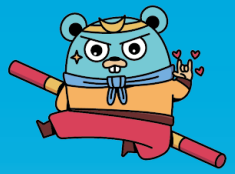


# 知乎社区核心业务 Golang 化实践

知乎 杜旭
xlzd@zhihu.com

探探 Gopher China 2019

# About me

- 杜旭 xlzd@zhihu.com
- 知乎后端工程师
- 知乎主页：https://zhihu.com/xlzd



# 目录

- 重构背景
- 重构成果
- 重构过程
- 一些实践经验



# 背景

# 优化 Python 代码

OR

# 换语言重写



# Why


ZAE
应用
联调
文档
ONCALL
提交 RFC
填写应用信息
3
选择代码模板
选择一个语言
Python
Go
NodeJS
Java
Other
BACK
提交申请

# Why

- 并发优势 & 执行效率
- 公司内部基础组件生态
- 静态类型 & 多人协作
- 学习成本 & 开发效率
- 工程师喜好



# Why

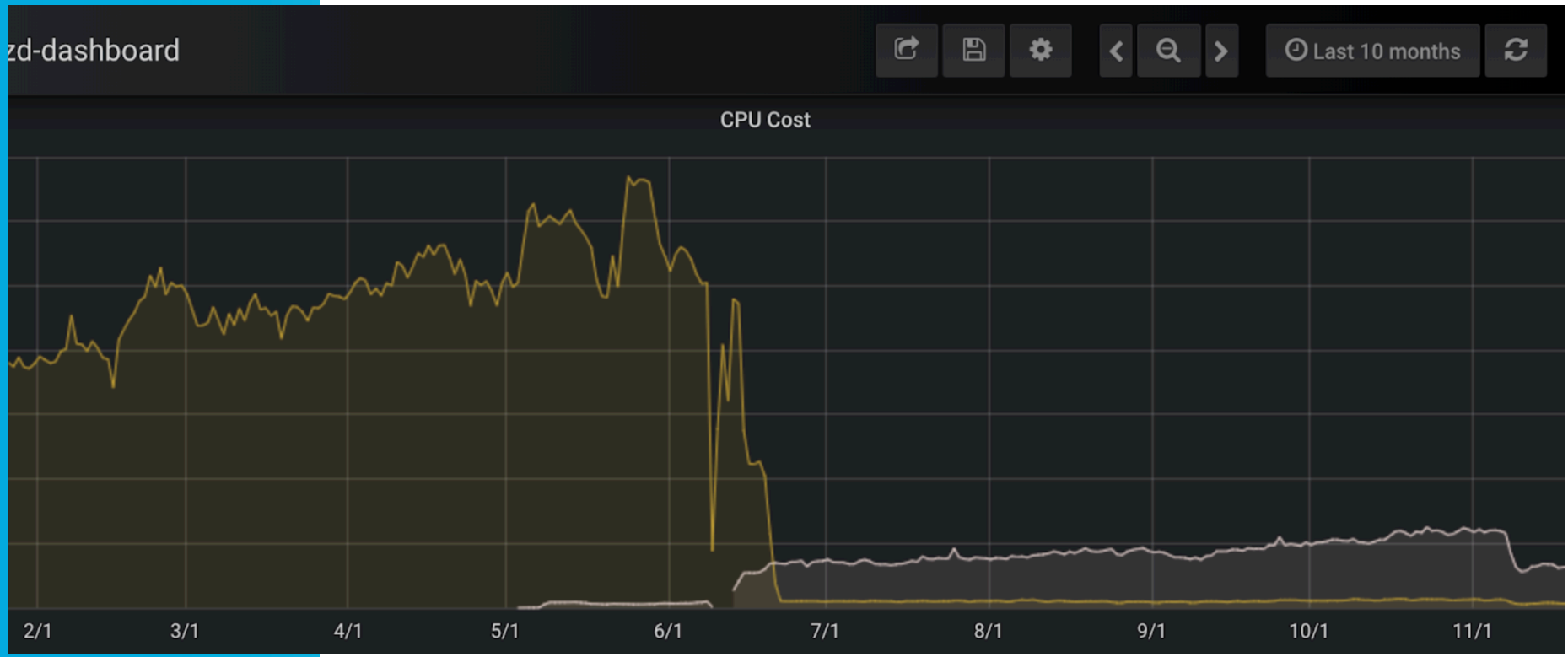
```
package main
```



# 重构成果






zd-dashboard
Last 10 months
CPU Cost
2/1
3/1
4/1
5/1
6/1
7/1
8/1
9/1
10/1
11/1

# 重构成果

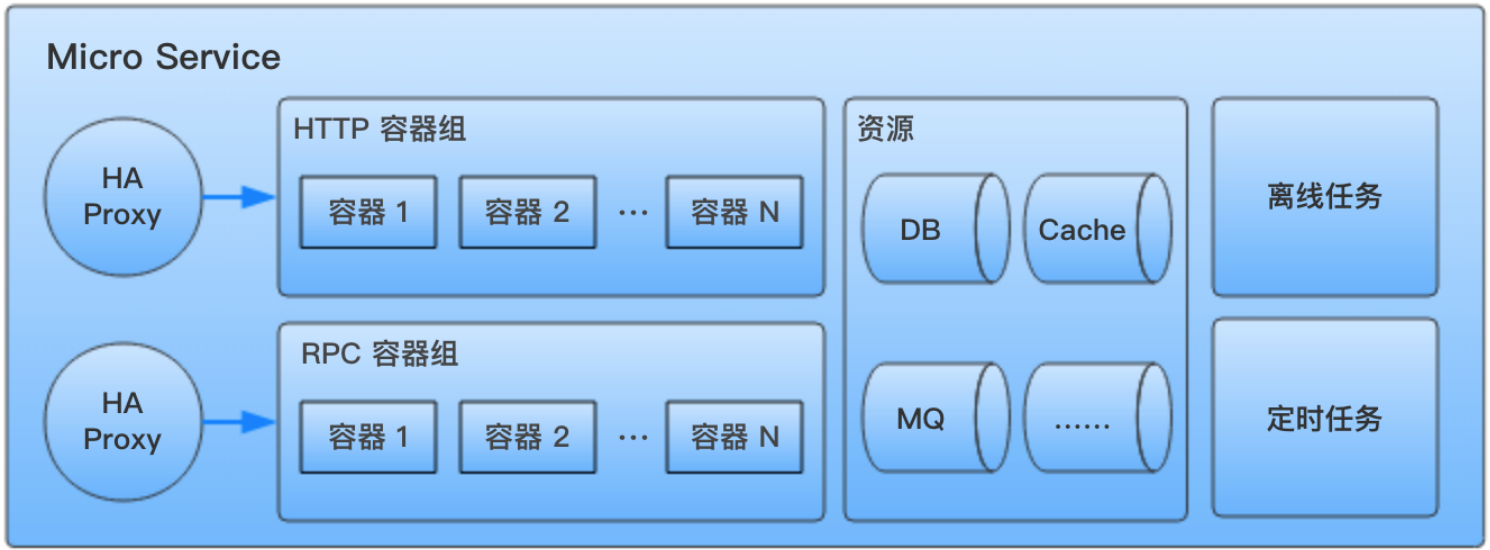
- 节约了超过 80% 服务器资源
- 多人协作开发成本
- 踩坑，Golang 成为内部推荐语言



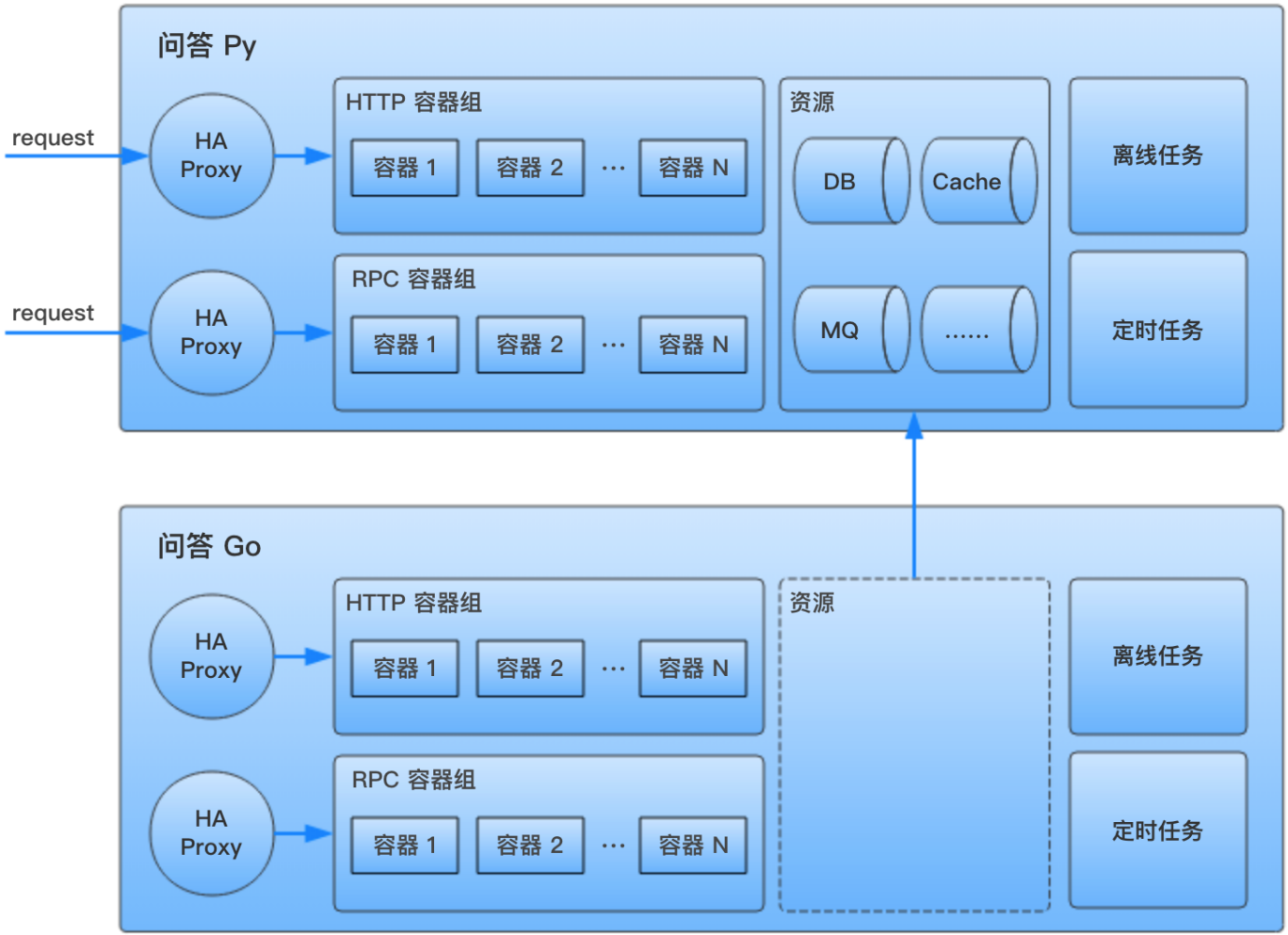
# 重构过程



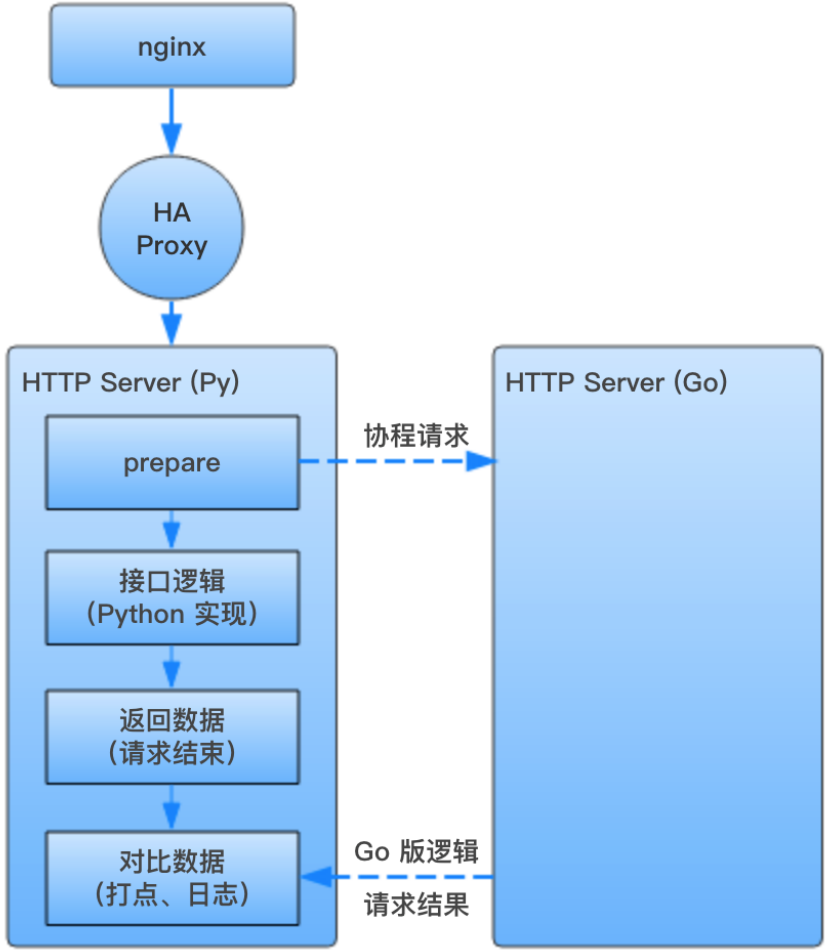
# 重构过程


Micro Service
HA Proxy
HA Proxy
HTTP 容器组
容器 1
容器 2
…
容器 N
RPC 容器组
容器 1
容器 2
…
容器 N
资源
DB
Cache
MQ
……
离线任务
定时任务

# 重构过程


问答 Py
request
HA Proxy
HTTP 容器组
容器 1
容器 2
…
容器 N
资源
DB
Cache
离线任务
request
HA Proxy
RPC 容器组
容器 1
容器 2
…
容器 N
MQ
……
定时任务
问答 Go
HA Proxy
HTTP 容器组
容器 1
容器 2
…
容器 N
资源
离线任务
HA Proxy
RPC 容器组
容器 1
容器 2
…
容器 N
定时任务

# 重构过程


nginx
HA Proxy
HTTP Server (Py)
prepare
接口逻辑 (Python 实现)
返回数据 (请求结束)
对比数据 (打点、日志)
协程请求
Go 版逻辑
请求结果
HTTP Server (Go)

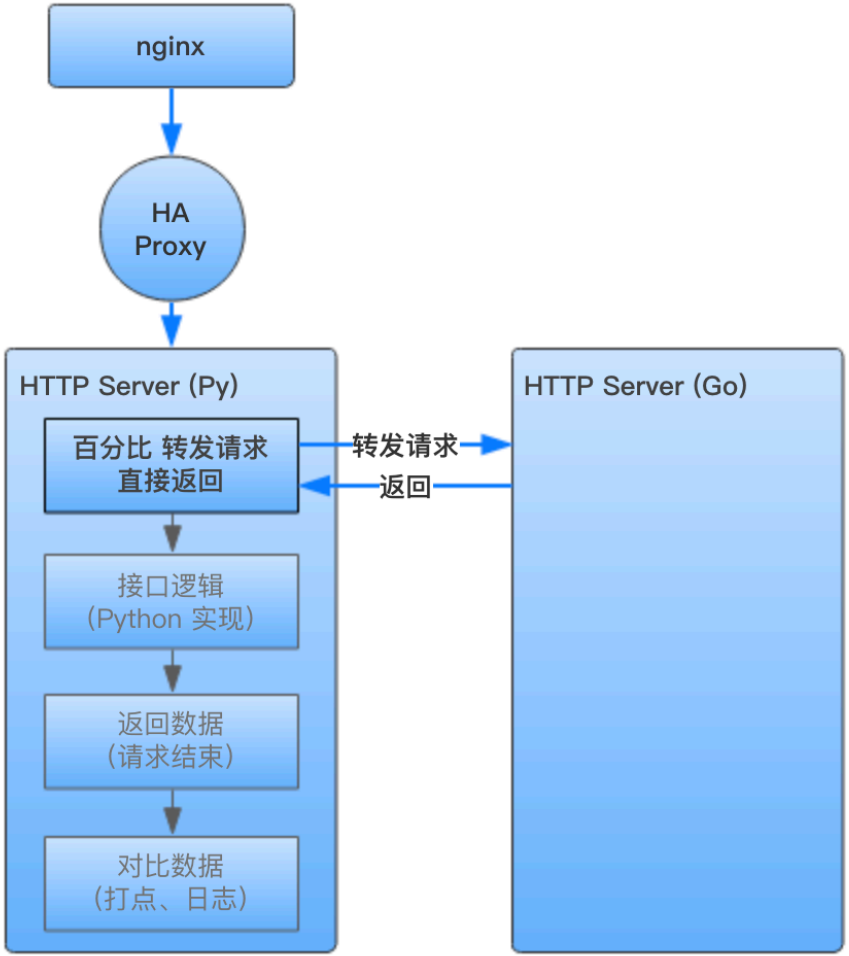
# 重构过程

写接口

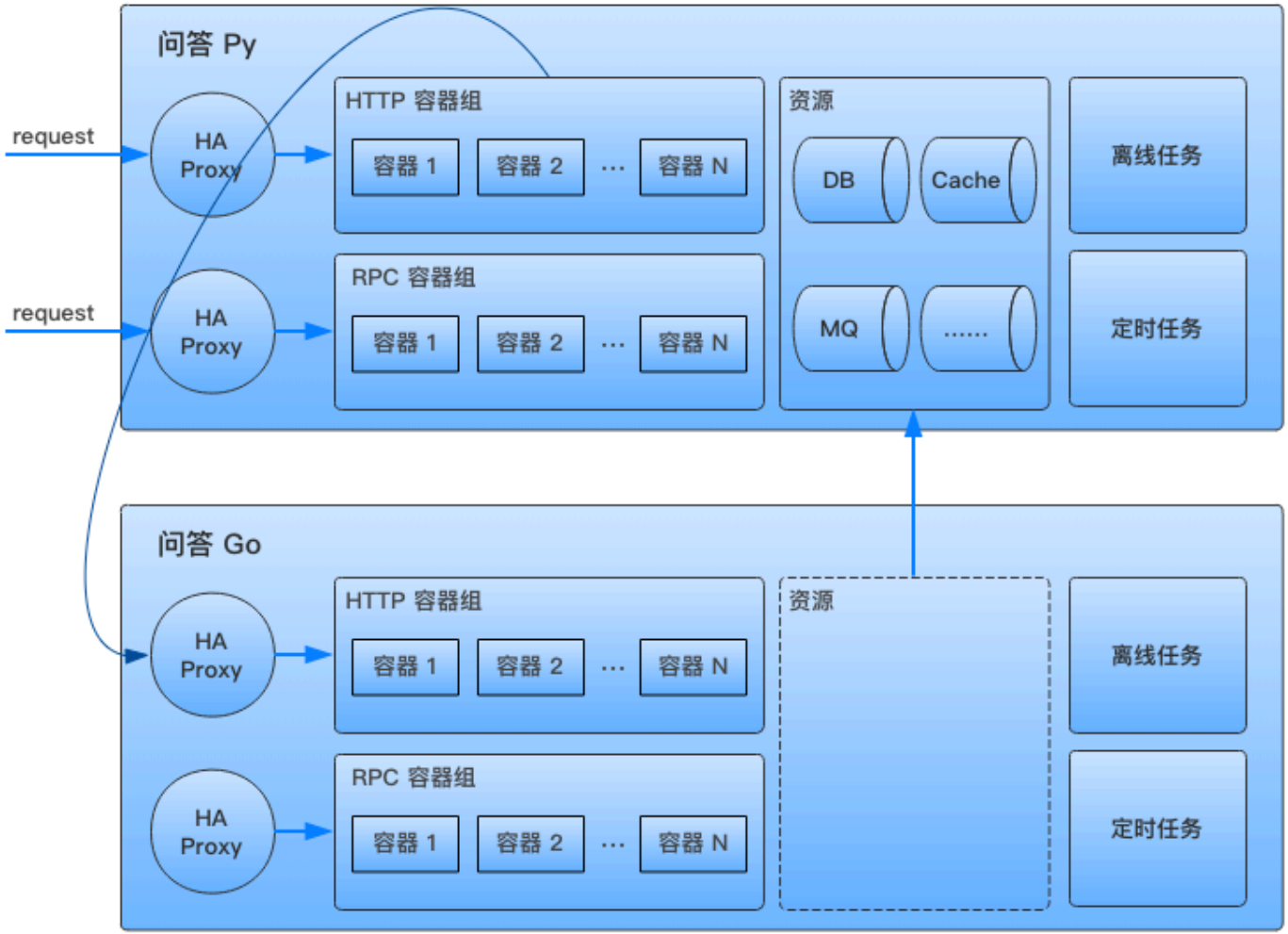
- 单元测试保证
- 开发者验证
- QA 验证
- 办公室环境验证



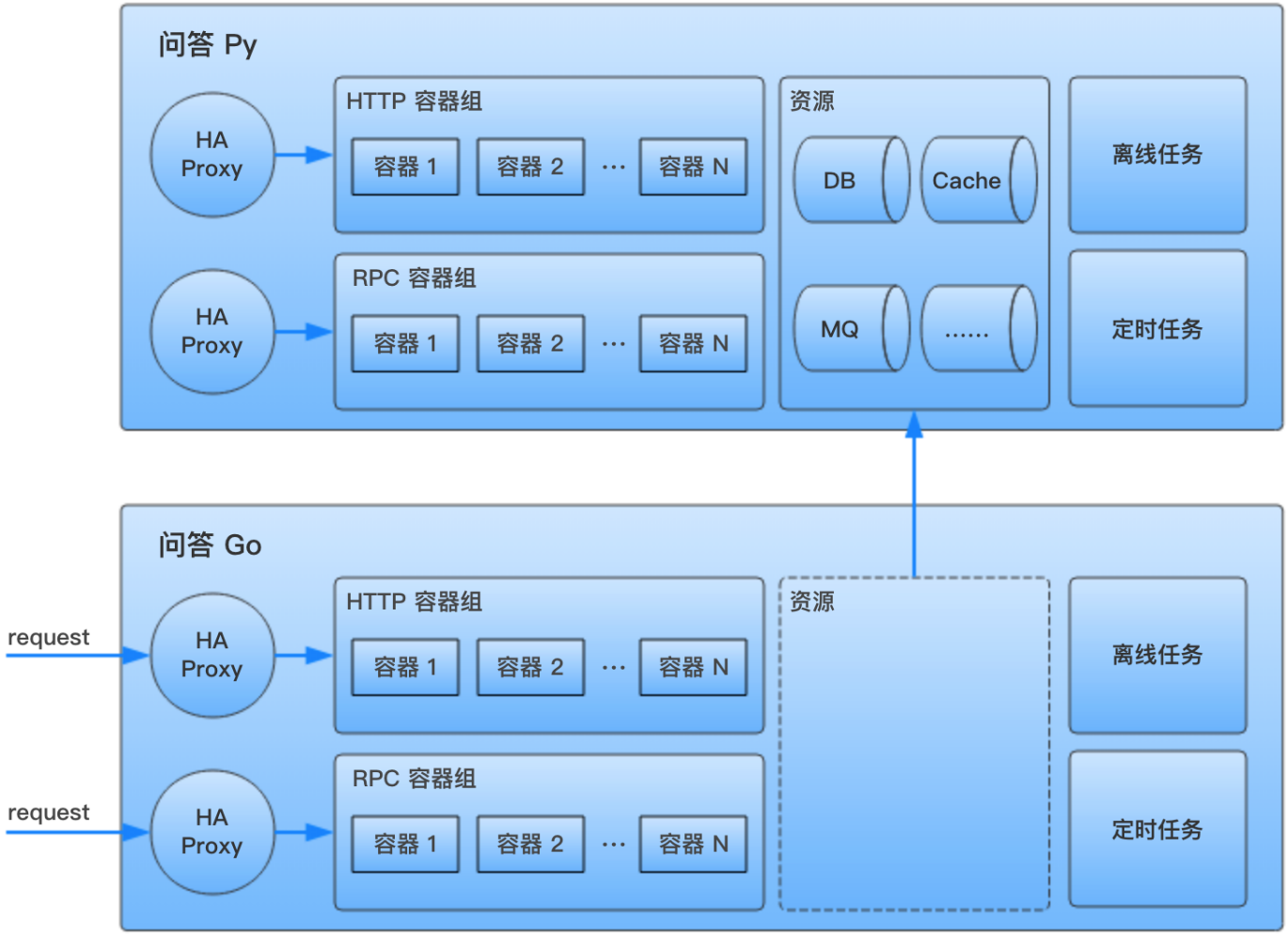
# 重构过程


nginx
HA Proxy
HTTP Server (Py)
百分比 转发请求
直接返回
接口逻辑
(Python 实现)
返回数据
(请求结束)
对比数据
(打点、日志)
转发请求
返回
HTTP Server (Go)

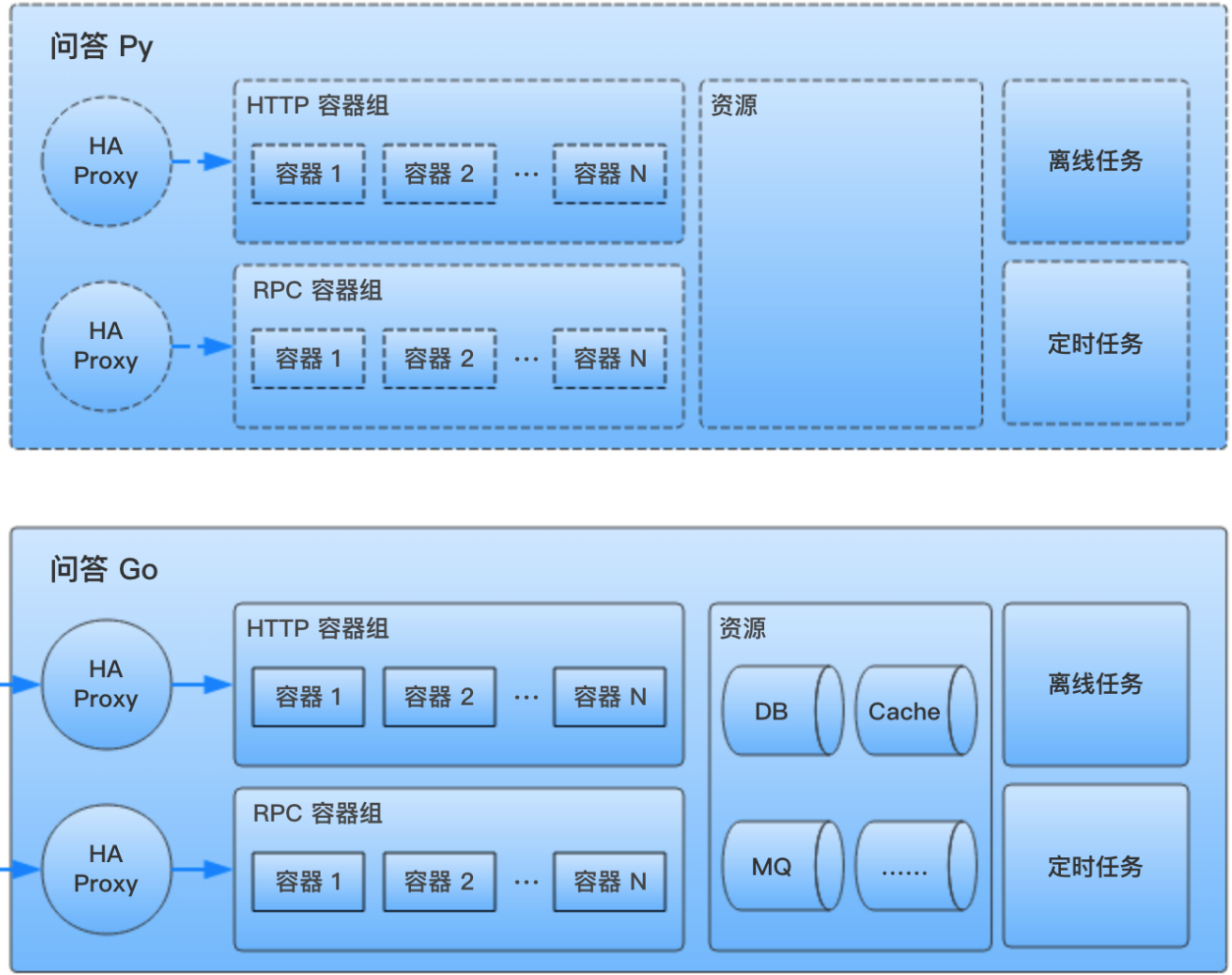
# 重构过程


问答 Py
request
HA Proxy
HTTP 容器组
容器 1
容器 2
…
容器 N
资源
DB
Cache
离线任务
request
HA Proxy
RPC 容器组
容器 1
容器 2
…
容器 N
MQ
.....
定时任务
问答 Go
HA Proxy
HTTP 容器组
容器 1
容器 2
…
容器 N
资源
离线任务
HA Proxy
RPC 容器组
容器 1
容器 2
…
容器 N
定时任务

# 重构过程


问答 Py
HA Proxy
HTTP 容器组
容器 1
容器 2
…
容器 N
资源
DB
Cache
离线任务
HA Proxy
RPC 容器组
容器 1
容器 2
…
容器 N
MQ
.....
定时任务
问答 Go
request
HA Proxy
HTTP 容器组
容器 1
容器 2
…
容器 N
资源
离线任务
request
HA Proxy
RPC 容器组
容器 1
容器 2
…
容器 N
定时任务

# 重构过程


问答 Py
HA Proxy
HTTP 容器组
容器 1
容器 2
…
容器 N
资源
离线任务
HA Proxy
RPC 容器组
容器 1
容器 2
…
容器 N
定时任务
问答 Go
request
HA Proxy
HTTP 容器组
容器 1
容器 2
…
容器 N
资源
DB
Cache
离线任务
request
HA Proxy
RPC 容器组
容器 1
容器 2
…
容器 N
MQ
.....
定时任务

# 一些经验



# 一些经验

重构和优化不要同时做！

P0 新注册用户「串号」故障……



# 一些经验

```
.
├── bin                     --> 构建生成的可执行文件
├── cmd                     --> 各种服务的 main 函数入口（ RPC、Web 等）
│   ├── service
│   │   └── main.go
│   ├── web
│   └── worker
├── gen-go                  --> 根据 RPC thrift 接口自动生成
├── pkg                     --> 真正的实现部分（下面详细介绍）
│   ├── controller
│   ├── dao
│   ├── rpc
│   ├── service
│   └── web
│       ├── controller
│       ├── handler
│       ├── model
│       └── router
├── thrift_files            --> thrift 接口定义
│   └── interface.thrift
├── vendor                  --> 依赖的第三方库（ dep ensure 自动拉取）
├── Gopkg.lock              --> 第三方依赖版本控制
├── Gopkg.toml
├── joker.yml               --> 应用构建配置
├── Makefile                --> 本项目下常用的构建命令
└── README.md
```

# 一些经验

```
pkg/
├── controller
│   ├── ctl.go          --> 接口
│   ├── impl            --> 接口的业务实现
│   │   └── ctl.go
│   └── mock            --> 接口的 mock 实现
│       └── mock_ctl.go
├── dao
│   ├── impl
│   └── mock
├── rpc
│   ├── impl
│   └── mock
├── service             --> 本项目 RPC 服务接口入口
│   ├── impl
│   └── mock
└── web                 --> Web 层（提供 HTTP 服务）
    ├── controller      --> Web 层 controller 逻辑
    │   ├── impl
    │   └── mock
    ├── handler         --> 各种 HTTP 接口实现
    ├── model           -->
    ├── formatter       --> 把 model 转换成输出给外部的格式
    └── router          --> 路由
```



# 一些经验

golang-standards / **project-layout**



# 一些经验

静态代码检查 越早越好

- alecthomas / gometalinter
- golangci / golangci-lint



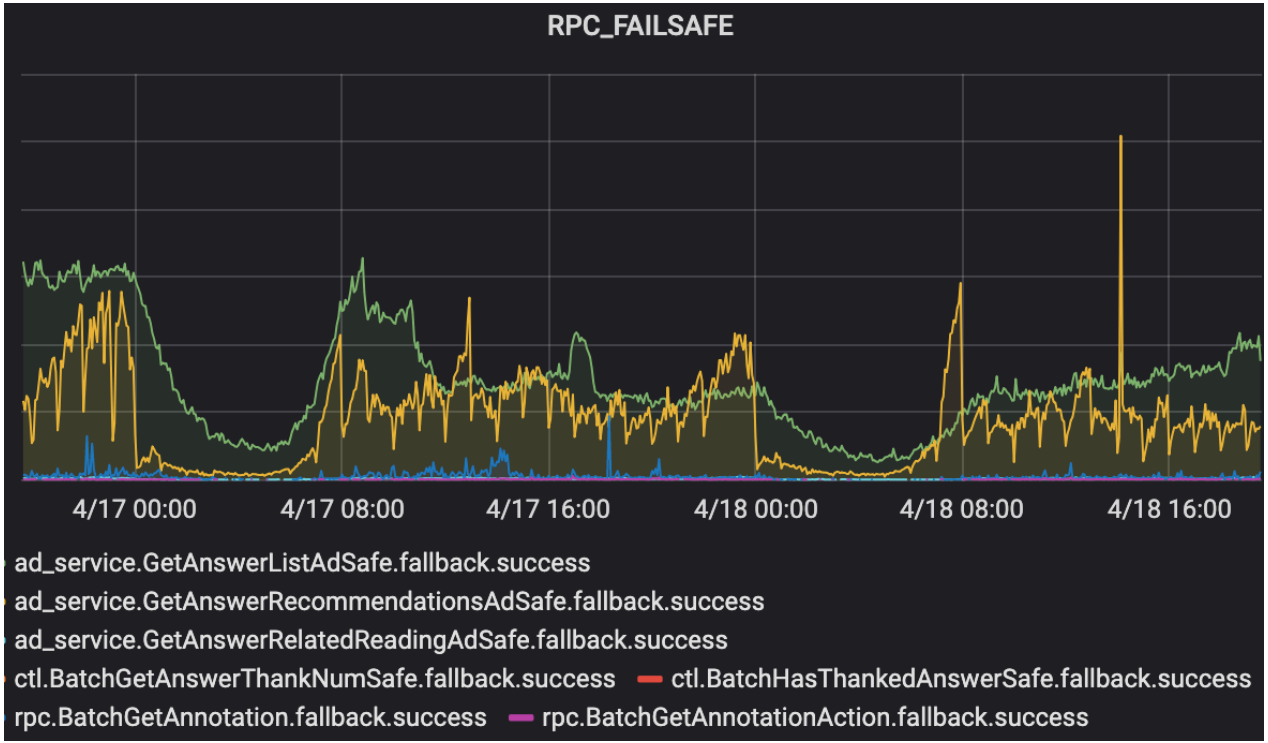
# 一些经验

## 降级 RPC vs 功能

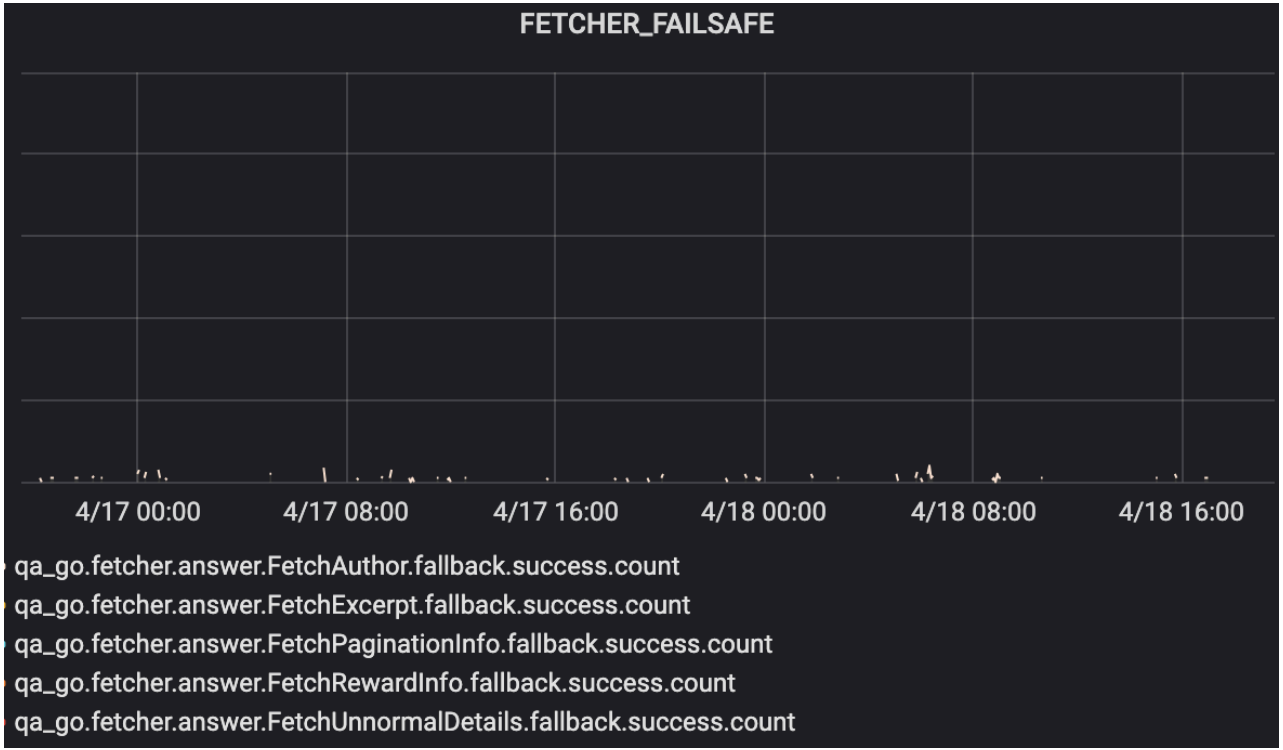
cep21 / circuit



# 一些经验


RPC_FAILSAFE
4/17 00:00
4/17 08:00
4/17 16:00
4/18 00:00
4/18 08:00
4/18 16:00
ad_service.GetAnswerListAdSafe.fallback.success
ad_service.GetAnswerRecommendationsAdSafe.fallback.success
ad_service.GetAnswerRelatedReadingAdSafe.fallback.success
ctl.BatchGetAnswerThankNumSafe.fallback.success
ctl.BatchHasThankedAnswerSafe.fallback.success
rpc.BatchGetAnnotation.fallback.success
rpc.BatchGetAnnotationAction.fallback.success

# 一些经验


FETCHER_FAILSAFE
4/17 00:00
4/17 08:00
4/17 16:00
4/18 00:00
4/18 08:00
4/18 16:00
qa_go.fetcher.answer.FetchAuthor.fallback.success.count
qa_go.fetcher.answer.FetchExcerpt.fallback.success.count
qa_go.fetcher.answer.FetchPaginationInfo.fallback.success.count
qa_go.fetcher.answer.FetchRewardInfo.fallback.success.count
qa_go.fetcher.answer.FetchUnnormalDetails.fallback.success.count

# 一些经验

## panic & recover

不推荐，但真香



# 一些经验

```
 1 func (h *AnswerHandler) Get(w http.ResponseWriter, r *http.Request) {
 2     ctx := r.Context()
 3
 4     loginId, err := auth.GetLoginId(r)
 5     if err != nil {
 6         zapi.RenderError(err)
 7 --->     return
 8     }
 9
10     answer, err := h.PrepareAnswer(ctx, r, loginId)
11     if err != nil {
12         zapi.RenderError(err)
13 --->     return
14     }
15
16     formattedAnswer, err := h.ctl.FormatAnswer(ctx, loginId, answer)
17     if err != nil {
18         zapi.RenderError(err)
19 --->     return
20     }
21
22     zapi.RenderJSON(w, fAnswer)
23 }
```



# 一些经验

```go
func (h *AnswerHandler) Get(w http.ResponseWriter, r *http.Request) {
    ctx := r.Context()

    loginId := MustGetLoginId(r)

    answer := h.MustPrepareAnswer(ctx, r, loginId)

    formattedAnswer := h.ctl.MustFormatAnswer(ctx, loginId, answer)

    zapi.RenderJSON(w, fAnswer)
}
```



# 一些经验

## Render, then render

RenderOnce

```
{“error”: 12345}{“success”: true}
```



# 一些经验

## Goroutine panic

## Future

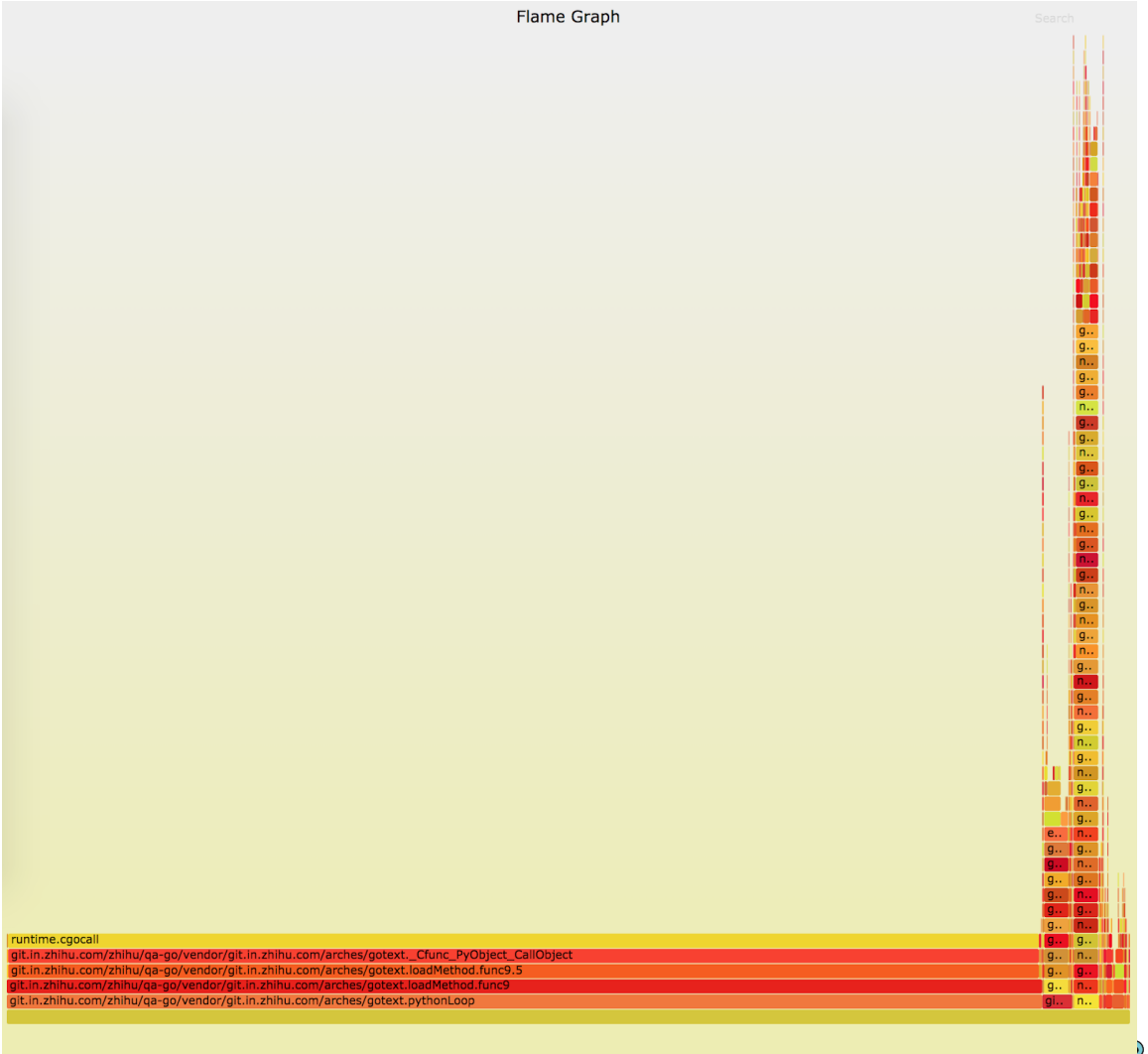


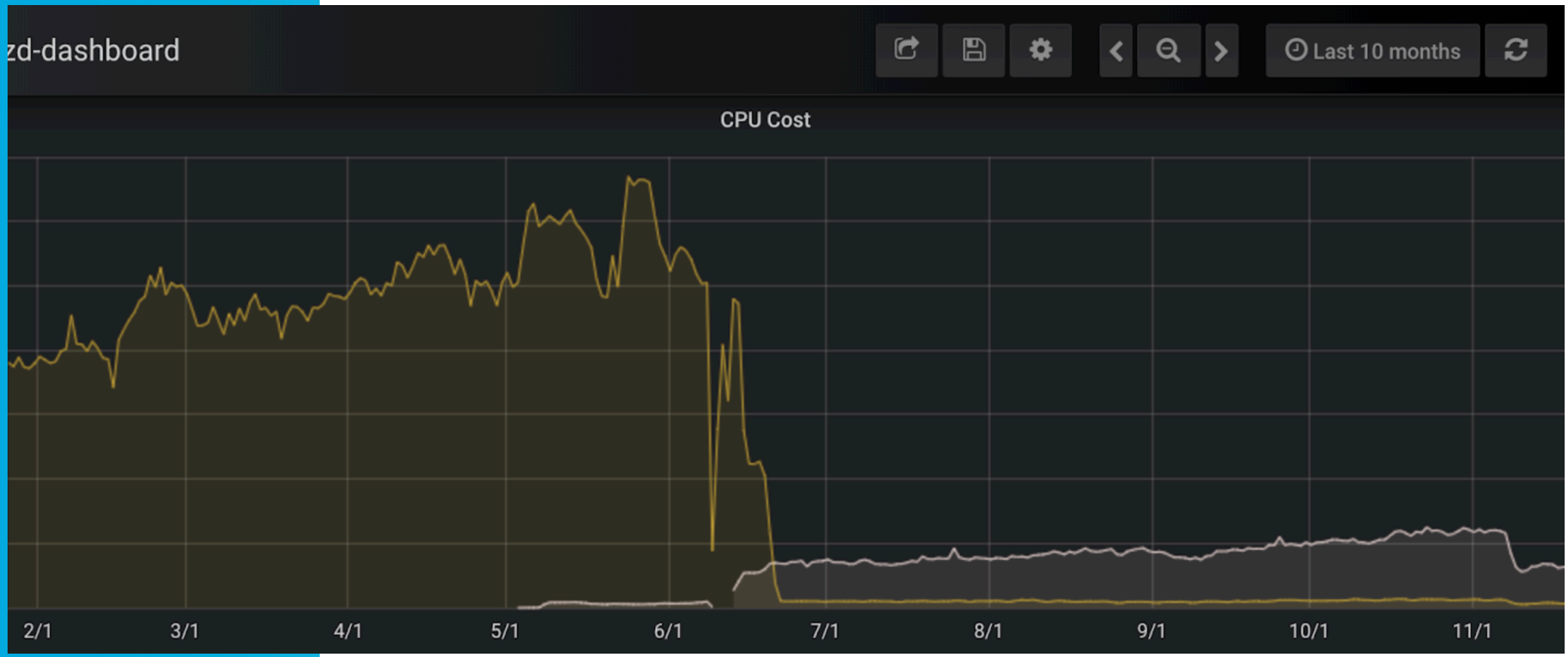
# 一些经验

Golang call Python?



# 一些经验


Flame Graph
Search
runtime.cgocall
git.in.zhihu.com/zhihu/qa-go/vendor/git.in.zhihu.com/arches/gotext._Cfunc_PyObject_CallObject
git.in.zhihu.com/zhihu/qa-go/vendor/git.in.zhihu.com/arches/gotext.loadMethod.func9.5
git.in.zhihu.com/zhihu/qa-go/vendor/git.in.zhihu.com/arches/gotext.loadMethod.func9
git.in.zhihu.com/zhihu/qa-go/vendor/git.in.zhihu.com/arches/gotext.pythonLoop


zd-dashboard
Last 10 months
CPU Cost
2/1
3/1
4/1
5/1
6/1
7/1
8/1
9/1
10/1
11/1

THANKS