企业级富容器技术 PouchContainer 详解

孙宏亮
阿里巴巴
2018.03.24

Agenda

- 阿里集团容器现状
- PouchContainer企业级技术优势
- PouchContainer的开源发展

PART 1

阿里集团容器现状

阿里集团容器现状

规模：

- 覆盖集团大部分BU
- 2017年双11百万级容器
- 在线业务100%容器化

覆盖场景：

- 运行模式
- 编程语言
- 技术栈

覆盖业务：

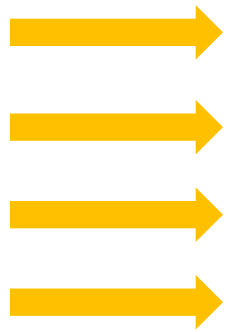
- 蚂蚁&交易&中间件
- B2B/CBU/ICBU/1688/村淘
- 合一集团（优酷）
- 菜鸟&高德&UC（接入中）
- 集团测试环境
- 广告（阿里妈妈）
- 阿里云专有云输出
- ……

阿里集团容器现状

- 本意育儿袋，隐喻贴身呵护应用
- 始于2011年
- 基于LXC
- 阿里内部容器技术产品，并于当年上线
- 2015年初开始吸收Docker镜像功能
- 容器结合阿里内核，大幅提高隔离性
- 大规模部署于阿里集团内部

PouchContainer演进之路

- 容器的要素--阿里内部运维和应用视角
  - 有独立IP
  - 能够ssh登陆
  - 独立的的文件系统
  - 资源隔离—使用量和可见性

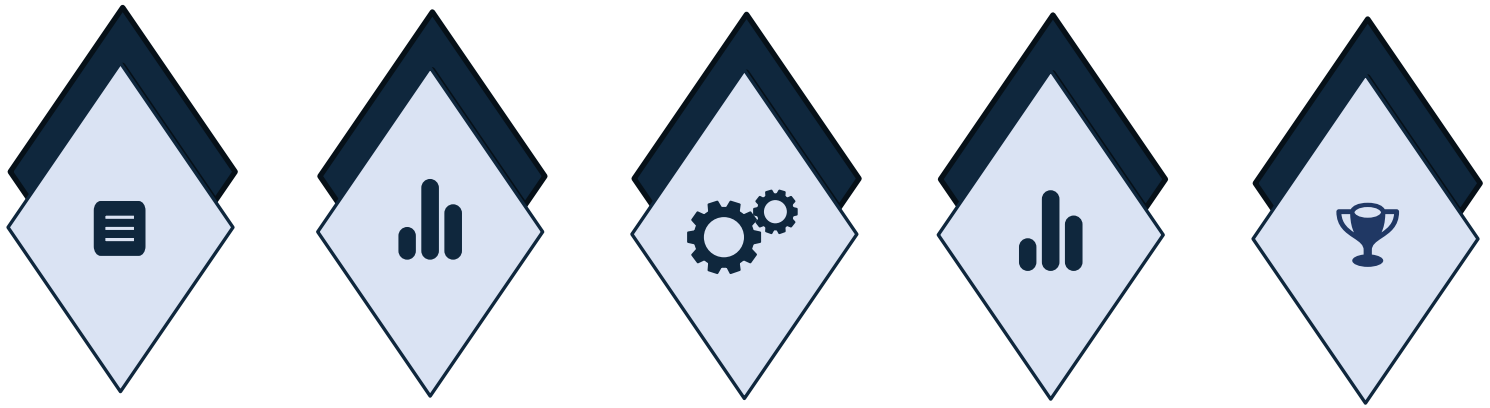


- 手工Hack实现容器要素
  - **虚拟网卡，网桥**
  - **sshd**
  - **Chroot (pivot_root)**
  - **CGroup，Namespace**

PART 2

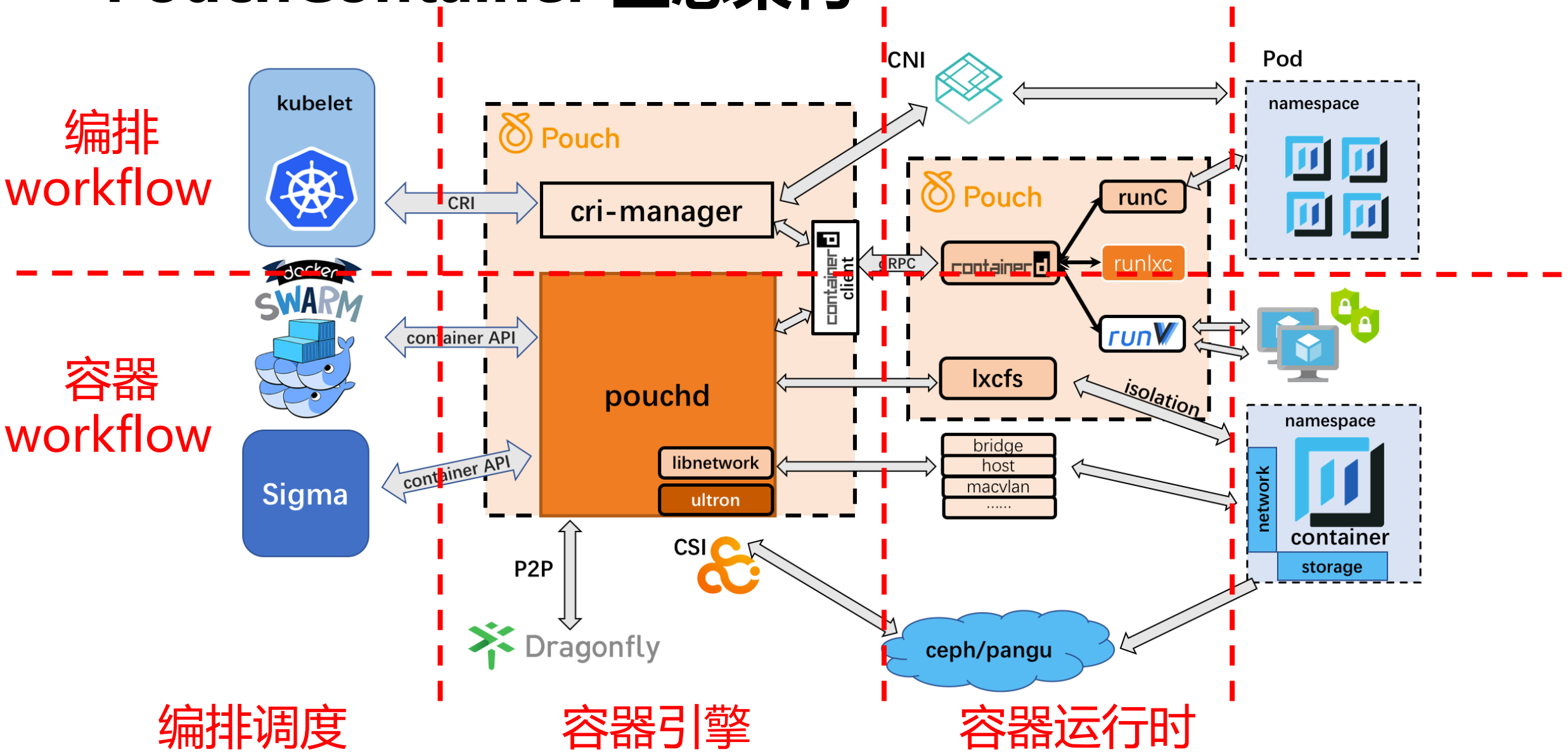
PouchContainer技术优势

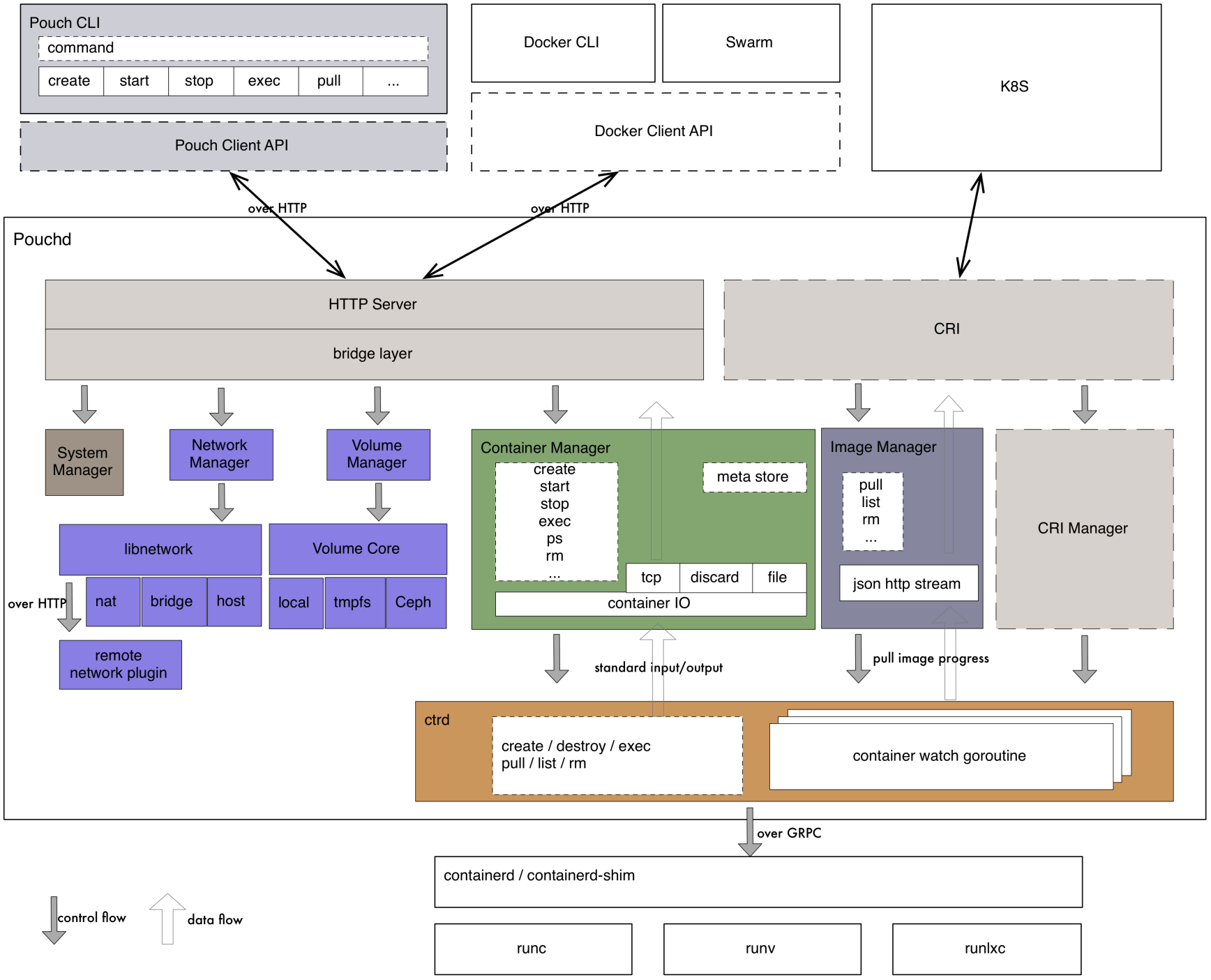
PouchContainer 技术优势



隔离性　P2P镜像分发　富容器　规模化考验　内核兼容性

PouchContainer 生态架构



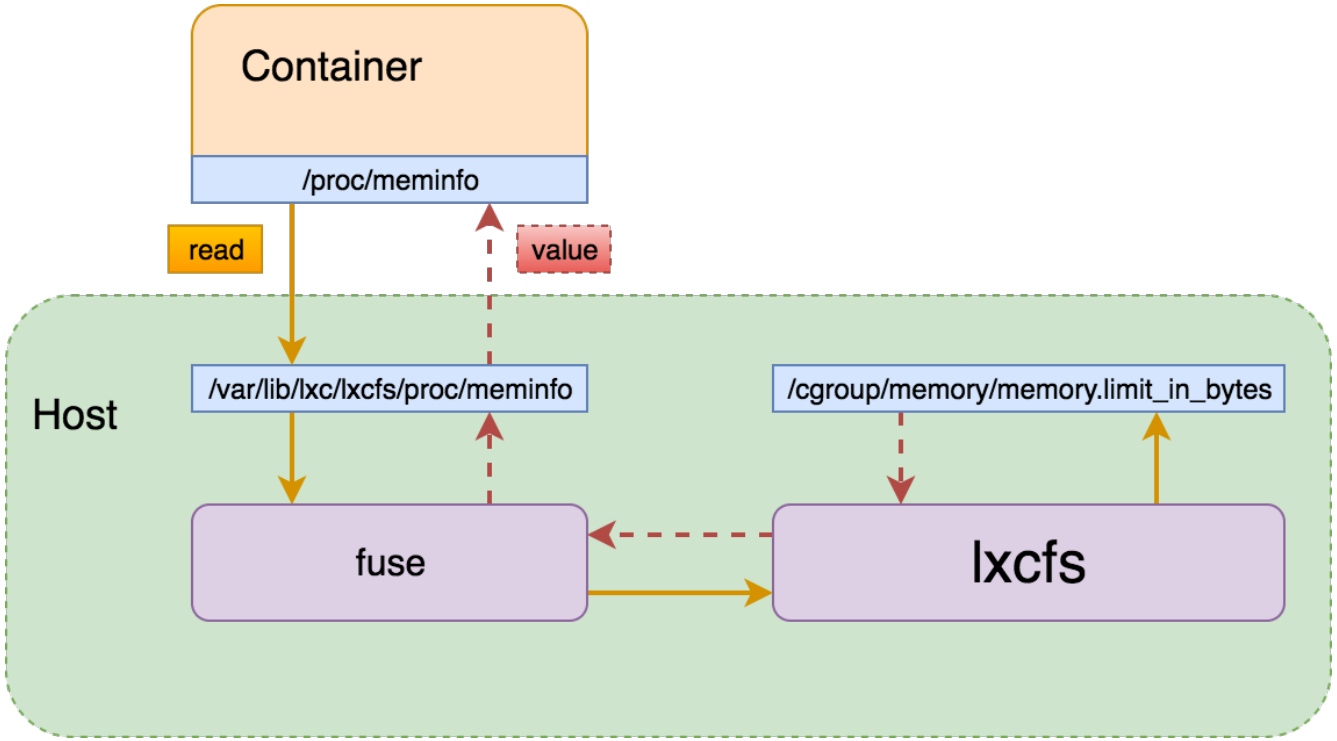

Pouch CLI
command
create
start
stop
exec
pull
...
Pouch Client API
Docker CLI
Swarm
Docker Client API
K8S
over HTTP
over HTTP
Pouchd
HTTP Server
bridge layer
CRI
System Manager
Network Manager
Volume Manager
Container Manager
create
start
stop
exec
ps
rm
...
meta store
tcp
discard
file
container IO
Image Manager
pull
list
rm
...
json http stream
CRI Manager
libnetwork
Volume Core
over HTTP
nat
bridge
host
local
tmpfs
Ceph
remote network plugin
standard input/output
pull image progress
ctrd
create / destroy / exec
pull / list / rm
container watch goroutine
over GRPC
containerd / containerd-shim
runc
runv
runlxc
control flow
data flow

丰富的隔离性

- 传统容器的隔离维度：namesapce，cgroup
- 更优的容器可见性隔离：内核patch，lxcfs
- 额外隔离维度：磁盘，网络等：diskquota
- 基于Hypervisor的强容器隔离
  - runV
  - clear container

资源可见性隔离 LXCFS

- 使用场景
  - Java应用判断资源大小动态分配堆栈大小，莫名OOM
  - Java中间件通过CPU核来创建线程数
  - /proc



资源可见性隔离 LXCFS

不使用LXCFS

```
$ pouch run -m 200m registry.hub.docker.com/library/ubuntu:16.04 free -h
              total        used        free      shared  buff/cache   available
Mem:           2.0G        103M        1.2G        3.3M        684M        1.7G
Swap:          2.0G          0B        2.0G
```

使用LXCFS

```
$ pouch run -m 200m --enableLxcfs registry.hub.docker.com/library/ubuntu:16.04 free -h
              total        used        free      shared  buff/cache   available
Mem:           200M        876K        199M        3.3M         12K        199M
Swap:          2.0G          0B        2.0G
```

https://github.com/alibaba/pouch/blob/master/docs/features/pouch_with_lxcfs.md

Diskquota容器磁盘限额

DiskQuota是一种限制文件系统磁盘空间使用的技术；

控制磁盘使用量的功能(Volume/容器rootfs)；

基于块设备的方式是可以直接控制磁盘的使用量（size/inode）；

DiskQuota功能在内核支持的版本情况：

	user/group quota	project quota
ext4	> 2.6	> 4.5
xfs	> 2.6	> 3.10

Diskquota容器磁盘限额

1. rootfs设置quota，通过--disk-quota的参数指定

```
# pouch run -ti --disk-quota 10g registry.hub.docker.com/library/busybox:latest df -h
Filesystem                Size      Used Available Use% Mounted on
overlay                  10.0G     24.0K     10.0G   0% /
tmpfs                    64.0M         0     64.0M   0% /dev
shm                      64.0M         0     64.0M   0% /dev/shm
tmpfs                    64.0M         0     64.0M   0% /run
tmpfs                    64.0M         0     64.0M   0% /proc/kcore
tmpfs                    64.0M         0     64.0M   0% /proc/timer_list
tmpfs                    64.0M         0     64.0M   0% /proc/sched_debug
tmpfs                     1.9G         0      1.9G   0% /sys/firmware
tmpfs                     1.9G         0      1.9G   0% /proc/scsi
```

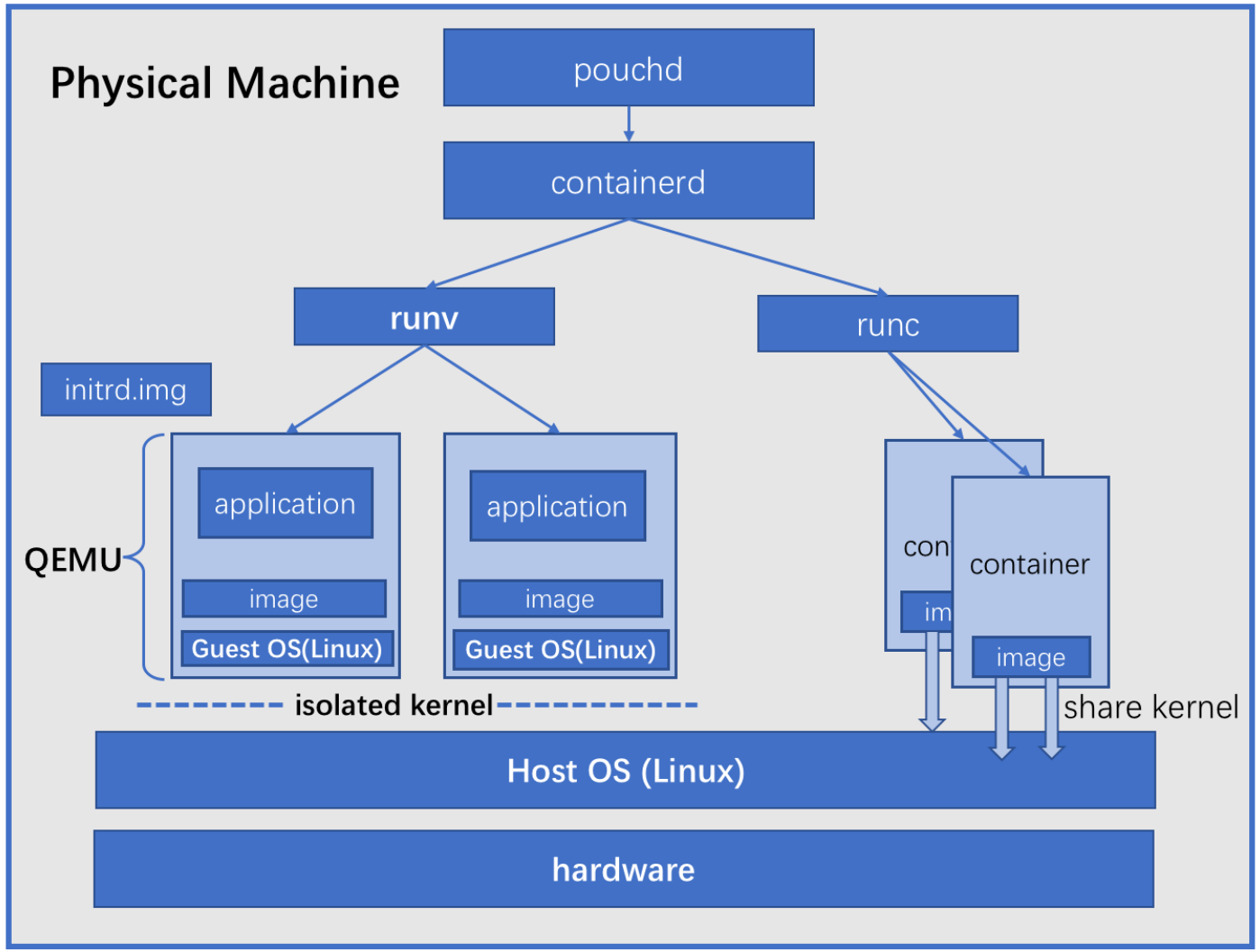
Diskquota容器磁盘限额

2. volume设置quota，通过设置volume size参数指定

```
# pouch volume create -n volume-quota-test -d local -o mount=/data/volume -o size=10g
Name:         volume-quota-test
Scope:
Status:       map[mount:/data/volume sifter:Default size:10g]
CreatedAt:    2018-3-24 13:35:08
Driver:       local
Labels:       map[]
Mountpoint:   /data/volume/volume-quota-test

# pouch run -ti -v volume-quota-test:/mnt registry.hub.docker.com/library/busybox:latest df -h
Filesystem                Size      Used Available Use% Mounted on
overlay                  20.9G    212.9M     19.6G   1% /
tmpfs                    64.0M         0     64.0M   0% /dev
shm                      64.0M         0     64.0M   0% /dev/shm
tmpfs                    64.0M         0     64.0M   0% /run
/dev/sdb2                10.0G      4.0K     10.0G   0% /mnt
tmpfs                    64.0M         0     64.0M   0% /proc/kcore
tmpfs                    64.0M         0     64.0M   0% /proc/timer_list
tmpfs                    64.0M         0     64.0M   0% /proc/sched_debug
tmpfs                     1.9G         0      1.9G   0% /sys/firmware
tmpfs                     1.9G         0      1.9G   0% /proc/scsi
```

Hypervisor-based Container



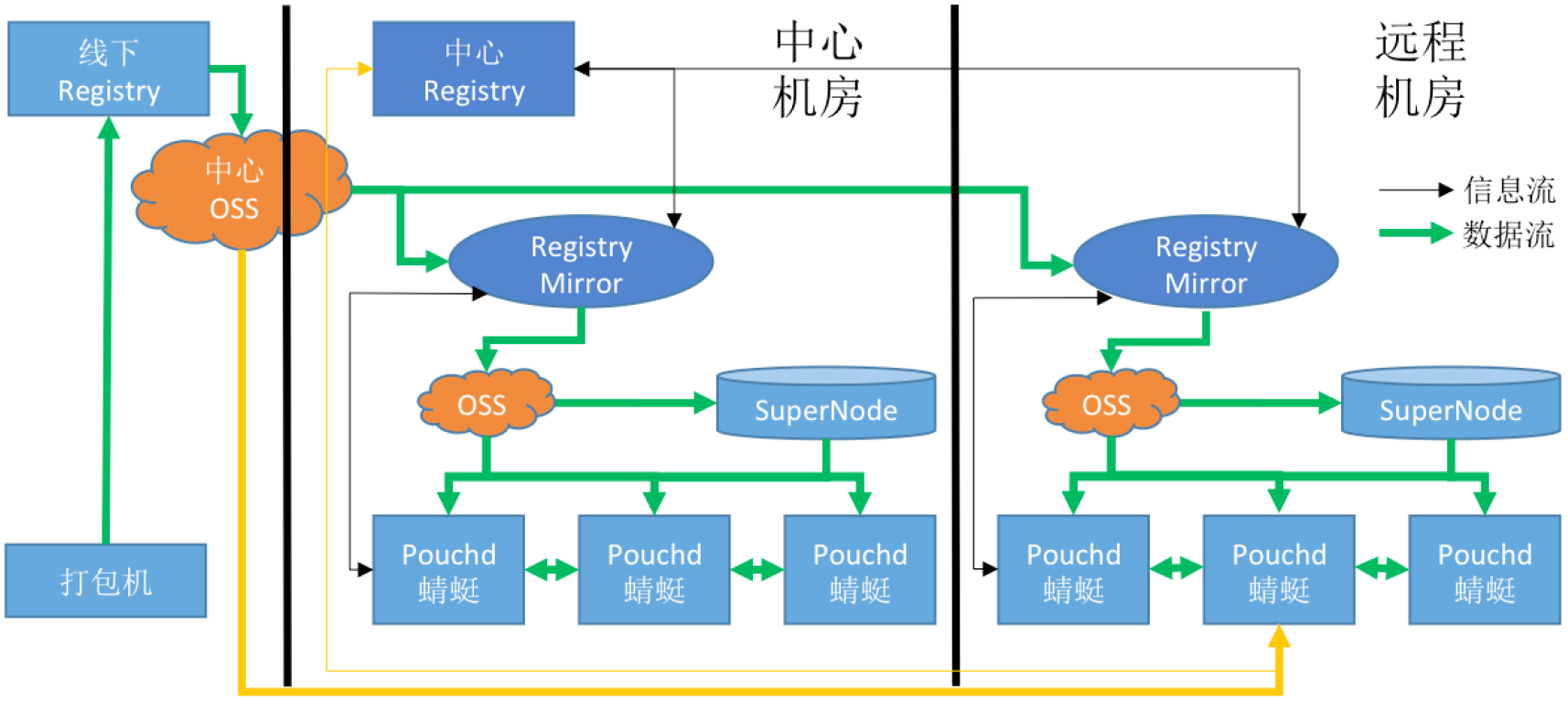
Hypervisor-based Container

```
$ pouch create --name hypervisor --runtime runv docker.io/library/busybox:latest
container ID: 95c8d52154515e58ab267f3c33ef74ff84c901ad77ab18ee6428a1ffac12400d, name: hypervisor
$
$ pouch ps
Name          ID       Status    Image                              Runtime
hypervisor    95c8d5   created   docker.io/library/busybox:latest   runv
4945c0        4945c0   stopped   docker.io/library/busybox:latest   runc
1dad17        1dad17   stopped   docker.io/library/busybox:latest   runv
fab7ef        fab7ef   created   docker.io/library/busybox:latest   runv
505571        505571   stopped   docker.io/library/busybox:latest   runc
```

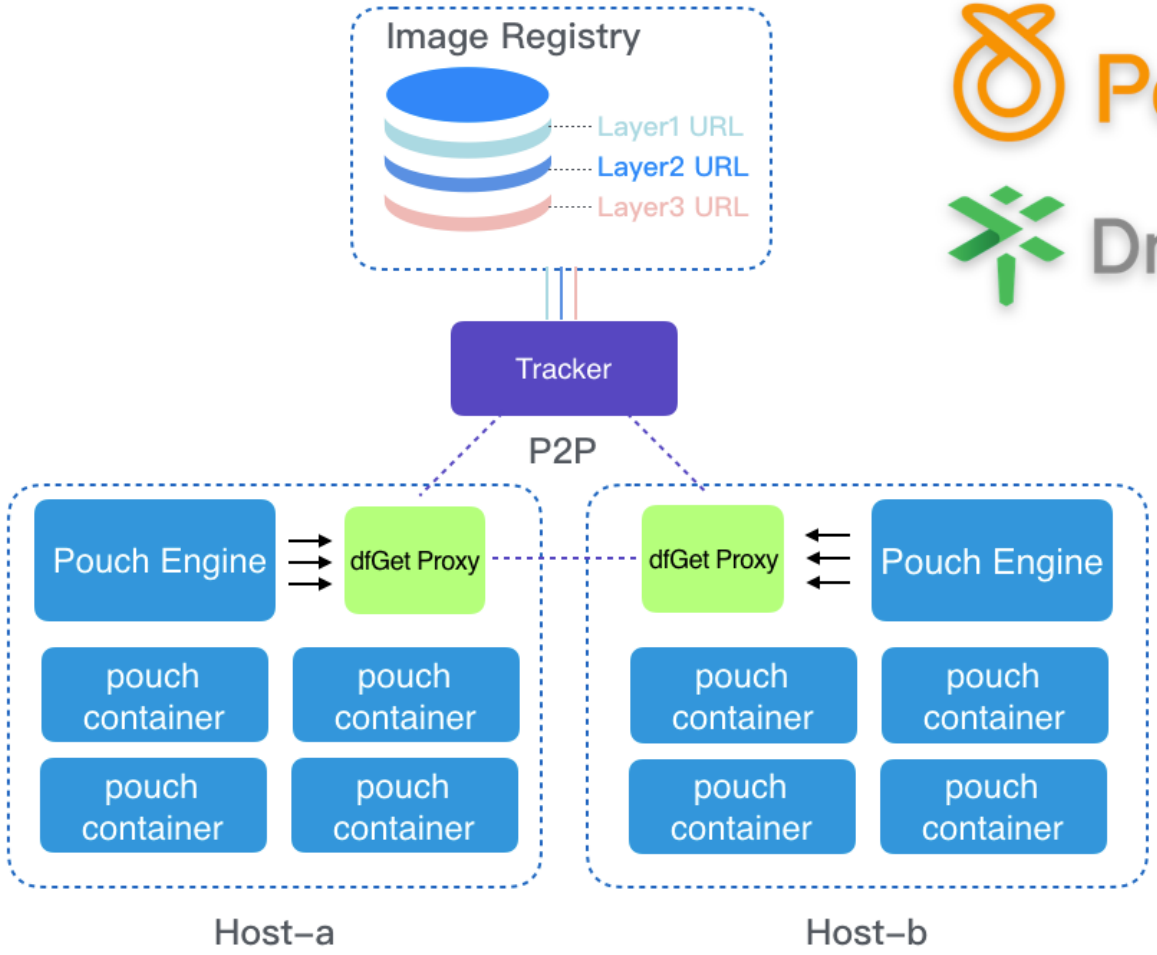
https://github.com/alibaba/pouch/blob/master/docs/features/pouch_with_runV.md

P2P镜像分发能力

Pouch Dragonfly



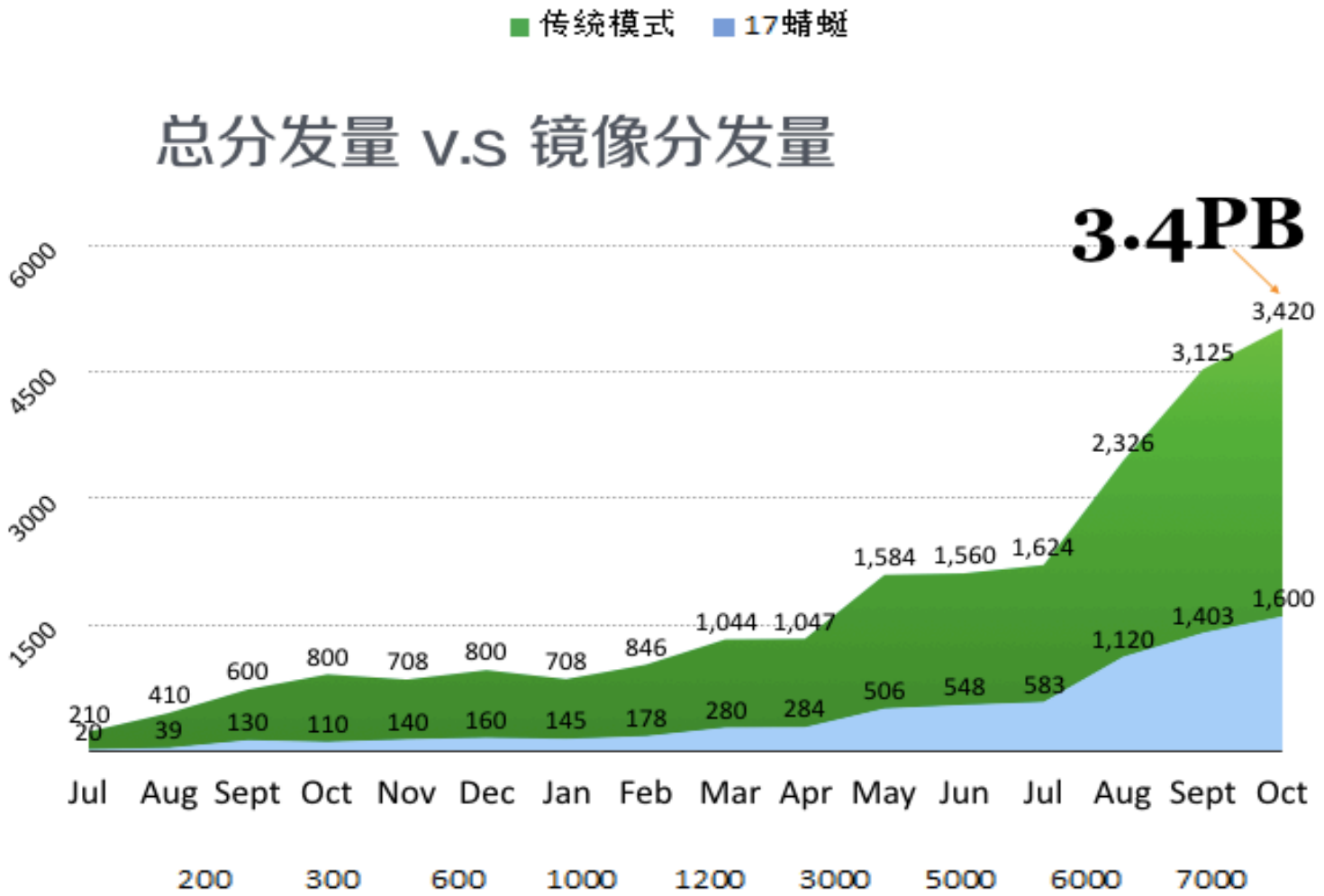
P2P镜像分发能力



Pouch

Dragonfly

P2P镜像分发能力


传统模式
17蜻蜓
总分发量 v.s 镜像分发量
3.4PB
6000
4500
3000
1500
210
410
600
800
708
800
708
846
1,044
1,047
1,584
1,560
1,624
2,326
3,125
3,420
20
39
130
110
140
160
145
178
280
284
506
548
583
1,120
1,403
1,600
Jul
Aug
Sept
Oct
Nov
Dec
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sept
Oct
200
300
600
1000
1200
3000
5000
6000
7000

富容器

- 容器内运行init进程，PID = 1
- 容器内运行系统服务
- 用户体验如传统 VM
- 极强的应用适配性
- 集团应用100%容器化的重要前提
- 容器内资源多维度隔离（alikernel支持）

https://github.com/alibaba/pouch/blob/master/docs/features/pouch_with_rich_container.md

规模化考验

- 阿里巴巴集团内部大规模场景验证
- 绝大部分BU
- 2016年双11，几十万容器应对超大规模负载
- 2017年容器常规规模达到数十万
- 2017年双11，容器规模达到百万级
- 混部支持

内核兼容性

- 阿里仍存有相当规模的 Linux 2.6.x 内核机器
- 规模效应，放大现有老系统的资产价值
- PouchContainer支持内部所有 Linux 2.6.x 的内核
- 部分支持来源指定系统调用的回避
- 部分支持来源内核补丁

PART 3

PouchContainer开源发展

https://github.com/alibaba/pouch

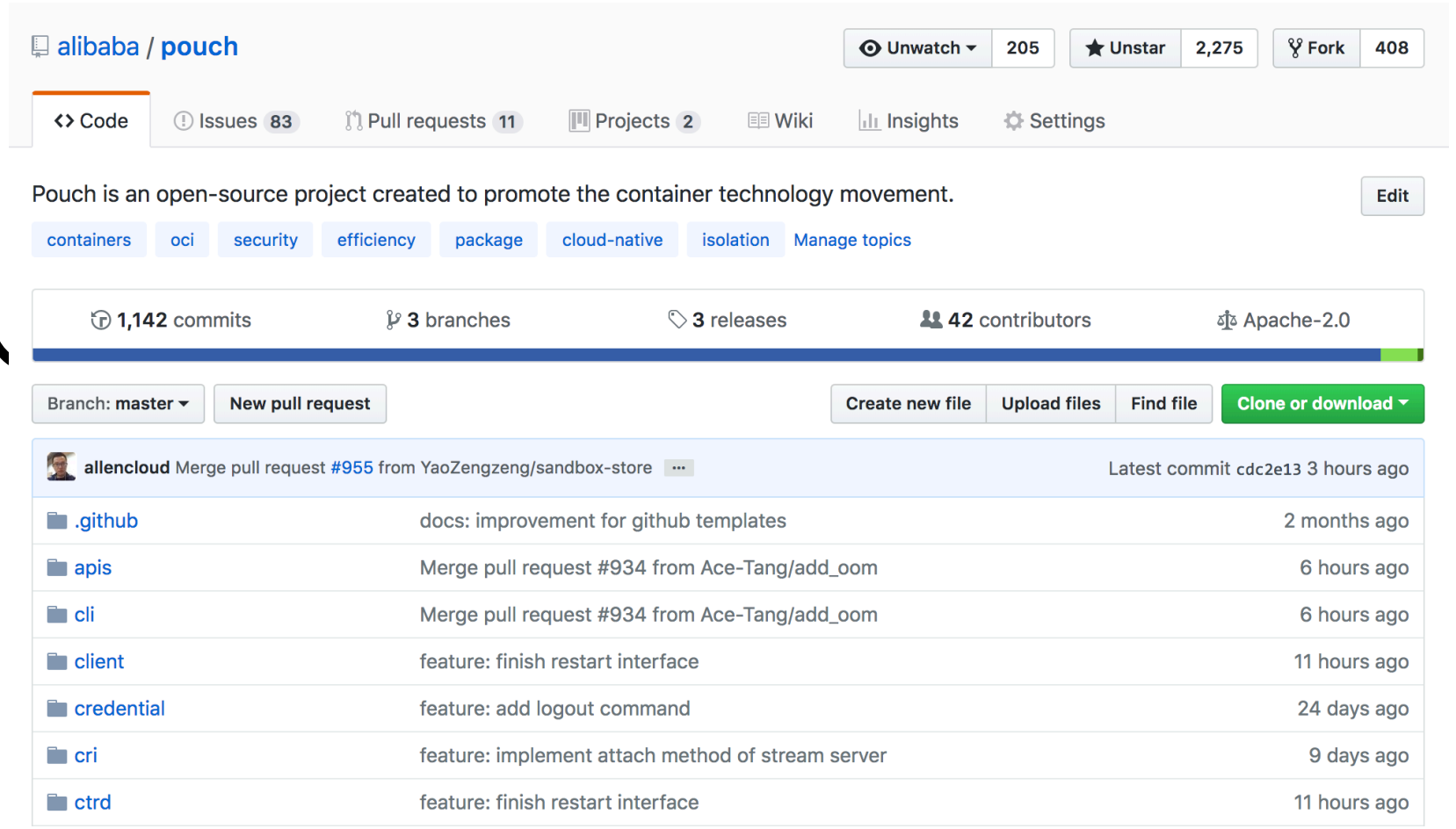
2275 star

43位贡献者

1位协作机器人

文档

测试



安装指南： https://github.com/alibaba/pouch/blob/master/INSTALLATION.md

pouchrobot

Label标签

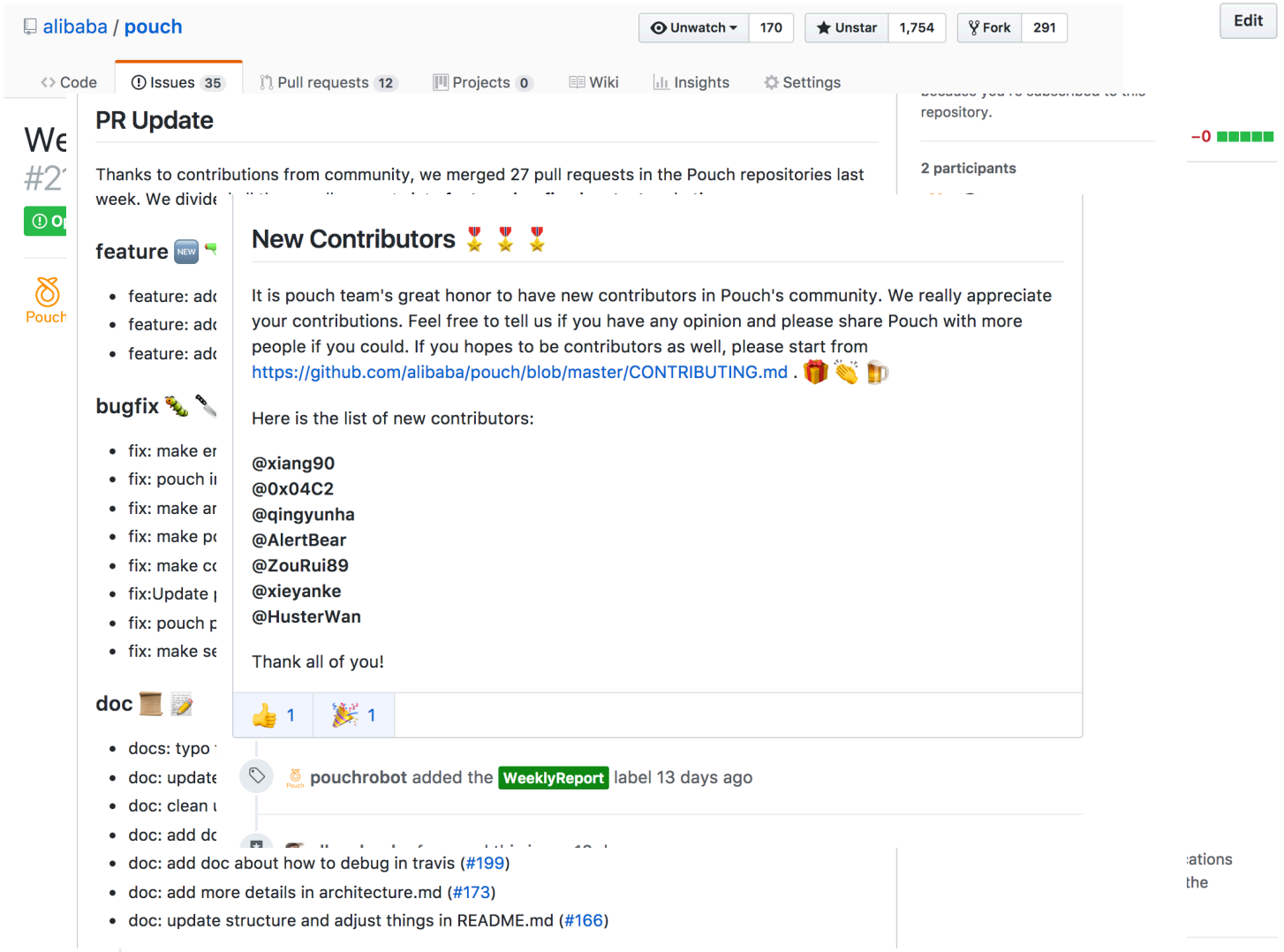
冲突检测

周报生成

文档生成

CI通知集成

分布式协作效率



如何参与 Pouch

- 在你的组织中使用Pouch
- 布道与宣传
- 贡献回你的bug修复、功能扩展以及文档
- 日常贡献 -> maintainer
- 说服你的朋友贡献新科技
- https://github.com/alibaba/pouch/blob/master/CONTRIBUTING.md

Q&A

We are hiring!

Email: allensun.shl@alibaba-inc.com
WeChat: shlallen

Thank You