



Golang 高性能实战

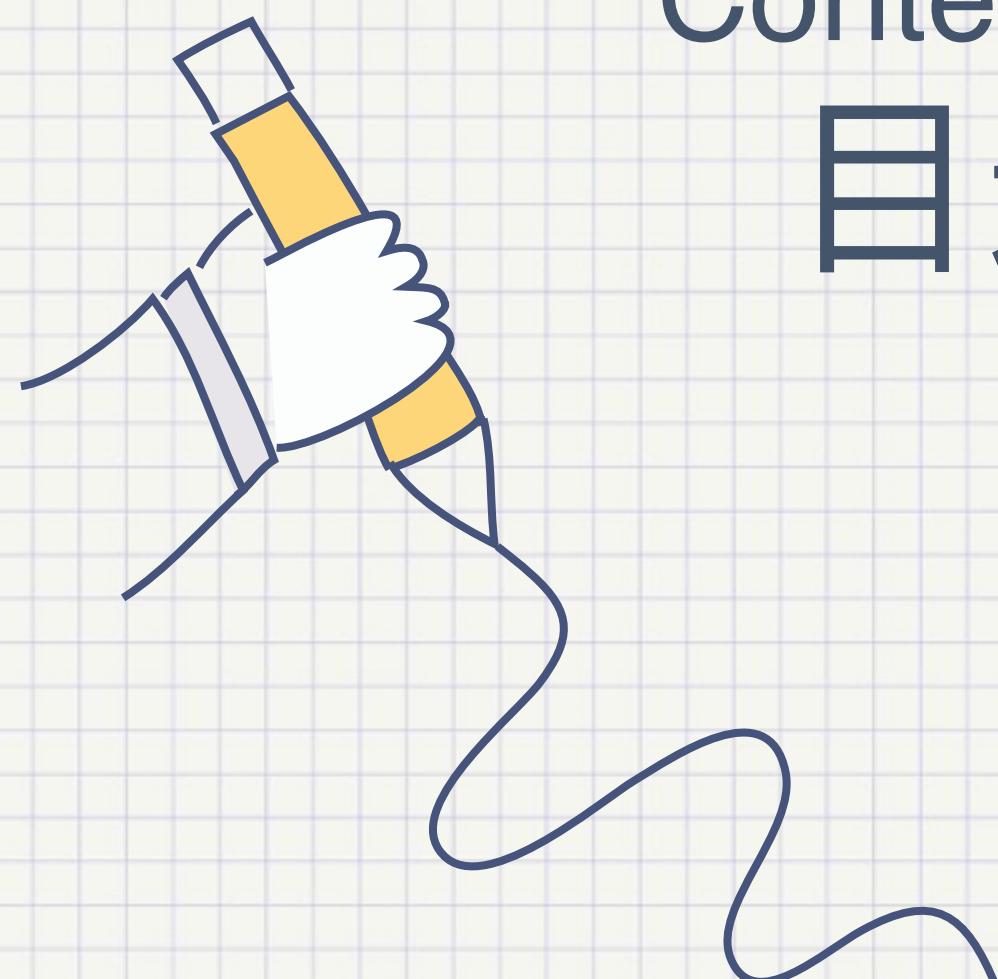
github.com/rfyiamcool
xiaorui.cc



CDN 刷新系统 是一个高性能、可扩
展的分布式系统.

支持全网刷新及预缓存业务.

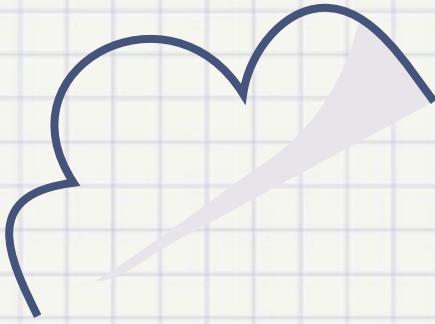




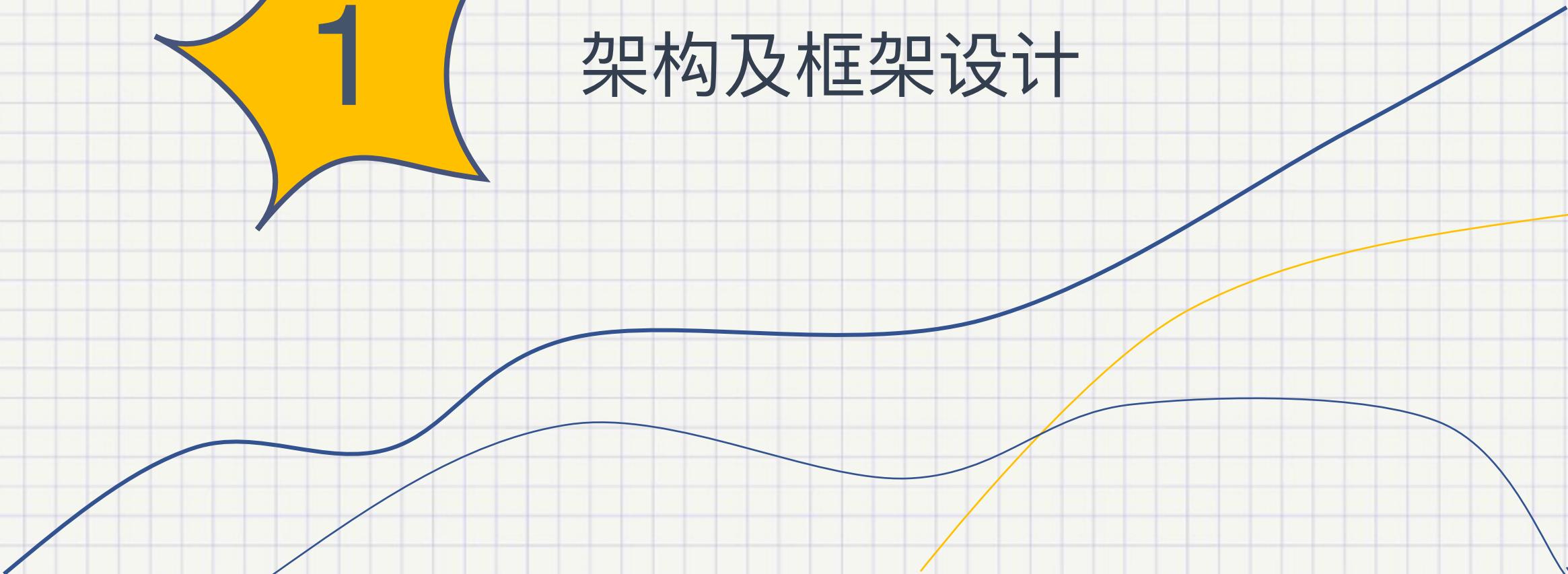
Contents

目录

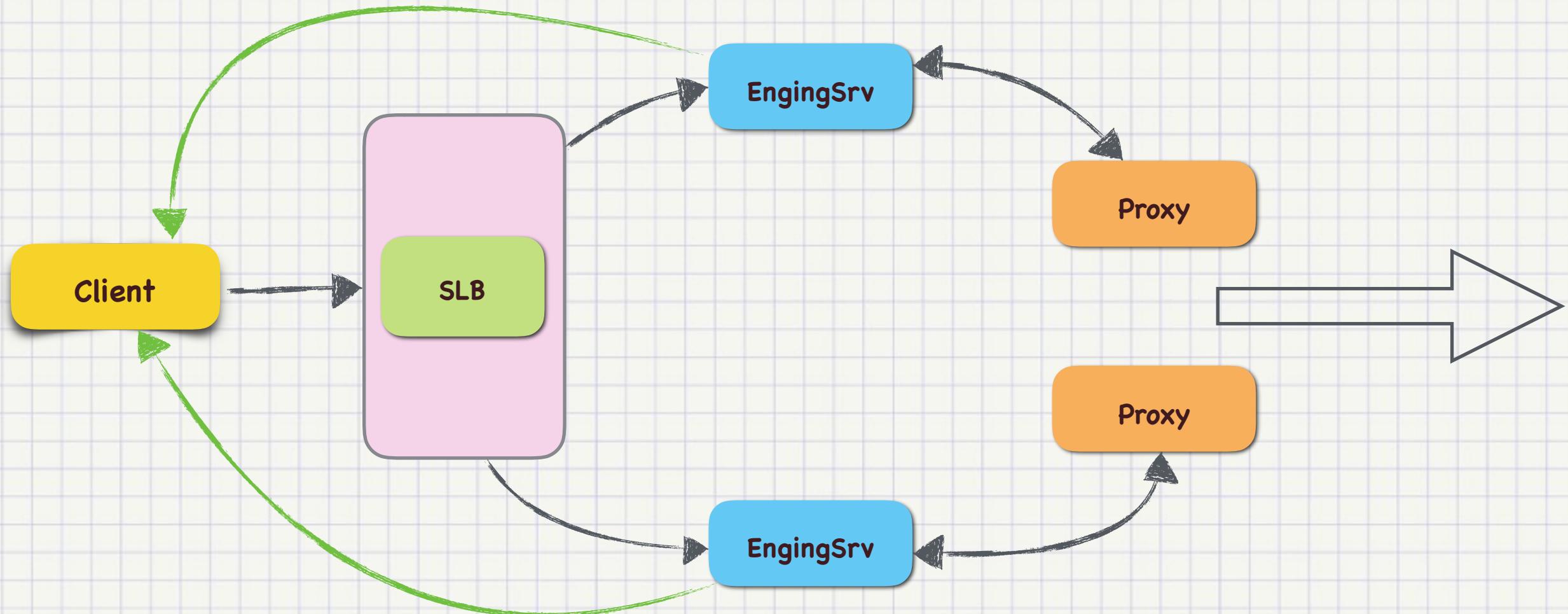
- 1 架构及框架
- 2 调优
- 3 深度调优
- 4 上线部署
- 5 总结



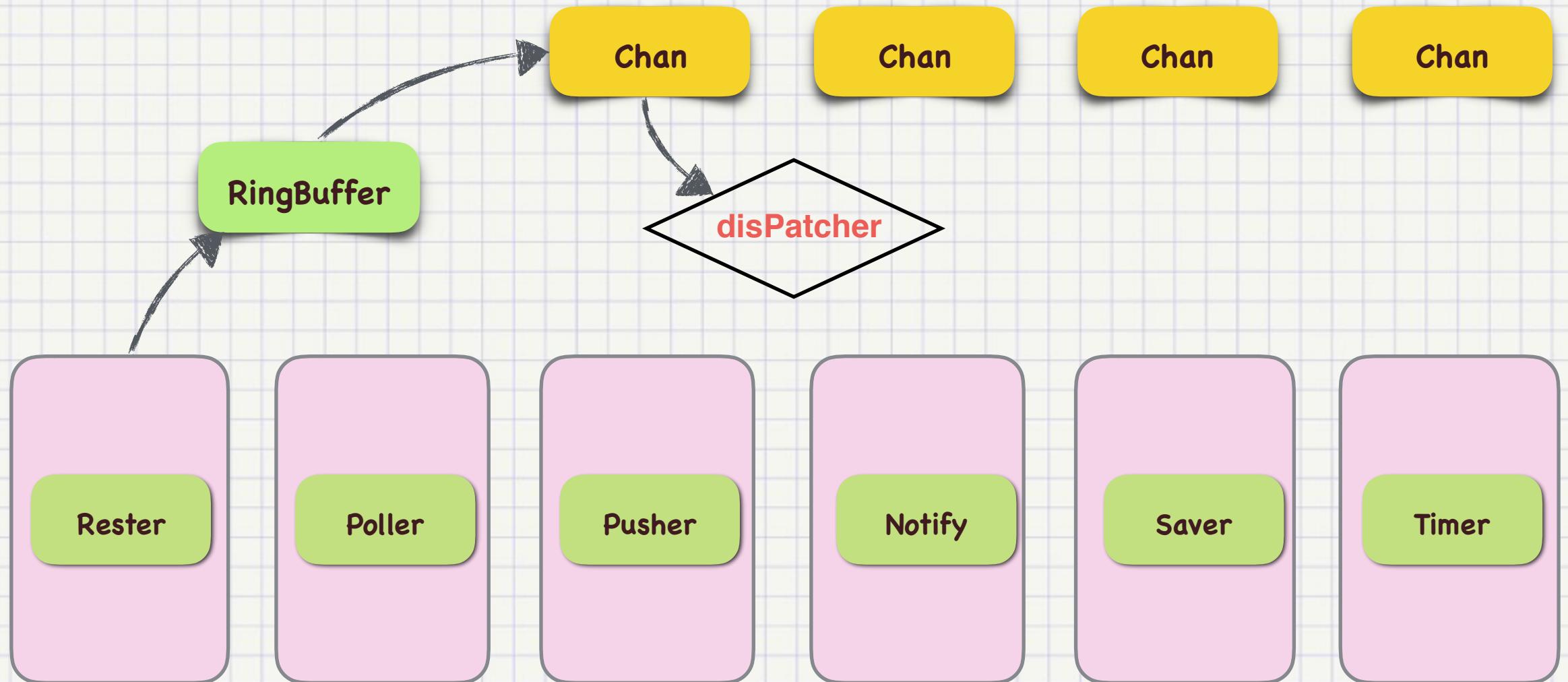
架构及框架设计

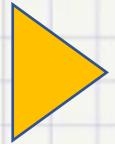


架构设计



框架设计





技术选型



语言: Golang



框架: Gin



压缩: Snappy

序列化: Json-iterator

ORM: Gorm

客户端: imroc/req



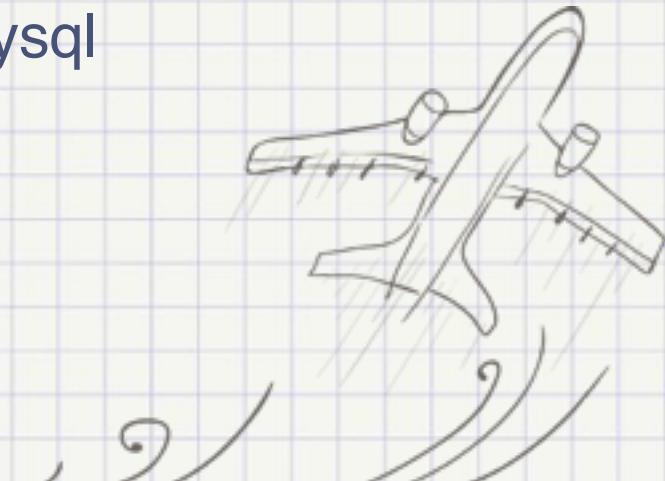
日志: Logrus

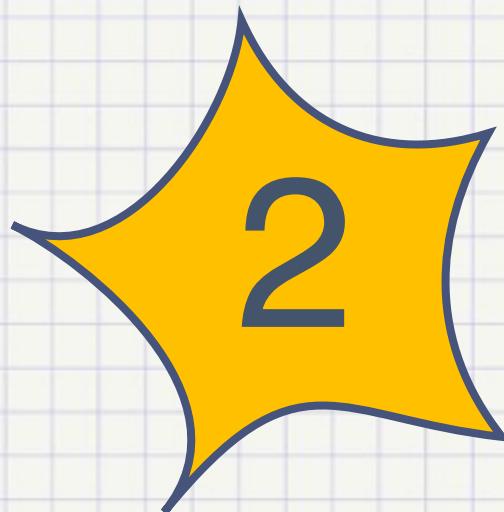


配置: Toml

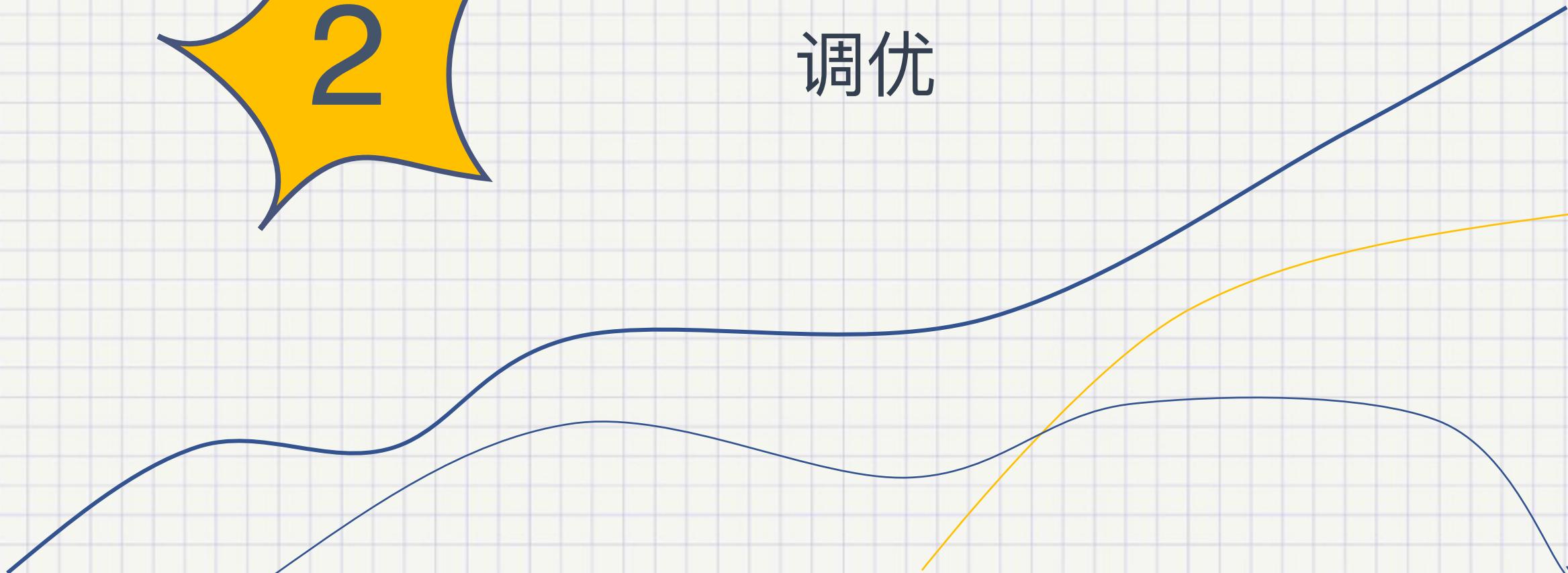
缓存: Redis

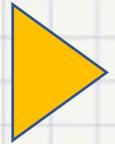
持久化: Mysql





调优





基本优化



不要滥用time.After, 减少gc压力

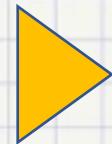
不要滥用goroutine, 减少gc压力

不要滥用锁, 引起runtime上下文切换



使用连接池, 减少网络RTT消耗。

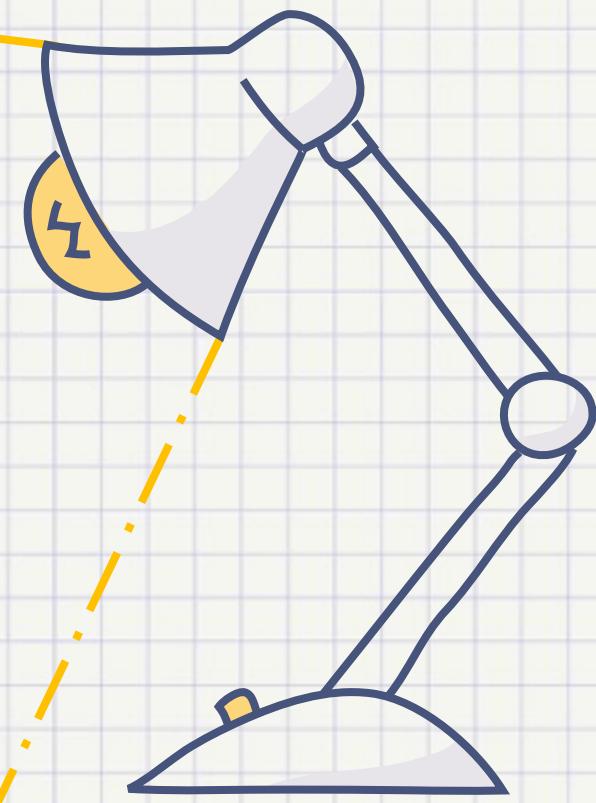


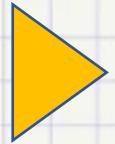


基本优化



提前预估size, make 之
临时的mapl slice, 可采用sync.pool
大于32kb, 也可使用





业务调优



减少网络io的消耗

批量接口

异步写入

网络调用链



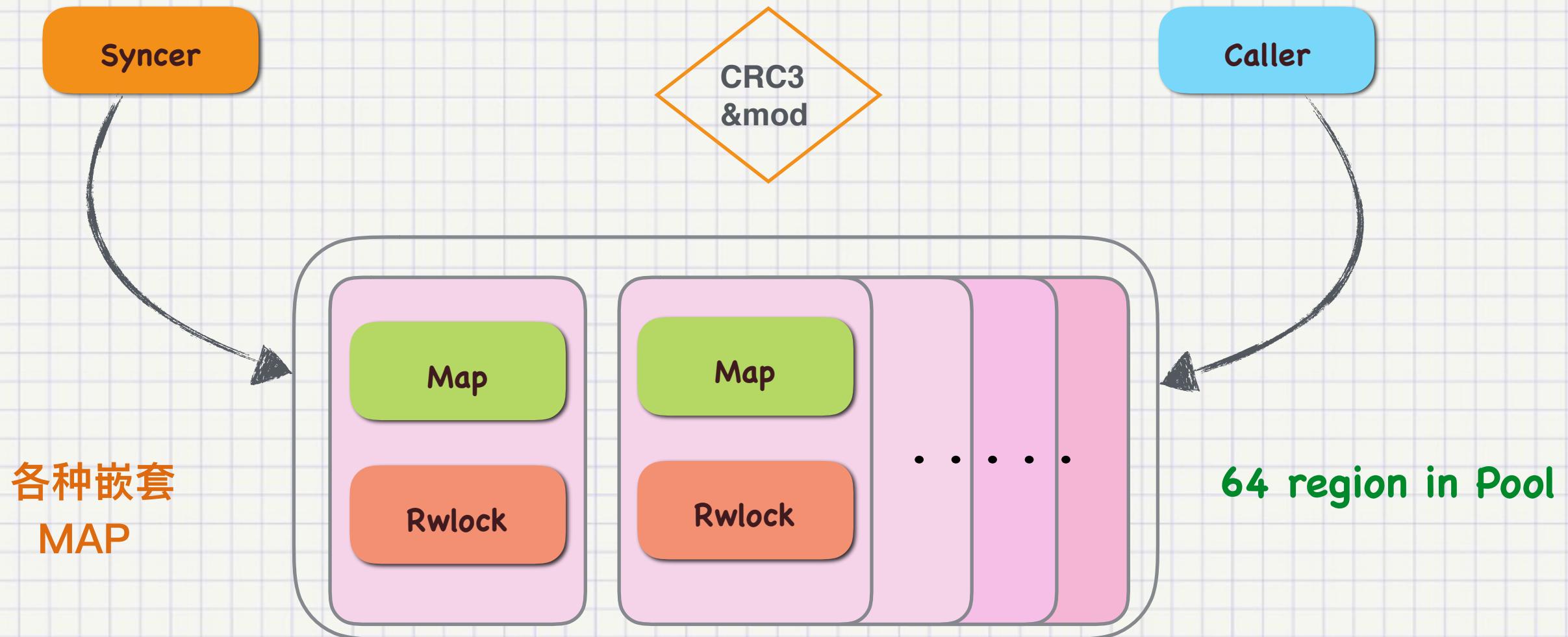
缓存内存化

减少同步逻辑

使用压缩，节省带宽

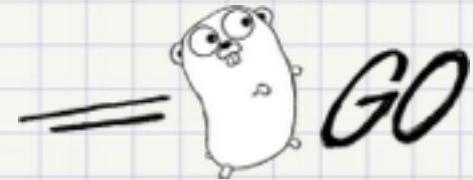


全局任务状态





全局任务状态-code



```
var (
    TaskMapRegionPool = [PreAllocRegion]*TaskMapRegion{}
)
```

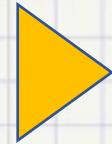
```
type TaskMapRegion struct {
    map[string]*DetailTaskStats
    *sync.RWMutex
}
```

```
type DetailTaskStats struct {
    TaskId      string
    TaskType    string
    WorkingLayer string
    Active      bool // 是否还需要执行?
    Counter     int   // 计数器
    Total       int   // 任务总数
    DueTimeStamp time.Time
    CreatedAt   time.Time
    UpdatedAt   time.Time
    Subtasks    map[string]map[string]string
}
```

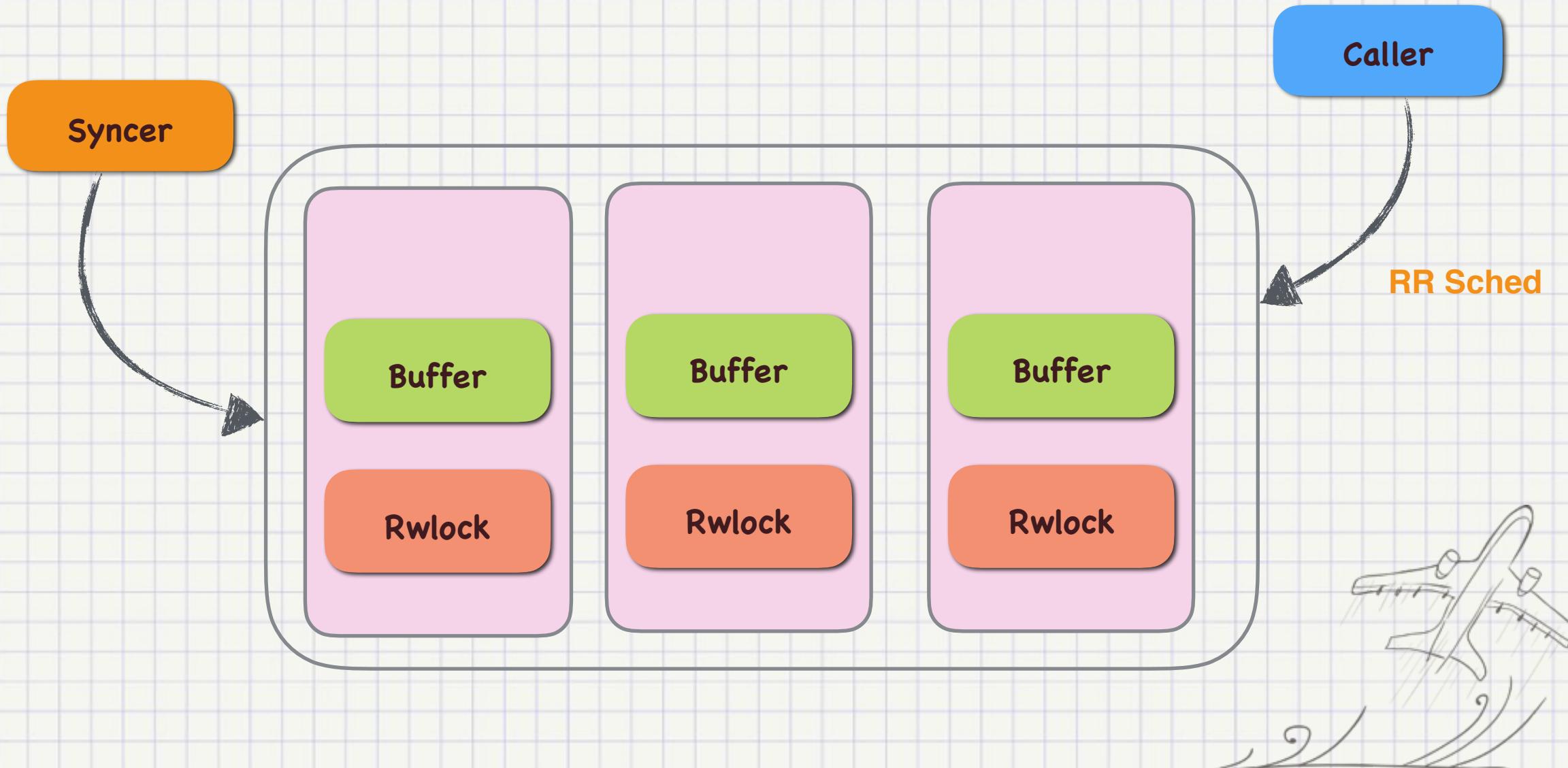
```
func GetOneMapRegion(taskId string) *TaskMapRegion {
    idx := utils.HashToInt(taskId) % PreAllocRegion
    return TaskMapRegionPool[idx]
}
```

```
func ExistInRegionPool(taskID string) bool {
    reg := GetOneMapRegion(taskID)
    reg.AcquireR()
    defer reg.ReleaseR()

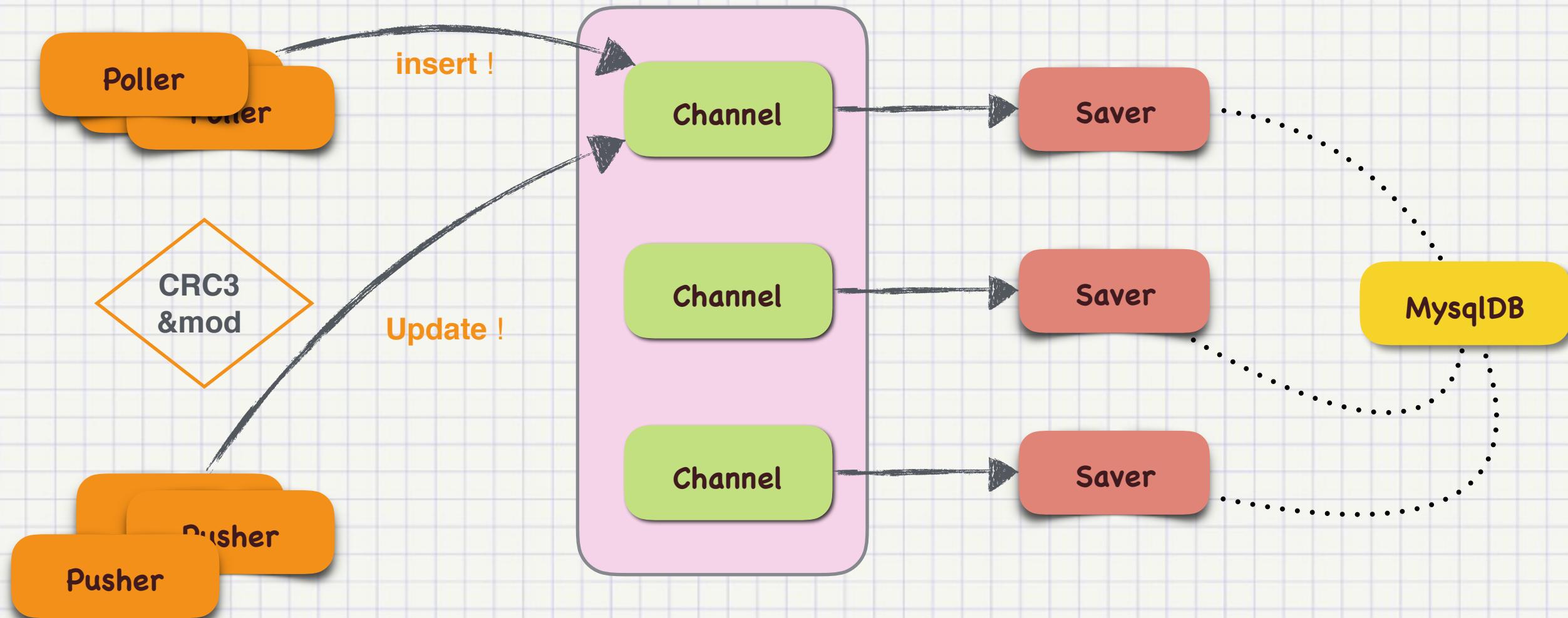
    _, ok := reg.HasTaskID(taskID)
    return ok
}
```



缓存多副本



操作顺序排队





操作顺序排队-CODE

```
var (
    preSaveNum      int
    saveQueuePool []chan *saveAction
)

func getOneSaveQueue(taskId string) chan *saveAction {
    idx := utils.HashToInt(taskId) % preSaveNum // range 0 <-> (preSaveNum - 1)
    return saveQueuePool[idx]
}

func pushOneSaveQueue(taskId string, data *saveAction) bool {
    queue := getOneSaveQueue(taskId)
    select {
    case queue <- data:
        return true
    default:
        // log
        time.Sleep(1 * time.Second)
        return false
    }
}

func GetSaverLength() int {
    c := 0
    for _, queue := range saveQueuePool {
        c += len(queue)
    }
    return c
}
```

```
// 消费队列
func (s *saveHandler) popQueue() (*saveAction, bool) {
    var outDto *saveAction
    var ok bool

    select {
    case outDto = <-saveQueuePool[s.slot]:
        ok = true
    case <-EngineWaitGroup.ExitQ:
        return outDto, false
    }

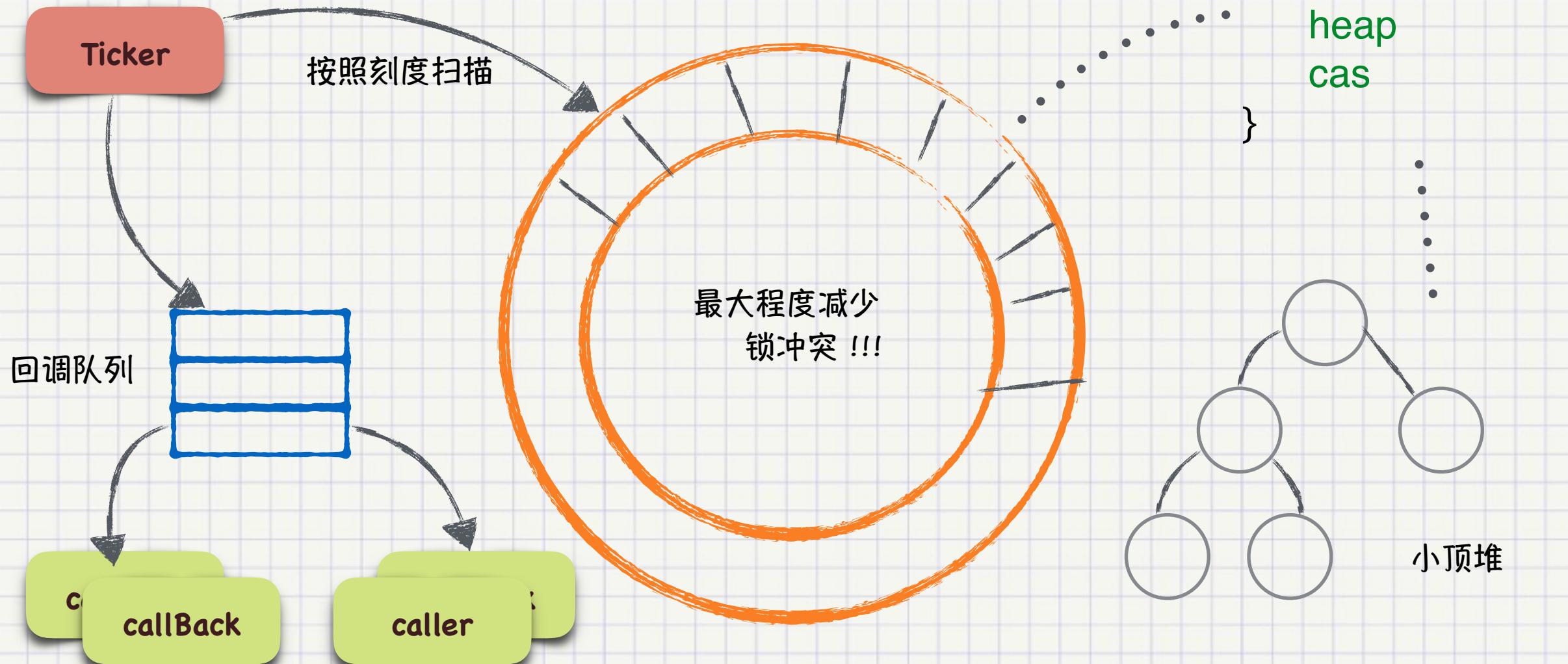
    return outDto, ok
}

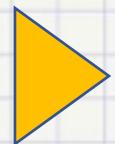
// 分段的槽位需要跟协程数对应
func initSaveQueuePool(size int) {
    var defOneSize = 100

    preSaveNum = conf.Gconf.Engine.SaverWorkerNum
    saveQueuePool = make([]chan *saveAction, preSaveNum, preSaveNum)

    oneSize := size / preSaveNum
    if oneSize < defOneSize {
        oneSize = defOneSize
    }
}
```

时间轮设计

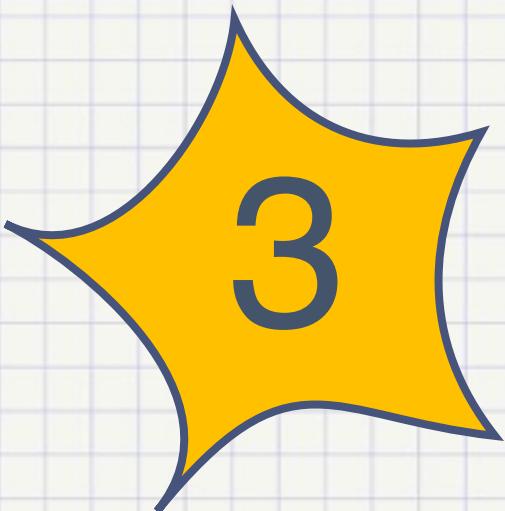


 stats 全局状态

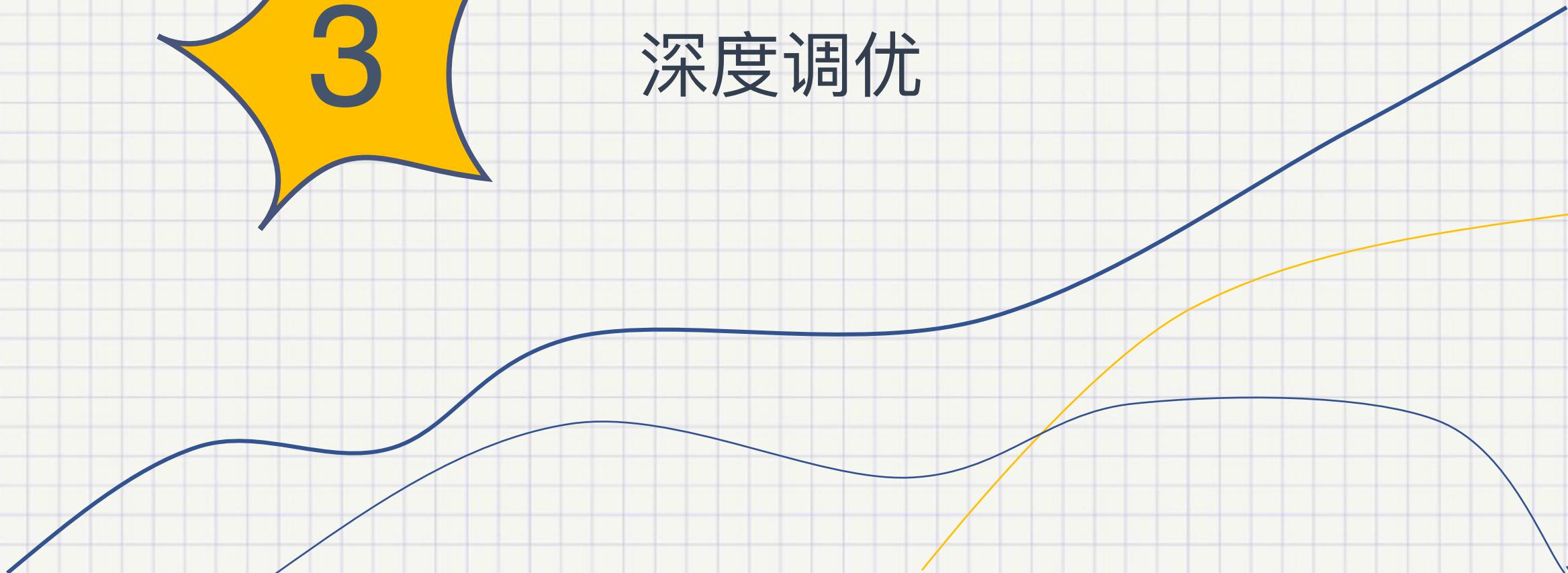
```
{  
  "alert_queue": 30,  
  "input_task_queue": 400,  
  "notify_queue": 1092,  
  "pushing_queue": 311,  
  "reporting_queue": 612,  
  "saver_queue": 3197,  
  "tasks_in_cache": 13189,  
  "timer_tasks": 13189,  
  "timestamp": 1521682815}
```



```
"qps_proxy": {  
  "15m": 9031,  
  "1m": 10221,  
  "5m": 11005,  
  "avg": 91110,  
  "count": 897811  
},  
"qps_boss": {  
  "15m": 5113,  
  "1m": 5200,  
  "5m": 5300,  
  "avg": 5001,  
  "count": 297811  
},  
}
```



深度调优





SystemTool



perf top



lsof、netstat、sar



go tool pprof



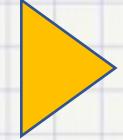
strace



iostat、vmstat、top



wrk



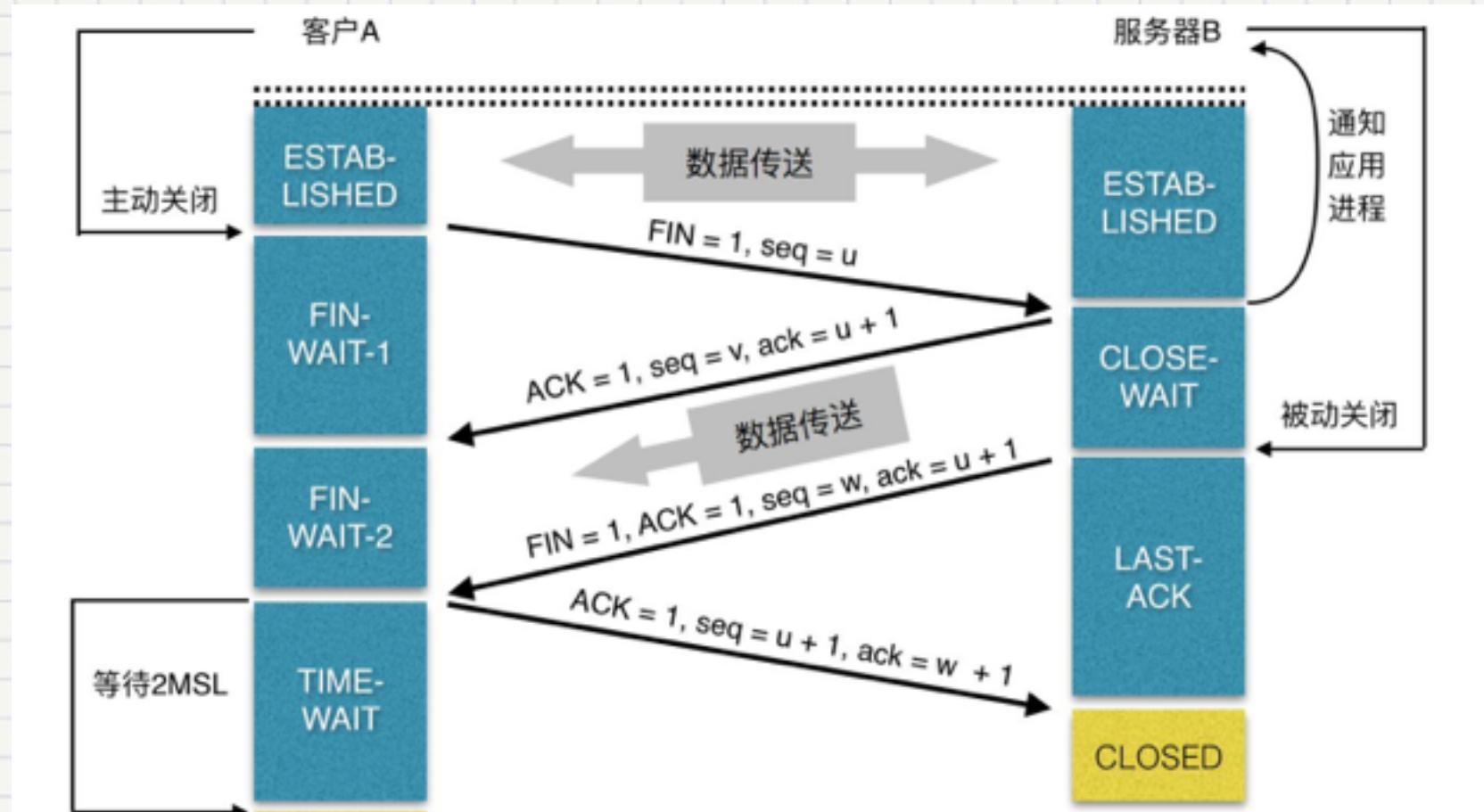
TimeWait 连接暴增问题

TimeWait 暴增到 3.5W ...

TimeWait 暴增到 3W ...

TimeWait 暴增到 4W ...

TimeWait 暴增到 5W ...





TimeWait 连接暴增问题

TimeWait 暴增到 3.5W ...

TimeWait 暴增到 3W ...

TimeWait 暴增到 4W ...

TimeWait 暴增到 5W ...

http.DefaultTransport.(*http.Transport).MaxIdleConnsPerHost = 200

const DefaultMaxIdleConnsPerHost = 2

t := &http.Transport{

...

DialContext: (&net.Dialer{

...

}).DialContext,

MaxIdleConnsPerHost: 200,

MaxIdleConns: 1000,

...



系统线程暴涨

profiles:

0 [block](#)

398 [goroutine](#)

760 [heap](#)

0 [mutex](#)

1009 [threadcreate](#)

[full goroutine stack dump](#)

wsec/s	avgrq-sz	avgqu-sz	await	r_await	w_await	svctm	%util
113328.00	809.49	145.55	602.14	0.00	602.14	7.14	100.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
wsec/s	avgrq-sz	avgqu-sz	await	r_await	w_await	svctm	%util
111480.00	995.36	132.68	1184.48	0.00	1184.48	8.93	100.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00





系统线程暴涨

`strace -fp pid -s 512`

```
write(8, .....
mmap(NULL, 10489856, PROT_READ|PROT_...
mprotect(0x7f04d1e6d000, 4096,...
clone(child_stack=0x7f04d286cff0,
...
clone(child_stack=0x7f04d286cff0,
```

- 线程不会被回收
- 每个线程8M内存
- 增加调度消耗
- **cpu cache miss**
- **more**

`lsof -P -p pid -n`

```
refresh_master 2184 root    8u   REG   ... refresh_master/logs/http.log
```



忙轮询

```
...
for {
    select {
        case res = <-dataQueue:
            return res
        case <-done:
            return res
        default:
            // do something
    }
    return res
}
```

Samples: 78K of event 'cpu-clock', Event count (approx.): 12156151512			
Overhead	Shared Object	Symbol	
49.09%	k	[.] runtime.chanrecv	
25.58%	k	[.] runtime.selectnbrecv	
17.98%	k	[.] main.runner	
1.01%	[kernell]	[k] finish_task_switch	

- **perf top** 可以看出 **select chan** 奇高
- **strace** 几乎无调用，单纯**cpu**指令
- **top** 直接 100%



忙轮询





Json性能问题





Json 性能问题

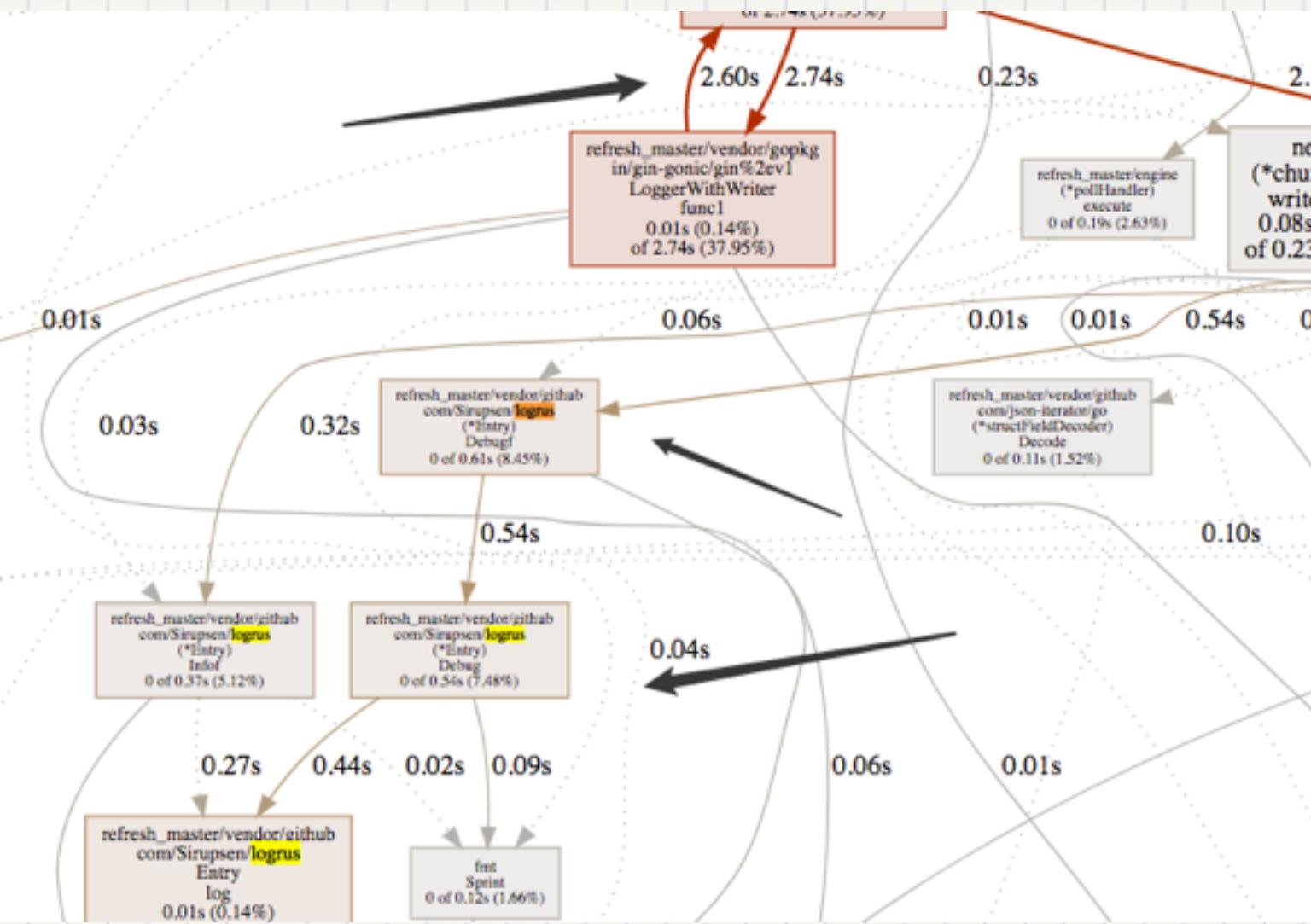
- 固定数据结构
 - *json-iterator/go*
- 不固定数据结构
 - *buger/jsonparser*

减少反射, 减少对象创建 !!!

- 上层代码
 - 直接替换
- 第三方库修改方法
 - 修改**Vendor**相关库的引用 .
 - 使用**Monkey**的替换标准库 .



pprof profile



追踪耗时的相关逻辑

由于日志级别过低
导致调用过于频繁 .



压测

- 压测工具
 - wrk > ab > golang
- 不要本地环路测试
- 多源压力
- 注意带宽问题
- 注意内核的种种error
- more...

Mar 15 16:56:55 xxx kernel: nf_conntrack: table full, dropping packet.
Mar 15 16:56:55 xxx kernel: nf_conntrack: table full, dropping packet.
Mar 15 16:56:55 xxx kernel: nf_conntrack: table full, dropping packet.

Mar 10 13:45:11 xxxx kernel: TCP: time wait bucket table overflow
Mar 10 13:45:11 xxxx kernel: TCP: time wait bucket table overflow
Mar 10 13:45:11 xxxx kernel: TCP: time wait bucket table overflow

more





Wrk

```
wrk -t8 -c800 -d100s -T10s --script=js --latency http://xxxxx:9181/v1/engine/task
```

thread addr: xxxxx

...

8 threads and 800 connections

Thread Stats	Avg	Stdev	Max	+/- Stdev
Latency	19.60ms	11.97ms	144.14ms	73.42%
Req/Sec	23.73k	2784.29	29.85k	79.29%

Latency Distribution

50%	17.90ms
75%	24.92ms
90%	34.63ms

...

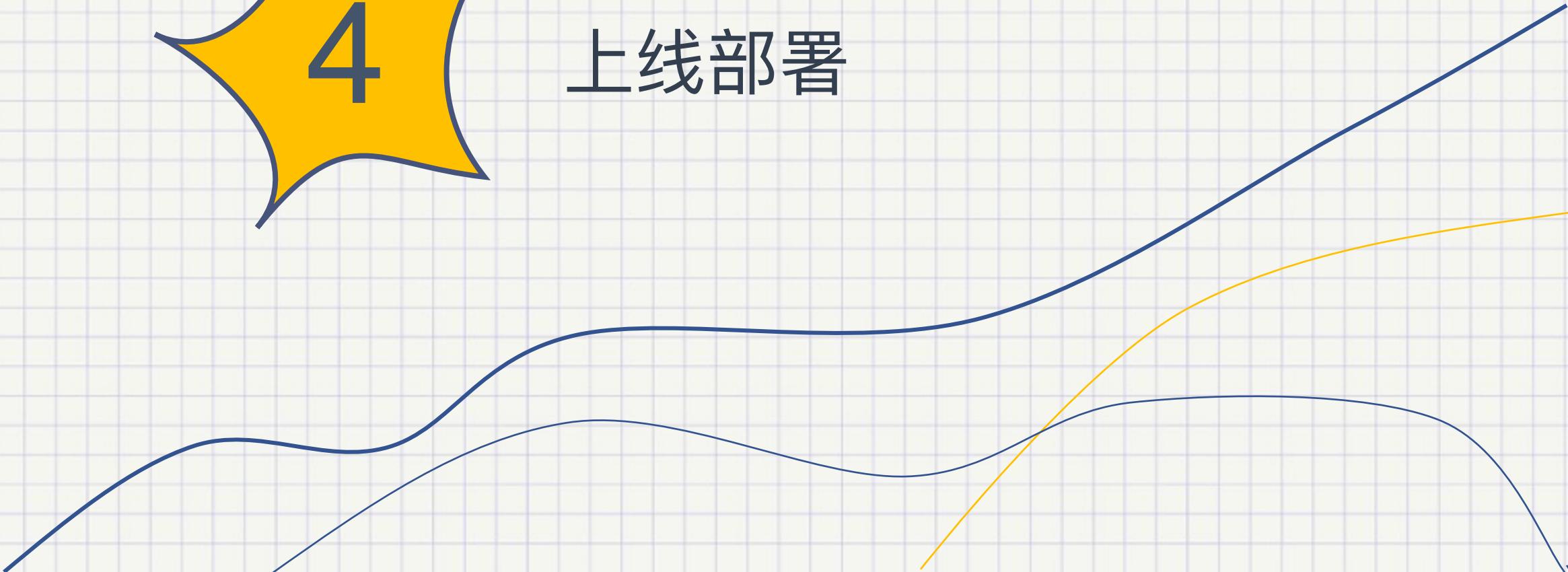
Requests/sec: 23189.11

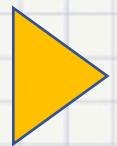
Transfer/sec: 5.06MB





上线部署





打包方法

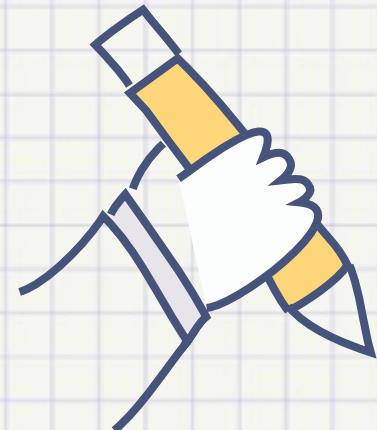


Enging端

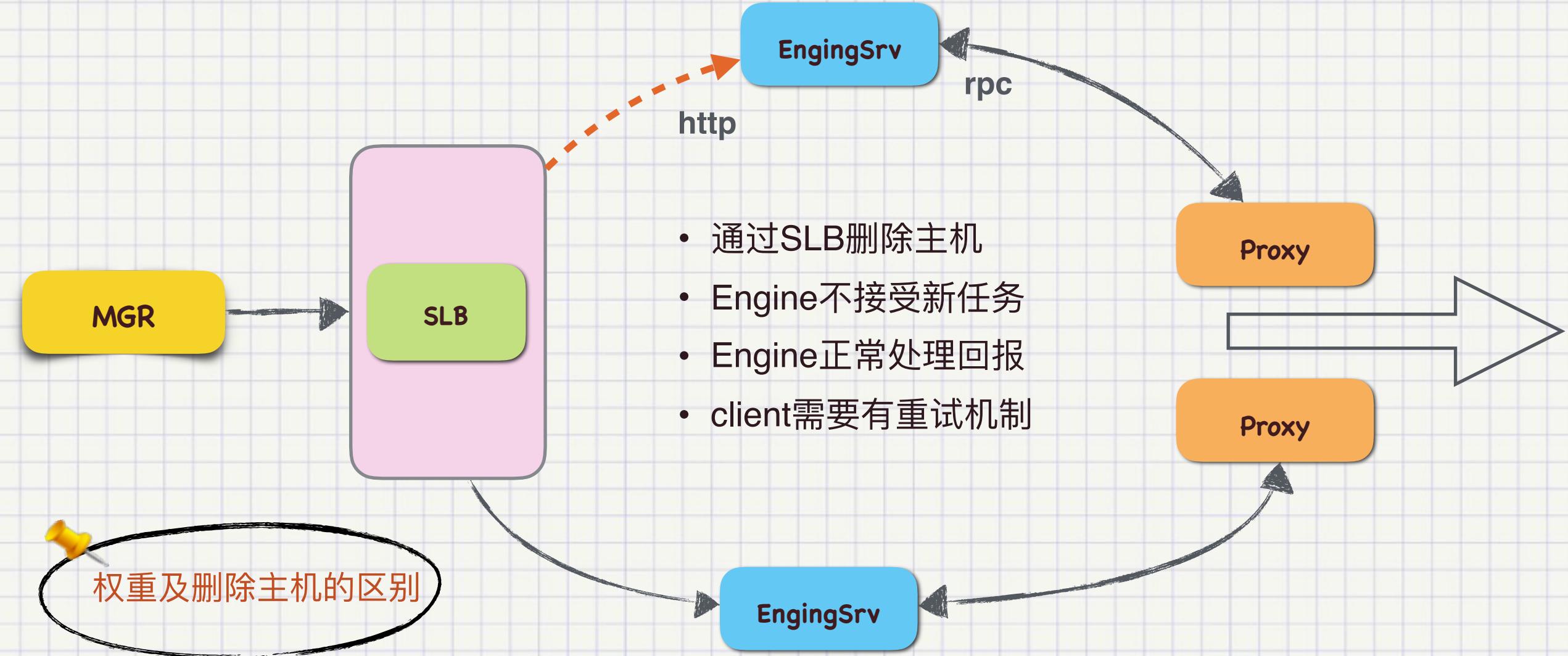
- build
- pack
- repo
- salt
- unpack
- reload

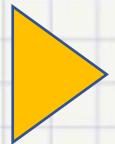


Proxy端, 封装RPM包上线 .

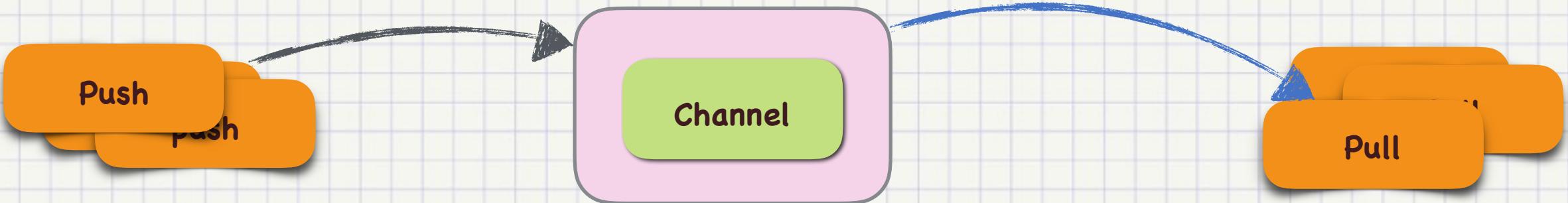


灰度平滑迁移-SLB层





灰度平滑迁移-框架层

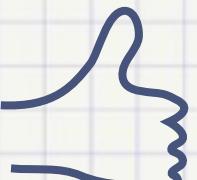


- 从调用链最前端开始**顺序**关闭.
- **持久化**的数据要优先保证
- join住所有的业务协程.

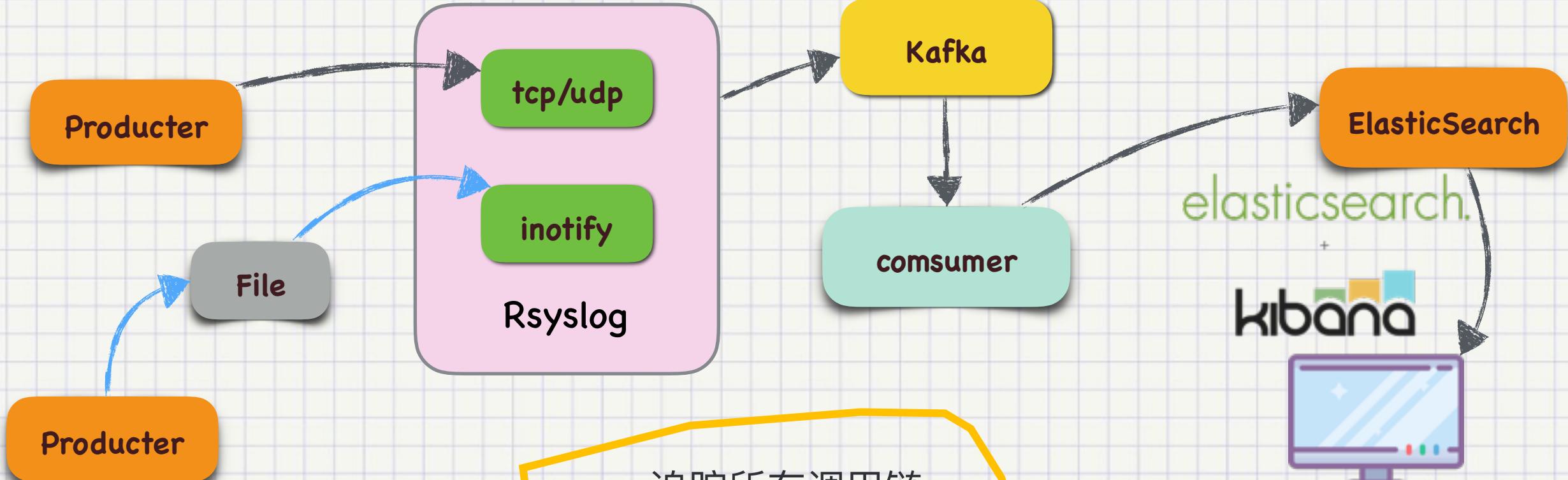
伪代码

```
recvAndSetExit(signal)  
publishSigWorker()
```

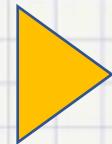
```
if globalExited && len(queue) == 0 && pushExit {  
    return  
}  
wait(saveQueueExit)  
wait(eventBus)
```



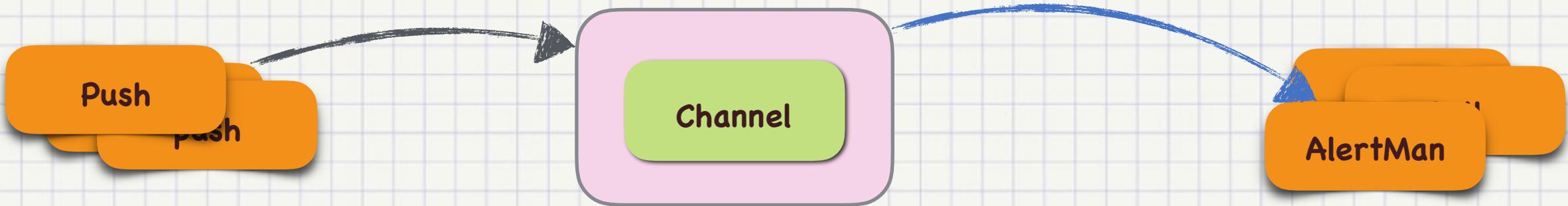
收集分布式日志



- 追踪所有调用链
- 计算耗时
- 排查问题



如何报警通知



⚠ 新告警通知

onealert ONEALERT

提醒 : refresh:engine loss

83

告警对象:

告警内容: {"extra_info": "post to engine failed"}

告警编号: 420

发生时间: 11:04

服务本身主动去上报异常！
异步去上报！
异常抽样报警！





状态监控页面

模块心跳

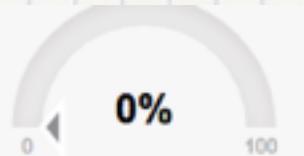
Module	Status	Diff	LastTime
poller	✓	2	2018-03-23 14:29:13
rester	✓	2	2018-03-23 14:29:13
scanner	✓	1	2018-03-23 14:29:14
node	■■■■■	2	2018-03-23 14:29:14



当前数值: 0
限频阈值: 50



当前数值: 0
限频阈值: 50



当前数值: 0
限频阈值: 50

节点心跳

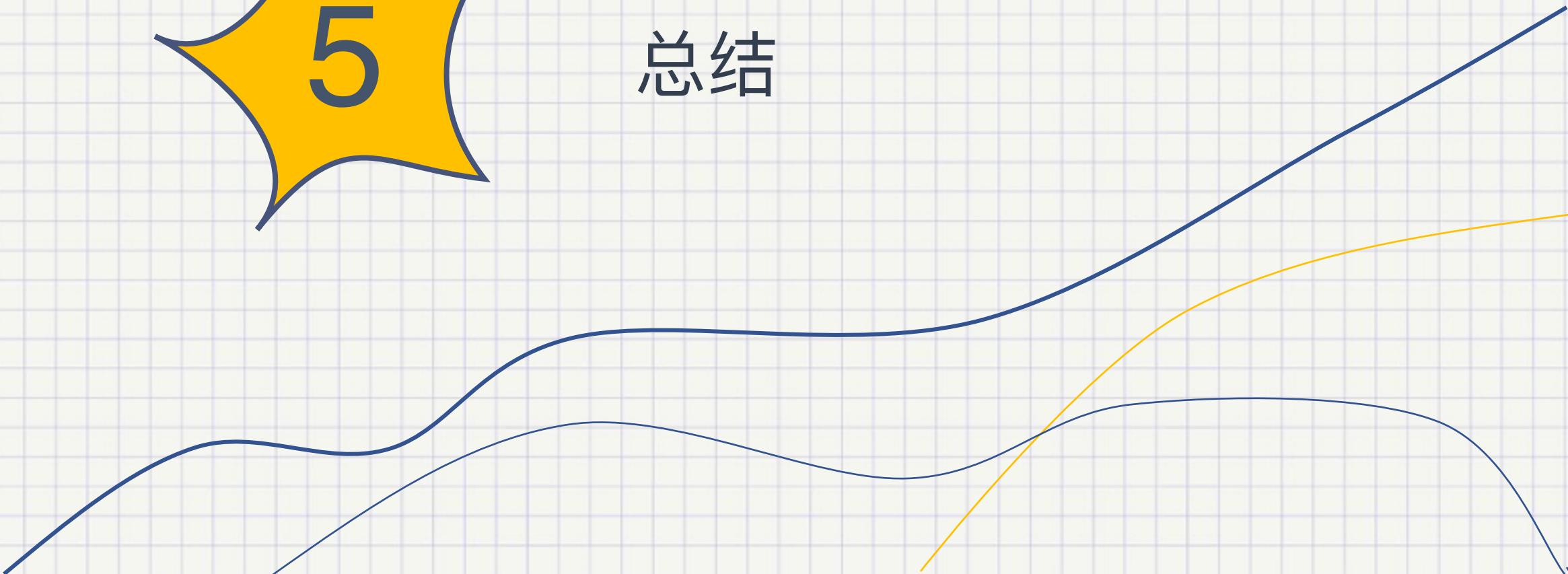
Module	Status	Diff	LastTir
node1	■■■■■	0	2018-03-23 14:29:13
node2	■■■■■	0	2018-03-23 14:29:13
node3	■■■■■	0	2018-03-23 14:29:13
node4	■■■■■	0	2018-03-23 14:29:13
node5	■■■■■	2	2018-03-23 14:29:14
node6	■■■■■	597	2018-03-23 14:29:14
node7	■■■■■	597	2018-03-23 14:29:14
node8	■■■■■	597	2018-03-23 14:29:14
node9	■■■■■	597	2018-03-23 14:29:14

API_ID计数器

API_ID	INIT	PENDING	RUNNING
2	0	0	0
3	0	0	0
4	0	0	0
5	0	0	0
6	0	0	0
7	0	0	0
8	0	0	0
9	0	0	0

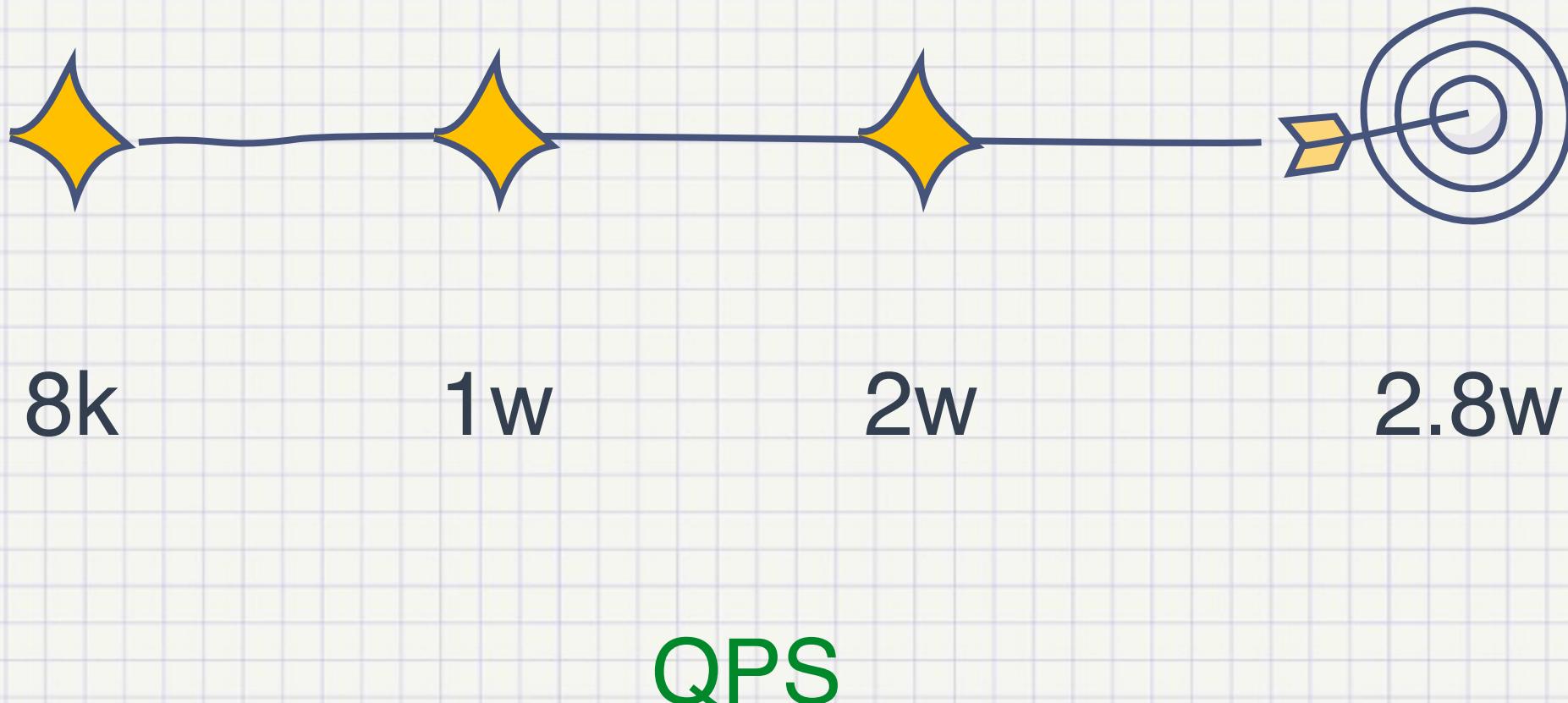


总结





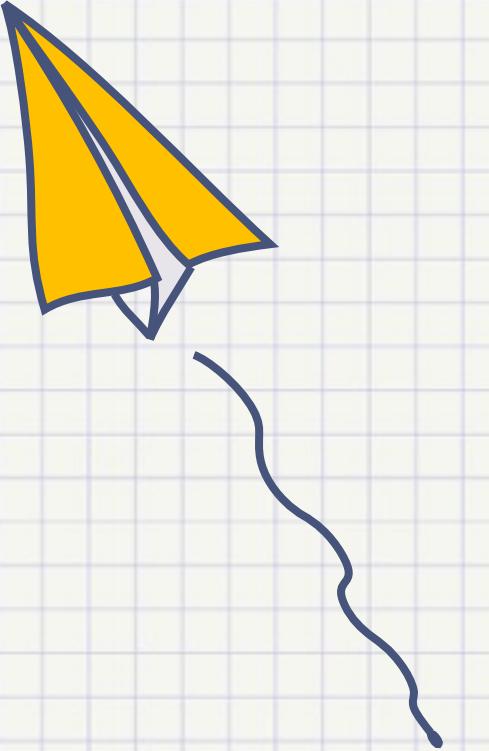
性能演进





GC 还需优化

Overhead	Shared Obj	Symbol
7.18%	engine_bin	[.] runtime.scanobject
4.70%	engine_bin	[.] runtime.mallocgc
2.70%	[kernel]	[k] iowrite16
2.63%	engine_bin	[.] runtime.greyobject
2.46%	engine_bin	[.] runtime.heapBitsForObject
1.75%	engine_bin	[.] runtime.heapBitsSetType
1.71%	engine_bin	[.] runtime.memmove
1.32%	engine_bin	[.] runtime.mapassign
1.13%	[kernel]	[k] __do_softirq
0.92%	engine_bin	[.] time.Time.AppendFormat
0.84%	engine_bin	[.] runtime.memclrNoHeapPointers
0.83%	engine_bin	[.] runtime.getitab
0.81%	engine_bin	[.] runtime.FastAlloc



Q & A

