GitLab GCP Migration Project

PLEASE NOTE THIS DOCUMENT IS INTENDED TO BE PUBLIC. PLEASE DO NOT ADD ANY CONFIDENTIAL INFORMATION TO IT

Other GCP Related Documents

- Diagram: Cloud Native Pods - Pablo
- Diagram: High-Level Architecture - Pablo
- Google Cloud Platform (GCP) and GitLab - Eliran
- Google and GitLab - Sid
- GitLab.com Environment, Deployment, and Migration tooling - Eric
- GCP Security Guidelines - Kathy
- GCP Migration Project Plan - Eric
- GCP Migration Folder on Google Drive - Andrew
- GitLab Geo GA Plan - Stan
- GCE migration ideas - Eric
- CI/CD GCP Migration - Kamil
- Product Marketing Draft: Reasons for Selecting GCP - Eric & William

Goals of the GCP Migration Project

In order of descending priority. Most important goals at the top.

1. **Use the opportunity of an intercloud migration to make GitLab.com suitable for mission critical client workloads**
2. **Migrate GitLab.com from the Microsoft Azure Cloud platform to the Google Cloud** while keeping downtime to a minimum
3. **Use the same helm charts for GitLab.com as our EEP customers use**
    a. The goal here is for customers to be able to spin up a 10 person GitLab EEP instance in Kubernetes and scale it up to 100k users (or more) with little effort.
4. **Use the migration as a marketing opportunity** for GitLab Inc through creation of technical content

Note: priority here is important. In the event of a lower priority goal leads to major setbacks to higher priority items, then the lower priority goal should be reconsidered first.

Executive Summary of Plan

Migration Gitlab.com from Azure to Google Cloud Platform *and* update the GitLab.com architecture to be cloud native.

A summary of changes that this will require:

1. **GitLab.com provisioned using helm charts on GKE and Terraform**. This will replace Omnibus for GitLab.com. => Terraform vs. GKE? Terraform makes postgreSQL server and GKE cluster. Helm charts will be used whenever possible. PostgreSQL HA and PGbouncer will be managed with Chef.
2. Develop a **continuous delivery process** and build a deployment tool for Kubernetes and Docker, probably based on our CI/CD tooling.
3. **GitLab.com running in Kubernetes** instead of processes running on VMs.
4. **Application changes to remove all dependencies on shared volumes** in EEP. All uploads, LFS objects, build logs, will need to be sent direct to object storage without being written to shared volumes first. Sid => Move everything non-git and GitLab pages to object storage.
5. **Other application changes**
    - GitLab Pages' *reliance on shared-filesystem removed* so that Pages can remain on Azure after the DNS flip. GitLab Pages will migrate in the weeks after the main migration. Sid => proof of concept to move the rebuild to Redis instead of communicating via files to say what it done, Pages will still have a local disk, suppose we have multiple stateless pages servers it might be a shared volume, or mount one volume on multiple machines with one reader
    - CDN / Asset pipeline changes
        i. Sid => We now use sticky sessions, in k8s that is harder, there is a gem that publishes all static assets, put that on a CDN. Jason => We're designing the charts with NGINX as the ingress/controller,

which supports session affinity. Not that it will affect CDN for static assets.

The major risks in this project are likely to be:

1. Relative **lack of experience** within the team in regards to the new stack.
    a. Mitigations:
        i. Kubernetes consultants
            1. ReactiveOps: 2 people x 3 months. Commercial details in private document.
            2. Possibility of ongoing second-level support engagement after the consulting period.
        ii. Game-days: https://blog.newrelic.com/2016/09/30/game-day-testing/ => failure scenario's
        iii. Run registry.gitlab.com on GKE + GCP as early as possible
        iv. Regular functional demos
        v. Jarv: incremental deliverables
        vi. Sid: Load testing? => part of the plan

2. **Unforeseen issues post-migration**: experience with Gitaly has shown that components with the GitLab application interact in unexpected ways. Without adequate manual and automated testing, our users may find unexpected issues once switchover occurs. In the worst case, switchback would be expensive in terms of time, site downtime and reputation.
    a. Mitigations:
        i. Continued work to improve GitLab-QA and automated testing of new environments with GitLab-QA
        ii. Jarv: Static pre-production environment with continuous QA, dashboards and alerting
        iii. One-time manual QA
        iv. Public Preview - allow public access to read-only secondary for a period. For example https://gitlab-gcp.com
            1. Jarv: I'm interested in the details of this, would we be making gitaly calls from GCP to azure over the public internet?
            2. Jarv: as a thought exercise, would it be possible to have a secure connection between clouds and even have a write version of the site with added latency for database queries? I could even see us using a single storage shard in GCP that is only used for this.
        v. Jarv: Rollback plan for the migration?
        vi. Sid: OpenTracing?

3. **Data loss and security breaches.**
    a. With a good backup policy, chances of an outright loss of data is unlikely, but an undetected partial data loss (for example, *some lfs objects* or *git blobs*) could go undetected for a period before being discovered. Security attacks

could target an incorrectly secured GCP environment, the data in-transit between Azure or GitLab, or decommissioned data left behind in Azure.

b. Mitigations:
    i. Store the final backups of Azure dataset permanently, or at least for an extended period
        1. Jarv: formalize the tear-down procedure for azure resources
    ii. Consider building tools to verify (eg using checksums) that Azure and GCP data is equivalent
    iii. During the migration, the backup and restore procedures should be practiced frequently
    iv. Full security audit and penetration testing.

4. **Critical Path Tasks** pushing back the project delivery dates
    a. If any of the following tasks take longer than expected, the project delivery date will be pushed back:
        i. **Cloud Native-compatible GitLab application changes**: the plan is based on this work kicking off on 8 Dec for GitLab 10.3. Any push back on start or delivery date will impact Helm Chart delivery as well as this project.
        ii. **Final Load Testing**, **QA** and **Penetration Testing**: once the environment, application changes and replication are completed.
    b. Although the following tasks are not currently on the critical path, serious delays in their delivery could impact the project
        i. **Geo GA**
        ii. **Gitaly v1.0**

Initial Milestones

See Monthly Milestones section for more details. We'll use the first two milestones as a measure of velocity of the team and feed this back into our project delivery date estimates as we progress.

December 15 Demo Day

https://gitlab.com/gitlab-com/migration/issues/47
What to expect: first iteration pets for postgres, redis, vault. Prometheus with some scraping. Helm charts will generate a docker-mega container, plus unicorn with workhorse charts. K8s pods will use the pets generated by the environment team.

January 15 Demo Day

What to expect: environment setup complete, progress on the cloud native helm charts
https://gitlab.com/gitlab-com/migration/issues/48

Delivery Date

The SSOT of work for this project is the GCP Migration tracker, and the issues and milestones contained there. However, as a secondary, project planning tool, and as a way to model dependencies between the various tasks, OmniPlan has been used for project-planning purposes. Each task has been given an expected duration, as well as minimum duration and maximum duration, and this has been used for monte-carlo simulations to predict a possible date for the switchover to GCP.

A PDF version of the latest copy of the plan is always available at: https://drive.google.com/open?id=1xtOU9nrbt8H_-4sn8tTMVvdNI5OjJQvA

Date of GCP Switchover

This project still has many unknowns. I've added an additional month of contingency onto the plans to reflect this.

Scenario	**Calculated Date**	**Add One Month Contingency**
Best Case (0% likelihood)	2 April 2018	Early May 2018
Expected (75% likelihood)	12 May 2018	Mid June 2018
Worst Case (100% likelihood)	1 June 2018	Early July 2018

Note that this date for the switchover. The final date will include decommissioning and cleanup work on Azure.

Subsequent Iterations

Switchover to GCP is a major goal of the project, but subsequent iterations will focus on decommissioning of the Azure infrastructure and further improvements to the new environment to further improve stability.

1. Complete the move of GitLab Pages
2. Decommissioning of Azure environment
3. Enhanced Monitoring & Distributed Tracing
4. Auto-scaling of various components in GCP
5. Migration of object stores from S3 to GCP
6. GCP Cost Optimisations
7. Multi-region Geo Replication of GitLab.com post-Migration
8. Web Application Firewall: https://gitlab.com/gitlab-com/infrastructure/issues/3511

First Month

Once the project commences, these will be the initial tasks we will focus on:

1. **Architecture Definition**: Pablo: https://gitlab.com/gitlab-com/migration/issues/18
2. **Application Visibility & Alerting:** GCP Environment Setup Task Force: https://gitlab.com/gitlab-com/migration/issues/24
3. **Network Security: Virtual Private Cloud and VPN Access**: GCP Environment Setup Task Force: https://gitlab.com/gitlab-com/migration/issues/27
4. **Required Application Changes**: Backend Development Teams: https://gitlab.com/gitlab-com/migration/issues/23
5. **Migrate Gitlab.com to hashed storage**: Production Team: https://gitlab.com/gitlab-com/infrastructure/issues/2821
6. **Migrate LFS objects to Object Storage**: Production Team: https://gitlab.com/gitlab-com/infrastructure/issues/3233
7. **Migration of CI Artifacts on Gitlab.com to Object Storage:** Production Team
8. **Helm Chart Development:** Build Team https://gitlab.com/charts/helm.gitlab.io
9. **Environment Automation**: GCP Environment Automation & Deployment Task Force https://gitlab.com/gitlab-com/migration/issues/21
10. **Geo Secondary Environment in GCP Setup**: GCP Geo Migration Task Force https://gitlab.com/gitlab-com/migration/issues/20

Note: full project plan is available at https://drive.google.com/open?id=1xtOU9nrbt8H_-4sn8tTMVvdNI5OjJQvA (best downloaded and viewed locally)

Scope

List of items that are either in scope or out of scope for this project

Existing object storage in non-GCP stores (S3, Azure, DO?)	Out-of-scope
DNS - do we intend to move from AWS via Route53 to GCP, or is that out of scope for the migration?	Out-of-scope
GitLab **Pages**	In-scope, possibly over longer timeframes
GitLab **CI runners** https://gitlab.com/gitlab-com/migration/issues/32#note_47038777	Out-of-scope
File system to object storage migrations	In-scope

GitLab **Registry**	In-scope
GitLab **Git Data and Wiki Data (120TB+)**	In-scope
Postgres Data	In-scope
Attachment Data	In-scope
GitLab **CI Artifact Data**	In-scope
GitLab **CI Build Logs**	In-scope
GitLab **Persistent Redis** Instance	TBD
GitLab **Caching Redis** Instance	TBD
GitLab **CI Runners Cache**	Out-of-scope - connected with runners managers (Kamil)
Mattermost Instance	Out-of-scope
Elasticsearch Repository Search data	Out-of-scope
Elasticsearch ELK Logging data	Out-of-scope
Prometheus metric data	Out-of-scope
Profiling currently profiling uses shared volumes	Out-of-scope
Everything else. Any architectural, project ownership or other changes not directly related to the above goals	Out-of-scope
Auto-scaling of web/api/git workers	Out-of-scope
Mailroom	In-scope
What else?	

Assumptions

- Geo GA will be complete and able to replicate GitLab.com intercloud.
- Gitaly will have reached v1 - no need for NFS
- Helm Charts Exist for EE (no need for CE...)
    - In fact, EEP Helm Charts only
- All required application changes can be delivered in time

    - At this point, the GitLab application no longer requires shared file systems in order to operate.
- GitLab-Shell will be broken into separate components to assist with cloud native effort. https://gitlab.com/gitlab-org/gitaly/issues/713
- ...

Risks

List any risks associated with this project. Once this is complete, we can move relevant issues to the risk assessment spreadsheet.

1. Data Loss
    a. Risk:
        i. During the migration, we irrecoverably lose database, git content or other files
    b. Mitigation:
        i. Prior to Geo DR, perform backups, validate the restore process
        ii. Geo features to verify the correctness of
2. Data-corruption
    a. Worse case, the problem is only discovered late, making reconciliation of updated data and original uncorrupted data difficult
3. Security breaches caused by insecure new environment
4. Security breaches caused by insecure VPN between Azure & GCP
5. Insufficient bandwidth to maintain intercloud replication
6. Unplanned outages on GitLab.com
7. Lack of experience working in Google Cloud Platform & GKE
8. Lack of a support contract with Google Cloud (TBD?)
9. Lack of experience running a large Kubernetes instance
10. Issues related to architectural changes in the app in GCP vs Azure
11. Reputational damage from all of the above
12. Geo not ready
    a. Mitigation: plan would require extensive rework
13. Gitaly not ready
    a. Mitigation: operations still require NFS, so k8s nodes will need to be configured as nfs clients
14. More risks here please...

Roles & Responsibilities

Personas involved in this project, in no particular order

- **Director Infrastructure** / **VP Engineering** / **CTO** / **CEO**
    - Responsibilities:
        - Provide oversight, governance and direction for the project

        - Scheduling decisions when there is a conflict between areas of the business (ie, do we build a new CI feature, or modify an existing feature so that it will help with the cloud native effort)

- **GCP Migration Project Manager**
    - Responsibilities
        - Overall responsibility for the GCP Migration
        - Responsible for facilitating communication between other stakeholders
        - Responsible for maintaining the project backlog, scope & documentation
        - ...

- **Kubernetes Consultancy**
    - Responsibilities:
        - Work with production engineer responsible for architecture to validate the target architecture
        - Provide insight into GCP and GKE from past experience
        - Provide second level support for engineering team during and possibly after the project.
        - ...

- **GCP Solutions Architect**
    - Responsibilities:
        - Answer technical questions on GCP
        - Provide feedback on the architecture
        - ...

- **Geo Lead / Geo Project Manager**
    - Responsibilities (draft)
        - At the appropriate time, provide feedback on approximate duration for the intercloud geo migration
        - Go/no-go of Geo as the replication mechanism for the migration.
            - Milestone for this go-ahead TBD
        - Provide input on what data will not be transferred via Geo and what will need extra work
        - Work with the Production Engineer responsible for Intercloud Geo Replication to assist with issues during the replication process
        - ...

- **Production Engineer responsible for Architecture**
    - Responsibilities:
        - Design the ideal & backup architectures for post-GCP migration environments
        - Maintain a set of architecture diagrams which represent the SSOT for our target architecture.

        - Ensure that architecture takes potential pitfalls in the GitLab setup into account
        - Works with security stakeholders for security approval on architecture
        - Works with the build team to ensure architecture is possible in GKE, Helm, GCP etc
        - Accepts feedback from the build team, QA team and works to iteratively improve the architecture based on this feedback
        - Weigh up pros and cons of using Google Cloud Managed services vs running our own.
            - Examples: Google LB vs HAProxy
            - GKE vs own Kubernetes cluster
            - Cloud SQL for Postgres vs own
            - etc

- **Build Lead**
    - Responsibilities (draft):
        - Leads the development of GitLab EEP Helm Charts for use on GKE
        - Works with architecture to ensure that the architecture is converted into helm charts and that limitations, issues are fed back into the architecture.
        - Works with environment automation engineer to ensure that helm charts are executing correctly to create new environments
        - ….

- **Gitaly Lead**
    - Responsibilities (draft):
        - Provide feedback on the status of Gitaly
        - Work with the build team to ensure that Gitaly works well in a cloud native environment
        - Go/no-go of no-NFS git access via Gitaly
            - Milestone: TBD

- **Production Engineer responsible for Intercloud Geo Replication**
    - Responsibilities (draft):
        - Runs the Geo GitLab.com Intercloud Migration workstream
        - Work with environment automation engineer to ensure that snapshot copies of the replication geo data are available for testing purposes in other environments
        - Work with security lead and production engineer responsible for network security to ensure security of data-in-transit and geo replications environment in GCP
        - Work with Geo engineers to set up Geo testbed environments
        - Work with Geo engineers to resolve any issues during the Geo replication process to GCP
        - Work with Geo team to understand Geo DR failover process, when it is ready

        - …

- **Production Engineers Leading the Various Workstreams**
    - Who: John Jarvis, Alex, Ilya, John Northrup, Jason Tevnan, Victor Lopez, Daniele
    - Responsibilities:
        - Work to fully develop a plan for your workstream
        - Raise any issues, concerns or reasons why your workstream could be significantly delayed with the GCP Migration Project Manager
        - Create issues to represent the tasks in the plan
        - Create demo scorecards for demo days for your workstream
        - Work with Security Director & Lead if necessary to achieve security sign-off on the plan and implementation
        - If your workstream will lead to changes in processes, work with relevant teams to agree on changes before implementing the changes
            - For example, continuous delivery will change the way GitLab runs releases. This needs to be discussed, agreed, documented and well understood by product managers, leads and development teams before the change is rolled out.

- **Director of Security** / **Security Lead**
    - Responsibilities (draft):
        - Provide a list of sign-off milestones for the project, for example:
            - Ensure security of data-in-transit
            - Ensure security of environments
        - Application Security Review and penetration testing of new environments
        - Rate-limit setup of new environments?
        - Liaise with Google Cloud Platform on security issues

- **Director of Quality** / **Edge Lead**
    - Responsibilities (draft):
        - Lead the development of automated QA tools to testing of new environments (gitlab-qa)
        - Provide feedback to architecture, helm and environment automation teams on QA issues
        - Lead the development of load testing tools to ensure that new environment will scale correctly
        - Sign-off prior to switchover that replicated data are correct and all accounted for (?)
        - ….

- **Product Managers**
    - Responsibilities:

        - Work with GCP Migration Project Manager in scheduling the application changes needed for the migration
            - Most of these changes are related to cloud native work, specifically work to remove shared file system components from the GitLab application.
            - See https://gitlab.com/gitlab-com/migration/issues/23

- **Director of Strategic Partnerships**
    - Responsibilities (draft)
        - Maintain relationship with Google
        - ….

- **Content Marketing Manager**
    - (draft) Work with GCP Migration Project Manager on generating technical content about GitLab.com's migration to GCP.
        - Great Industry Examples:
            - Sentry: https://cloudplatform.googleblog.com/2017/10/looking-back-on-our-migration-from-bare-metal-to-GCP-Sentry.html
            - ServerDensity: https://medium.com/@davidmytton/the-hidden-costs-of-cloud-ddb702495e93
            - Spotify: https://labs.spotify.com/2016/03/03/spotifys-event-delivery-the-road-to-the-cloud-part-ii/
            - GitHub: https://githubengineering.com/kubernetes-at-github/

Project Stages & Major Milestones

1. **Planning (now!)**
    a. Define milestones and dates
    b. Define workstreams
    c. Define goals
2. **MILESTONE: Kick-off: 22 Nov 2017**
3. **Preparation**
    a. Get everything ready (main preparation workstreams follow in next section)
    b. **MILESTONE: Environment Setup Iteration 1**
    First candidate GCP environment (vpns, metrics, logging) ready
    Date Range: 7 Dec 2017 - 13 Dec 2017

    c. **MILESTONE: Environment Setup Iteration 2**
    Second candidate GCP environment (vpns, metrics, logging) ready
    Date Range: 15 Dec 2017 - 23 Dec 2017

d. **MILESTONE: Registry Rollout Complete**
registry.gitlab.com running on GKE
Date Range: 15 Dec 2017 - 23 Dec 2017

e. **MILESTONE: Environment Setup Complete**
Backup+restore, application visiblity, VPC, VPN connections, etc ready.
Date Range: 25 Dec 2017 - 03 Jan 2018

4. **MILESTONE: Preflight**
At this milestone, we have working secondary replica of the gitlab.com running in the GCP, using the new architecture. We now need to test that it is *correct*, *scalable* and *secure*.

Date Range: 15 Mar 2018 - 16 May 2018

5. **Preflight Testing**
    a. Verification of Migrated Data
    b. Load Testing
        i. Use this data to tune the auto-scaling configuration for environments
    c. Penetration Testing
    d. Using environment automation, quickly spin up a new environment
6. **MILESTONE: Preflight Complete**
All tests complete. Ready for switchover
7. **Switchover**
    a. Perform a Geo DR operation
    b. Main Switchover Process
        i. Planned outage
        ii. DNS Switchover
        iii. Promotion of GCP instance to primary
        iv. Azure instance to secondary (or shutdown)
        v. Outage over
    c. MILESTONE: Main Switchover Complete
    Gitlab.com is running on GCP, some auxiliary properties are still running on GCP. *Important. At this point, we will remove the ability to perform new deploys onto Azure. What remains on Azure will need to be migrated if it needs to be upgraded.*
    d. Secondary Switchover Processes
        i. Pages and any other remaining properties
8. **MILESTONE: Switchover Complete**
9. **Decommission**
    a. Shutdown resources in azure?
    b. Possibility to run as a secondary for a while?
10. **MILESTONE: GCP Migration Project Complete**

Monthly Milestones

Search the Demo label for planned demos.

8 December 2017 - 10.3 Kickoff

- Kickoff for GitLab Cloud Native Changes: https://gitlab.com/gitlab-com/migration/issues/23
- Progress on Helm Charts for Unicorn + Workhorse:
- Architecture

December 15 Demo Day

https://gitlab.com/gitlab-com/migration/issues/47
What to expect: first iteration pets for postgres, redis, vault. Prometheus with some scraping. Helm charts will generate a docker-mega container, plus unicorn with workhorse charts. K8s pods will use the pets generated by the environment team.

January 15 Demo Day

What to expect: environment setup complete, still need to discuss helm chart deliverables with Marin
https://gitlab.com/gitlab-com/migration/issues/48

Teams & Task Forces

The following teams will be involved in this project:

Cloud Native Build Team

Responsible for building out the helm charts that will be used in GKE by both GitLab.com and GitLab EEP customers.

- Marin - Lead
- DJ Mountney
- Jason Plum

Cloud Native App Developers

Application Development Team tasked with Delivering Changes to Make GitLab Cloud-Native compatible.

(Team yet to be decided)

Security Team

Responsible for security signoff of architecture, secure networking, VPNs, etc. Also responsible for pre-go-live penetration testing.

- Kathy - Director
- Brian - Lead

Production Team

Some tasks will be assigned to the production team as a group. Other tasks will be divided between the following three task forces:

Geo Migration Task Force

The Geo Migration Task Force will be responsible for performing a “lift-and-shift” migration from Azure to Google Cloud. Both ends of this migration will use Omnibus Provisioning, to reduce complexity in the lift-and-shift. Once we have a GCP secondary replica on the GCP end, we’ll use this as a source for snapshots with which to generate never, non-Omnibus environments.

- John Jarvis
- Alex

Environment Setup Task Force

This team will be responsible for provisioning & configuring all the infrastructure in a new environment that is not the GitLab application. This includes the “pets”: Postgres, Vault and Redis, as well as other components such as Prometheus, Consul, etc

- Ilya
- John Northrup
- Jason Tevnan
- Victor Lopez

Environment Automation & Deployment Task Force

This team will be responsible for automating the provision of new environments by combining the GitLab helm charts created by the Cloud Native Build Team with the infrastructure components (as both helm charts and chef scripts) created by the Environment Setup Task force. Ideally, the creation of new environments should be automated and driven from changes in the Environment Setup and Helm Charts repositories. After spinning up a new environment it should be automatically tested with GitLab QA.

Additionally, this team will lead the continuous delivery workstream, working closely with product and engineering teams to deliver a continuous delivery process that is well

understood by those who will be using it, and from that process, a set of continuous delivery tools, based on GitLab CI/CD if possible.

- Pablo
- Daniele

QA Team

Responsible for leading the development of GitLab QA which will be used to test candidate environments. Also responsible for engaging with consultants on Load Testing. The QA team will perform a final QA signoff on the candidate environment prior to switchover.

- BJ (Director)
- Rémy (Lead)

Preparation Stage Workstreams

GCP Migration will require coordination of multiple streams of work, particularly if we the cloud native re-architecture of GitLab.com is to remain a deliverable.

The major streams of work are listed here: https://gitlab.com/gitlab-com/migration/issues?label_name%5B%5D=Workstream. They are:

Architecture & Planning Workstreams

Owned by: **GCP New Architecture Lead (Pablo Carranza)**

Step 1: come up with a plan!

1. **Architecture Definition**
    - Define ideal cloud-native GitLab architecture
    - Perform an audit of this architecture, listing all required application changes that will need to take place in order to achieve this ideal architecture, around, for example, removing shared file-system access. These changes are listed in #23
    - Define backup "legacy" (VM-based) architecture in the event that the required application changes cannot be scheduled within a reasonable time-frame

Environment Setup Workstreams

Owned by: **GCP Environment Setup Task Force**

These workstreams are mostly only dependent on the architecture being delivered in step 1 above. Once this has been defined, most of these tasks can be completed without waiting for upstream blockers.

2. **Application Visibility & Alerting**
    - Assigned to: Victor Lopez
    - Create a combination of chef-scripts (for VM components) and helm-charts (for k8s components) to provision application visibility & alerting metrics
    - Ensure that all existing metrics, logging and alerting systems are migrated to GCP or replaced with GCP equivalents
3. **Network Security: Virtual Private Cloud and VPN Access**
    - Assigned to: TBD
    - Allow GitLab operators to access the VPC network (or networks) via a VPN connection.
    - Ideally rely on Google Cloud authentication methods
    - Ideally segment VPN access to different projects, so that access to staging or dev can be differentiated from production.
4. **Registry Rollout: Working with Kubernetes in Production**
    - Assigned to: John Northrup
    - Gain actual production experience with Google Cloud, Kubernetes, GKE and many more technologies that many of the team currently lack production experience using by migrating the GitLab.com container registry to GKE.
5. **Backup & Restore**
    - Assigned to: Alex?
    - Plan and architect a backup procedure for the new environment which matches the existing backup policy at https://about.gitlab.com/handbook/infrastructure/production/#summary-of-backup-strategy
6. **Vault and Secrets**
    - Assigned to: Ilya?
    - Ensure that vault + secret data can be provisioned in a new environment using Chef.
7. **Postgres + HA + PG Bouncer + Consul**
    - Assigned to: Jason Tevnan
    - Ensure postgres HA, PG Bouncer and consul can be provisioned together in a new environment
8. **HAProxy**
    - Assigned To: John Northrup
    - Provision HAProxy in Kubernetes as pods using Helm for rate limiting and other purposes
9. **Redis**
    - Assigned to: TBD. Daniele?
    - Using Chef, provision two Redis Primary + Secondary + Sentinel instances for a new environment, one for persist data, second one for caching data

Application Development Workstreams

Owned by: **Various Backend Teams, Product**

These work streams require application development to be carried out and will need to be scheduled in concert with Product Managers from various teams.

10. **Geo GA**
    - Perform the full GCP Migration using GitLab Geo functionality
11. **Gitaly v1.0**
    - All Git access on GitLab.com through Gitaly, no shared filesystem access
12. **Required Application Changes**
    - Make all required changes to the GitLab rails monolith that will facilitate
        i. The migration to GCP (if any changes are needed)
        ii. The transition to GKE + Kubernetes. Most of these changes are related to removal of shared file-system access.
13. **GitLab Pages**
    - Ensure the stability of GitLab Pages during the migration process and the migration of the gitlab.io property from Azure to GCP. This does not necessarily have to happen within the same timeframes as the rest of the migration.

Deployment & Environment Automation Workstreams

Owned by: **Build Team for Helm Charts working with GCP Environment Automation & Deployment Task Force** “Envs & Deployment”

14. **Required GitLab.com Changes**
    - Assigned to: various production team members per task
    - Carry out required changes to the existing GitLab.com Azure instance prior to migration to GCP
15. **Helm Chart Development**
    - Assigned to: Build Team
    - Starting with the omnibus all-in-one docker image, iteratively break the monolithic container down into multiple pods.
    - Through iterative releases of the charts, progressively move towards the proposed architecture
16. **Environment Automation**
    - Assigned to: Daniele
    - Using the most recent iteration of helm charts produced in “Helm Charts” workstream, automate the generation of new environments within GCP.
    - Initially, these environments would be pristine, but once the full Geo replication is completed, environment generation would need to include data (possibly scrubbed).
17. **GitLab.com Continuous Delivery to GCP**
    - Assigned to: Pablo

    - Define a continuous delivery process for GitLab.com
    - Develop the required infrastructure to implement the continuous delivery process

Migration Workstreams

Owned by: **GCP Geo Migration Task Force**

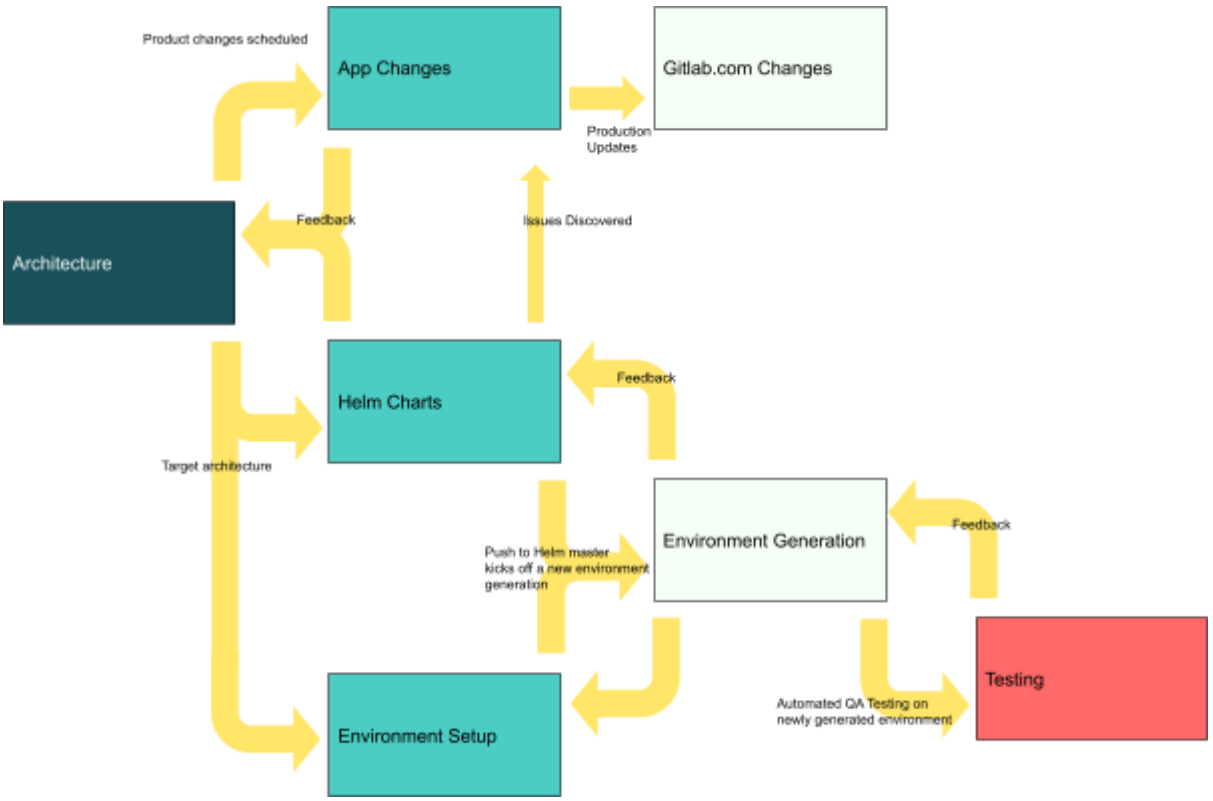
18. **Geo GitLab.com Migration to the Golden Secondary**
    - Assigned to: TBD. John Jarvis?
    - Have a full working secondary replicated from the GitLab.com master running on Azure, to a secondary running on GCP
    - Once the replication is completed, ensure that the secondary stays up-to-date with the master
    - Ensure that the data in the secondary can be used to seed additional environments in GCP

Workstream Interfacing

Some of the streams of work will need to work together in order to deliver their workstreams. This diagram illustrates how the target architecture will drive application changes and downstream production changes, as well as the helm charts, the automated environment generation and downstream testing.

Each of these flows will then feedback upstream for the next iteration.

The goal is to automate this process as such as possible. In particular, changes to the helm charts should automatically drive new environments to be created in GCP which can then be automatically tested using GitLab QA. Warnings and failures in each of these stages will lead to feedback in the previous stage.



Related Projects

- **Geo**
  - Transfer of data from Azure to Google Cloud
  - Owner: Geo Project Manager / Ernst van Nierop
  - https://gitlab.com/groups/gitlab-org/issues?label_name%5B%5D=Geo+GA
  - https://gitlab.com/groups/gitlab-org/issues?label_name%5B%5D=Geo
- **Gitaly**
  - If Gitaly reaches v1.0 in time, the architecture can be built without NFS requirements
  - Owner: Jacob Vosmaer
- **Security Review**
  - Ensure that our new environment is secure
  - Owner: Kathy Wang
- **Environment Load Testing Project (TBD)**
  - Ensure that new environment is ready to receive load
  - Owner: BJ Gopinath

Questions & Answers

Q: How are we going to distribute our cattle and pets between zones in GCP.

- Primary and Secondaries should be split between zones in the same region.

Q: Which GCP region are we going to use? North Virginia / us-east4? Do we have any idea of what the latencies between our current Azure region and our selected GCP region are?

- Both the GCP and the Azure DC are in Ashburn, VA and share many of the same providers for ISP connectivity.

Q: Does Geo have switchover functionality? Can we failover from a primary to a secondary?

- Not yet, this will be part of the “Geo DR” functionality, which will arrive after Geo GA, but before the GCP Switchover.

Q: Does Geo enforce “read-only” mode on the secondary?

- A: Geo secondaries currently display a “read-only” banner. Future versions will also remove writeable input elements from the UI.

Q: Can we deploy to the GCP cluster using GitLab CI/CD?

- Kamil has an example of this? https://gitlab.com/gitlab-examples/kubernetes-deploy
- Jason P modified the AutoDevOps template to work for helm.gitlab.io https://gitlab.com/charts/helm.gitlab.io/blob/master/.gitlab-ci.yml

Q: Do we need to provision a secondary GitLab instance (for example ops.gitlab.net) for deploying the GitLab primary instance?

- Yes, although we’ll probably change the name.

Q: Do we have a definitive list of application changes that need to happen to the rails monolith before we can migrate?

- A: these are listed in https://gitlab.com/gitlab-com/migration/issues/23

Q: Have we considered alternative approaches to migrating to Azure (for example: VM Migration: https://cloud.google.com/migrate/ or a straight “lift-and-shift”)

- A: since the primary goal of this project is “Make GitLab.com suitable for mission critical client workloads” and a VM-to-VM migration would not further this goal, it is not currently being considered.

Q: Consulting firms: can we bring external expertise into the project to assist in some areas?

- A: we will be employing the services of a Kubernetes consultancy (2 consultants for 12 weeks) and possibly also a company to help with load testing of our new environments.

Q: Do we really need consul? It seems like we could get away with using Kubernetes configuration components and skip consul.

- A from Jason Tevnan: “consul is needed for the pg-ha setup and since we made the decision to pursue this course of action in omnibus, it's not something we can easily change. removing it would be extremely hard and time consuming.” https://gitlab.com/gitlab-com/migration/issues/39

Project Process

- **Slack Channel**: #gcp_migration
- **Mailing List** (will also be used for GCP permissions): gcpmigration@gitlab.com
- **Project**: https://gitlab.com/gitlab-com/migration

Demos

This project will need to bring people together from across the company. Let's use frequent, regular scheduled demo sessions to highlight progress, invite discussion and make sure everyone is synced up.

Proposal: every [*Friday*] at [*15h00 UTC*] we schedule a demo presentation, all are welcome to join. We rotate the presentation between different teams, depending on the tasks that have been covered in the past 7 days.

Communications

The project will communicate through the following channels

Daily Slack Standup

If you're working on GCP, please participate in the Slack Daily Standup by updating the team on your progress when you start work for the day.

Please answer the following questions:

> What did you work on yesterday?
> What did you work on today?
> What are you blocked on?

Weekly GitLab GCP Migration Meeting

TODO: Should occur once a week. What is the best slot. 17h00 UTC Tuesdays has been provisionally booked. This can change.

Calendar:
https://calendar.google.com/event?action=TEMPLATE&tmeid=MTAyNHZnNzNvb244NHJ1N29vcDlxNWUzYm1fMjAxNzEyMDFUMTYwMDAwWiBhbmRyZXdAZ2l0bGFiLmNvbQ&tmsrc=andrew%40gitlab.com&scp=ALL

GCP Migration Project: Schedule
https://calendar.google.com/calendar/embed?src=gitlab.com_3o4uhe8prttg8lv71mkqpmoim8%40group.calendar.google.com&ctz=Europe%2FLondon

Agenda:
https://docs.google.com/document/d/1kl-bZAvKFfjQIUW4mcoh3aDrJ7hvZdiRz2C5MNk8Nec/edit

Zoom: https://gitlab.zoom.us/j/508139928

Fortnightly GitLab - Google Cloud Update

Occurs every two weeks on a Wednesday at 17h30 UTC

Calendar:
https://calendar.google.com/event?action=TEMPLATE&tmeid=NTFjMWk4MzUyZGRhcGdnaml1b3FsY2dibmxfMjAxNzExMTVUMTczMDAwWiBhbmRyZXdAZ2l0bGFiLmNvbQ&tmsrc=andrew%40gitlab.com&scp=ALL

Agenda:
https://docs.google.com/document/d/1CcCmbK7JPs8Ni96WcvAMIIRAZe0ESMVPvaSUZgv7Lw8/edit

Zoom: https://gitlab.zoom.us/j/315675476

Executive Update - 18 January 2018

Previous Estimate

Monte carlo worst case July 1.

Current State of Play

- **Summary**
  - Monte carlo worst case August 1
  - Key drivers (in order)
    i. - Scalable Geo not ready (planned Geo secondary sync: 1 Feb, likely March)
    ii. - Pablo leaving (a higher proportion of the team needs to be on-call now)

        iii. - Object storage (Discussion) work arrived a release late, delaying CI/CD handoff
        iv. + ReactiveOps signed and engaged
    - Other lessons learned
        i. Nature of project involves sequential operations and long-running sync processes, leading to challenges in compressing the timeline
        ii. Under-estimation on effort involved in environment automation work
        iii. Over-estimation on production group availability (due to on-call Azure/Intel, etc)
        iv. Ramp-up and rockiness in working in demo-driven process
        v. Ill-defined lines between Andrew and production group

- **ReactiveOps Consultancy Engagement**
    - Terms agreed, 25% discount for comarketing engagements
    - Paul has signed the contract

- **Object Storage**:
    - ***Summary****: first component has slipped behind a little, but the complexity in the work was underestimated. Kamil, Sean, Douwe and PMs have put together a well defined roadmap towards a cloud-native GitLab application.*
    - Sean & Victor volunteered the Discussion group to handle the required changes to GitLab to allow for all upload types to be migrated to object storage
    - https://gitlab.com/gitlab-org/gitlab-ee/issues/4163 + https://gitlab.com/gitlab-org/gitlab-ee/merge_requests/3867
    - Object storage missed the 10.4 cutoff but appears to be on track for 10.5
    - This change gives us the ability to load any upload type (LFS, avatar, attachment, etc) into object storage, but still requires an intermediate shared file-system (prior to move to object store).
    - Next step is direct upload to object storage, without requiring the shared file system: https://gitlab.com/gitlab-org/gitlab-ee/issues/4184 - this requires Workhorse changes in Golang, so most likely will require CI development.
    - Additionally, we need to handle CI traces (aka logs): https://gitlab.com/gitlab-org/gitlab-ee/issues/4607#note_54806686. The streaming nature of “live” traces makes this difficult to work with in an immutable object storage world, but there are plans.
    - Kamil, Fabio and I spent time looking at whether we could throw more people at the CI Object Storage issues. **It’s unlikely that throwing more people at the CI tasks will bring the dates in.**

- **Pages**:
    - ***Summary****: no movement here and have struggled to find the right developer to own this. Jacob is interested in helping post Gitaly v1.0.*
    - Changes need to be scheduled, concern around assigning ownership for required changed (but may fall on CI group)
    - https://gitlab.com/gitlab-com/migration/issues/29

- Andrew: practically, there are three people this task could fall on: Kamil, Nick & Jacob. Kamil is focused on the CI components of the GCP Migration, Nick needs to focus on Geo. Jacob may have some slack after we complete the endpoints work.
- **Jacob is open to doing this work in the March or April timeframe.**
- **=> Sid: make webhook**

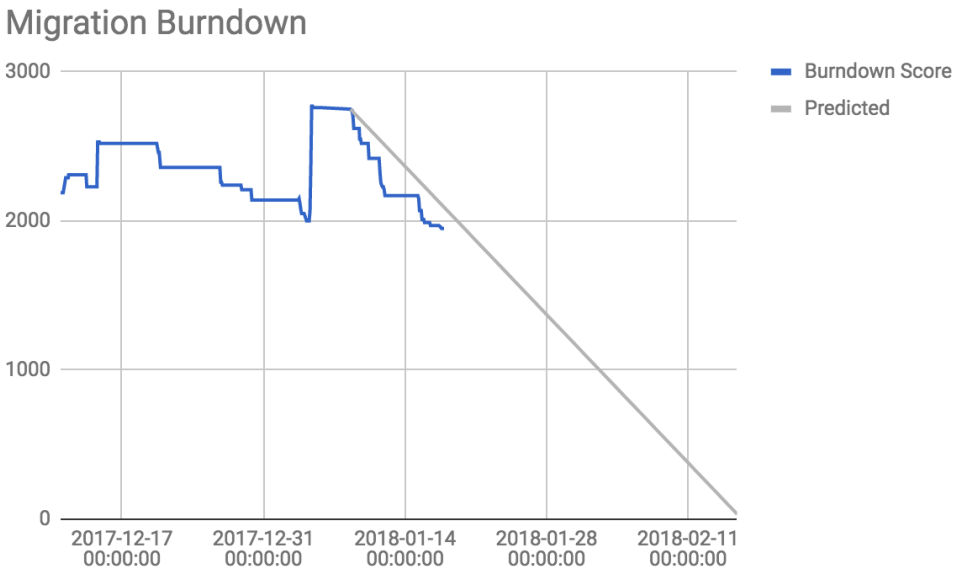
- **Geo**:
  - ***Summary**: having a replicated secondary of GitLab.com in GCP is a high priority for GCP, we should focus on being able to do this as soon as possible.*
  - Geo group priority currently DR
  - GCP Migration priorities for Geo:
    i. Gitlab.com-scale Geo
      - Check frequency of namespace renames / delete / creates on GitLab.com https://gitlab.com/gitlab-org/gitlab-ee/issues/4525
      - [meta] Ensure Geo replication can handle GitLab.com-scale update load https://gitlab.com/gitlab-org/gitlab-ee/issues/4030
      - [meta] Benchmark GitLab Geo https://gitlab.com/gitlab-org/gitlab-ee/issues/3997

    ii. Hashed Storage GA:
      - Production readiness for enabling hashed storage on gitlab.com https://gitlab.com/gitlab-com/infrastructure/issues/3542
      - Transition test suite to use hashed storage: https://gitlab.com/gitlab-org/gitlab-ce/issues/40744
      - Formulate plan to go from "hashed storage tested" to hashed storage GA https://gitlab.com/gitlab-org/gitlab-ce/issues/40970
  - GCP plan originally had “Golden Secondary” replication starting early February: this is currently blocked by Gitlab.com-scale Geo. Lots of planning and discussions around how to move forward to this, but I still feel the problem is open-ended without a fixed delivery date
    i. **GCP Project Concern:** We need date on which we can commence a full replication to the Golden Secondary environment in GCP.
  - Hashed storage not required for initial replication; will be necessary once users access GitLab.com through secondary. Implementation of hashed storage in current form would require rollout and rollback plan, prone to bugs. Group discussing implementing hashed storage that does not require migration of existing projects.

- **Environment Setup Changes**
  - ***Summary**: delivery is behind the November plan*
  - Initial plan had this work completing end of January but this will extend out further.
  - We are behind the initial plan but the plan was probably far too optimistic in this area.
    i. Production group have been able to spend less time than I expected

        ii. Plan assumed that more of the existing assets (chef cookbooks) would be reusable, with tweaks.
    - Concerns around the amount of effort the production group can devote to GCP, due to
        i. On-call rotation
        ii. Loss of Pablo from group.
        iii. Jarv taking on management tasks
        iv. Other small tasks eg BizOps
        v. “The usual” unexpected firefighting (for example, Meltdown/Spectre resulting in little GCP work in first week January)
    - Updates:
        i. Observability: https://gitlab.com/gitlab-com/migration/issues/24
            - ELK Cluster - Victor making good progress on this
            - Prometheus - Victor demonstrated good progress
            - Outstanding Work includes
                - Grafana
                - Alertmanager
                - OAuthProxy
                - Influxdb
            - **NB: Andrew: reminder: Sid, I am awaiting your MR regarding product management involvement.**
        ii. Chef - complete or near complete https://gitlab.com/gitlab-com/migration/issues/55
        iii. Consul and service discovery - some progress, requires demo
        iv. Redis - good progress https://gitlab.com/gitlab-com/migration/issues/44
        v. Postgres - no HA, pgbouncer yet: https://gitlab.com/gitlab-com/migration/issues/42
        vi. Backup - no progress: https://gitlab.com/gitlab-com/migration/issues/25
        vii. Ingress - no demo yet: https://gitlab.com/gitlab-com/migration/issues/43
        viii. Registry - no demo yet: https://gitlab.com/gitlab-com/migration/issues/26

- **Gitaly**
    - ***Summary****: Gitaly will remain off the critical path if current (post-New Year) burndown rate is maintained and Gitaly enters post-migration, AT phase mid-February.*



  - Cloud Native project / Helm Charts require all Gitaly features to be ready (for testing) as soon as possible. This graph represents the progress towards that goal. **On track to complete all endpoints to ready for CN project 15 February**, and focused on delivering Cloud Native project blocker issues as a priority.
  - **Expectation management: testing and production-related fixes will need to be performed after this deadline,** but the cloud native group will be unblocked.
    i. Following endpoint delivery, a period of acceptance testing and fixing bugs. If we can deliver this within 6 weeks, Gitaly will avoid being on the critical path.
  - Gitaly plan: ramp up Gitaly to 100% on Azure, but no effort will be put into decommissioning the NFS on Azure - focus on GCP Migration instead and commissioning in GCP without NFS.
    i. Certification using production workers without volumes https://gitlab.com/gitlab-org/gitaly/issues/932

- **Helm Charts, Cloud Native**
  - ***Summary:*** *making good progress but ultimately they will be blocked by required application changes (Object Storage, Gitaly, Pages)*
  - Cloud Native group have integrated GitLab-QA into the project, for automated testing of helm deployments

- **Testing and QA**
  - ***Summary:*** *GitLab-QA is a good start, but we need much higher coverage, and a solution to load testing.*
  - QA project has fortunately been delivered at the perfect time for GCP Migration (and related) projects, but...
    i. We need integration test coverage to be much higher if we are to rely on it

    ii. We need a way of performing load balancing (Jarv's suggestion: many GitLab-QA in parallel using CI?)
    iii. Expectation management: even if GitLab-QA expanded at a huge rate, it would not on it's own be sufficient testing, particularly if we continue with the big-bang single migration approach

Questions

- Andrew: are there any contractual time-constraints on our move to GCP?
    - Gayatri from Google: "What's the timeline we are shooting for migration, and announcement?"
- Eric: Are we clear to put the Geo GCP work ahead of the DR work?

Changes

Hold and de-risk the Aug date.

- CI/CD:
    - Developers pair on object storage work: https://docs.google.com/spreadsheets/d/1ZhofmMP-zQYEwYH1AZO60VqhkiTOGIyS6srzSq8WJX8/edit
    - Shorten sync process by only syncing recent artifacts
- Pages: Jacob V to work on the pages fix in April
- Geo: meta issue: https://gitlab.com/gitlab-com/migration/issues/19
    - Put GCP work before DR work
    - Primary High-level Goals:
        - Ideally: a timeline, with dates, for Geo at GitLab.com-scale workloads
            - This involves many steps, but from a GCP migration standpoint, prefer to focus on the more general milestone
            - (Required for start of Golden Secondary workstream)
        - Secondary goal: a timeline, with dates, for full migration to hashed storage (required for GCP Switchover)
    - Intermediate goals (*Andrew: from a high-level perspective, these are less important to me than the goal of being able to a) replicate GitLab.com to GCP and b) Enable hashed storage before the switchover*)
        - Estimate DB load, and turn on Geo replication in staging to characterize performance
            - Build stress test: https://gitlab.com/gitlab-org/gitlab-ee/issues/4030
        - Hashed storage rolled out in way that does not require migrating old projects
            - https://gitlab.com/gitlab-org/gitlab-ce/issues/40970#note_54982502
        - Selective sync at gitlab.com scale - https://gitlab.com/gitlab-org/gitlab-ee/issues/4625
    - Production to work on alt backup plan to Geo as migration tool?

- GitLab-QA:
    - Engage with QA group for
        - more GitLab-QA coverage
        - A load-testing solution
- People
    - Andrew engages more closely with Geo and CI/CD leads/managers/groups since they are both critical path and don't have their permanent, full time managers yet
    - Eric works with Stan to make sure that he (Geo) and Kamil (CI/CD) are focused on the highest priority deliverables
- Hiring: Push to hire Phil Kates for the production team (k8s expert) https://www.linkedin.com/in/philkates/