

during project planning always account time for

interruptions, on-call shifts, holidays and last minute changes



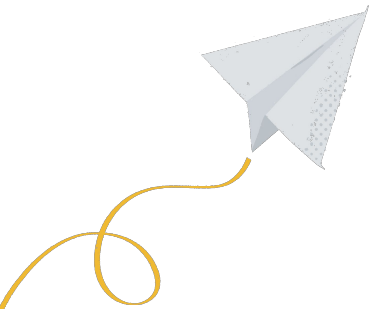
Be a good engineer?



Be a good engineer?

never change too many things at once

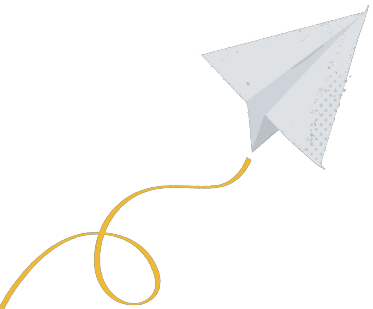
but if you do that, don’t be scared!

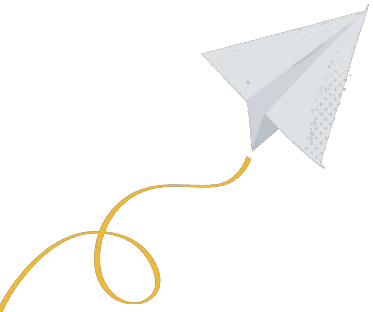


Commitment

launching a new infrastructure

is only the first step











slack

Thank You!

Q&A