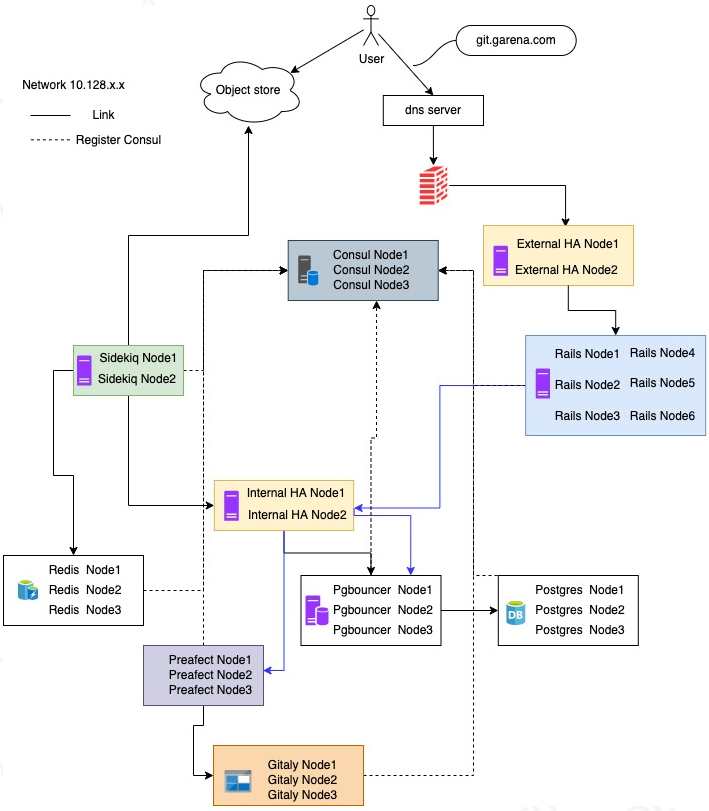
Gitlab 集群现状

当前现状

35台集群搭建的主节点及一个 all in one 的geo节点



风险点

以下情况下会出现整个集群完全不可用

所属方	组件	状况	表现形式	修复手段	预计修复时间（需要验证）
集群内部	External HA	所有HA组件完全Down	用户完全无法访问, 一般报错为链接超时	1. 新建ha 节点并配置相关IP	1. 30min ~ 1h
	Rails	所有rails节点完全down	用户完全无法访问, 一般报错为502	1. 新建rails 节点挂载到ha上	1. 10min
	Internal HA	所有HA组件完全Down	用户完全无法访问, 一般报错为500异常	1. 新建ha 节点并配置相关IP	1. 30min ~ 1h
	Pgbouncer	所有pgbouncer组件完全Down	用户完全无法拉取代码, 一般报错为500异常	1. 新建pgbouncer节点，添加到集群中	1. 30min ~ 2h
	PG	所有PG 故障	用户完全无法拉取代码, 一般报错为500异常	1. 新建pg节点，添加到集群中	1. 6 ~ 12h (需要做数据恢复)
	Praefect	所有Praefect 故障	用户完全无法拉取代码, 一般报错为500异常	1. 新建raefect节点，添加到集群中	1. 30min ~ 1h

	Gitaly	所有Gitaly 故障	用户完全无法拉取代码, 一般报错为500异常	1. 新建gitaly节点，添加到集群中	1. 4d
	Sidekiq	所有Sidekiq 故障	用户合入MR缓慢，CI任务无法进行等	1. 新建sidekiq节点，添加到集群中	1. 30min ~ 1h
	Redis	所有Redis 故障	会影响Sidekiq的正常运行（猜测）	1. 新建redis节点，添加到集群中	1. 2 ~ 6h
外部依赖	DNS	DNS 故障	DNS 故障导致用户无法解析git域名，无法访问	寻求DNSteam的帮助	Case By Case
	S3存储	存储故障	1. 无法运行gitlab CI 2. 无法获取用户头像 3. 无法获取mr diff信息 4. 无法使用page	寻求存储团队的帮助	Case By Case
	机房故障	所有机器无法连接	git完全瘫痪	等待机房恢复	未知

风险点测试与应急响应方案

针对上述不可用的情况，我们目前具体修复步骤与修复时间无法保证，所以需要一套测试环境来模拟各种故障情况，针对故障情况输出相关的SOP文档及快速恢复机制。

由于目前我们的集群现状只有一个单点Geo节点，当线上集群完全不可用，当前Geo节点也无法承接所有线上流量，但可以缓解重要业务线的服务，以避免业务损失，但是相关操作切换需要时间约为2h+（根据上次集群迁移情况），同时这种没有反向同步机制，所以当主集群恢复时，在geo为主的数据，一是重新让客户推送一遍，二是让geo当主，10k 集群挂载为geo，进行数据同步（此种方案可以忽略）；

针对上述情况，我们有两种方案：

1、申请35台机器，搭建一个集群，做geo，当目前10k的主节点挂了，切换到geo集群，当10k恢复后，将10k当作geo挂载到这个geo（10k-backup）集群中；

2、将现有10k集群做成多机房，以免一个机房故障导致完全无法使用，需要调研；

测试环境

针对现状我们需要申请机器来做相关验证（如集群升级，上述故障现象及恢复手册，后续相关代码修改验证测试，自动化能力建设）

1. 申请VM机器，但Gitaly 节点存储需要超过2T，将现在的数据可以导入到其中；
2. 申请物理机，当我们完成测试后，可以当作GEO节点挂载到现有集群中，以防机房故障导致完全不可用；

两种机器的优缺点

VM

优点：配置可以低一些，来满足相关的功能测试，同时可以保留测试环境，方便各种测试；

缺点：无法当作GEO节点挂载到现有集群中；

物理机

优点：可以在测试完成后，挂载到集群中，当作Backup；

缺点：资源严重浪费，机房完全故障的几率较小，个人认为没有必要；