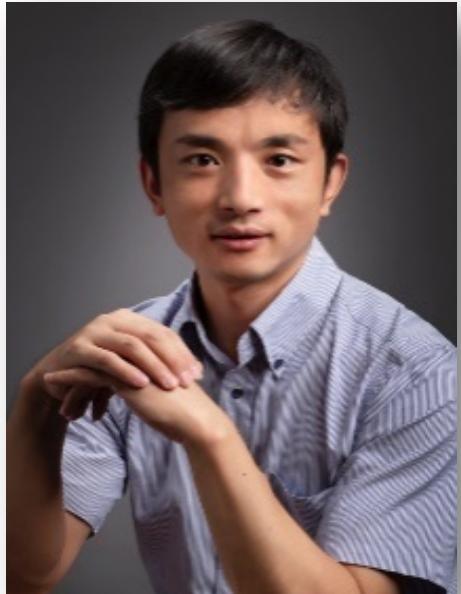


# Harbor-助你玩转云原生

Steven Zou (邹佳), VMware主任工程师/Harbor 核心维护者和架构师

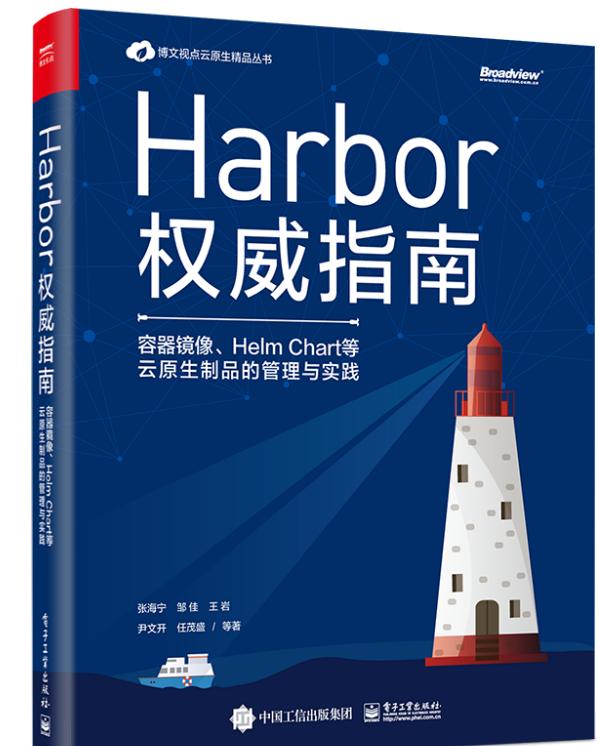


# 关于我



Steven(佳) Zou(邹)，VMware中国研发中心主任工程师，Harbor开源项目架构师及核心维护者，拥有十多年软件研发及架构经验，获得PMP资格认证及多项技术专利授权。曾在HPE、IBM等多家企业担任资深软件工程师和架构师，专注于云计算及云原生等相关领域的研究与创新。著有《Harbor权威指南》等书籍。

>> Email: [szou@vmware.com](mailto:szou@vmware.com)  
>> GitHub ID: [steven-zou](#)  
>> Slack: [steven zou](#)



# 目录



*Virtual*

- 开场：云原生与制品管理
- 初识Harbor：云原生制品仓库服务
- 使用Harbor搭建私有制品仓库服务
- 核心功能：
  - 资源隔离与多租户管理模型
  - 制品的高效分发(复制、缓存与P2P集成)
  - 制品的安全分发(签名、漏洞扫描与安全策略)
  - 资源清理与垃圾回收
- 构建高可用(HA)制品仓库服务
- Harbor集成与扩展
- 路线图
- 参与贡献Harbor社区

# 云原生与制品管理 [1]

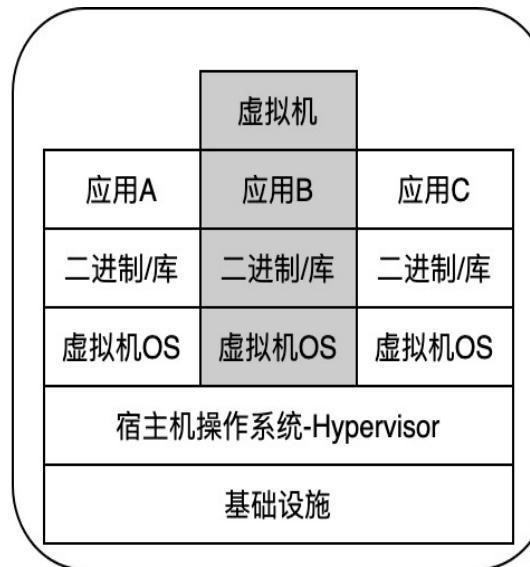
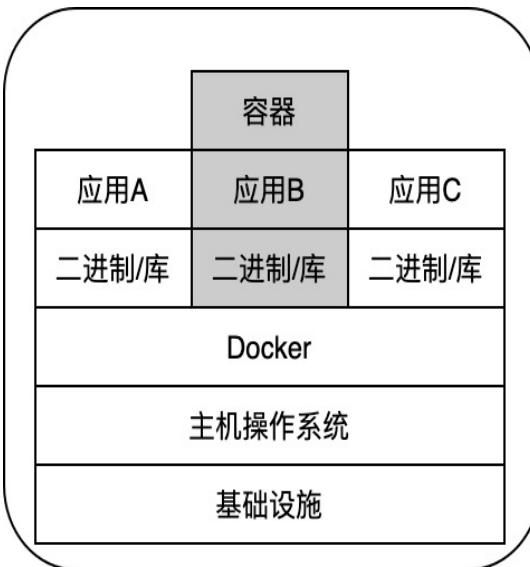


Virtual

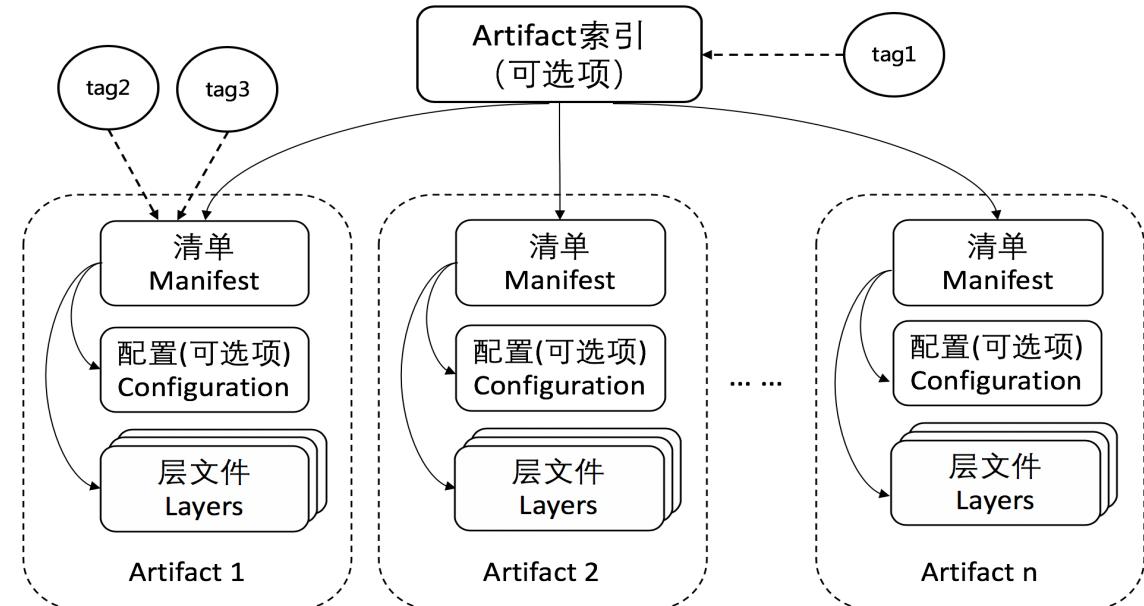
云原生(cloud-native)技术使组织能够在现代化和动态的环境下(如公有云、私有云和混合云)构建和运行可扩展的应用程序。云原生典型技术包括容器、服务网络、微服务、不可变基础设施和声明性API等。

v1.0 by CNCF

## 容器-更轻量级和灵活的虚拟化



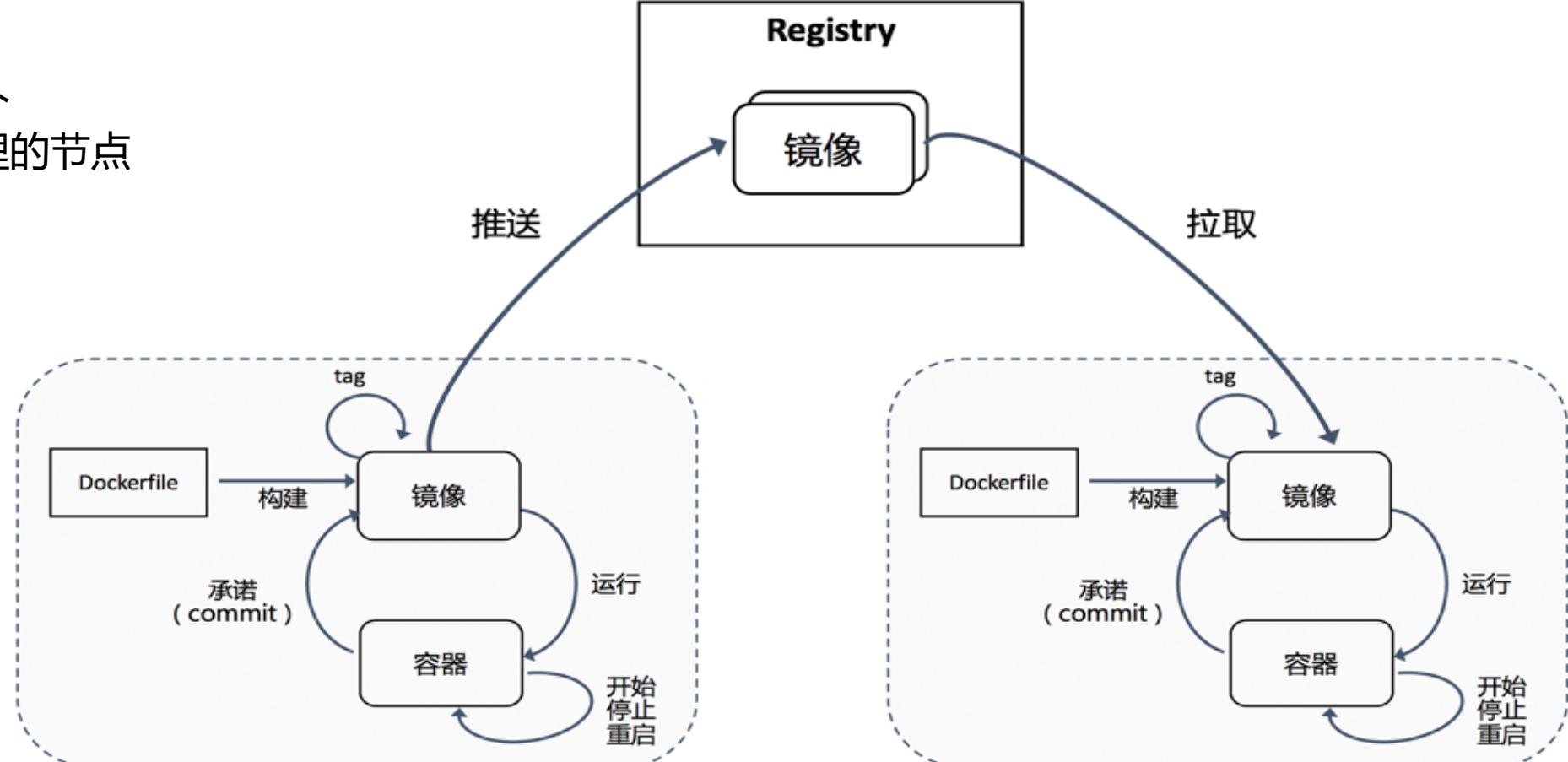
## 镜像-应用软件打包与分发



# 云原生与制品管理 [2]

## Registry:

- 制品存储仓库
- 分发制品的媒介
- 访问控制与管理的节点



# 初识Harbor [1]



Virtual



一个开源可信的云原生制品仓库项目用来存储、签名和管理相关内容。

官方网站：[goharbor.io](https://goharbor.io)



CNCF毕业项目



落地在很多企业级  
产品中



Apache 2.0协议下  
开源



GitHub代码库：  
<https://github.com/goharbor/harbor/>

# 初识Harbor [2] – 社区



Virtual

GitHub星

14.3K+

核心提交者

200+

贡献公司

50+

贡献者

3000+

Fork

3.6K+

有来自于5家公司的14位维护者



提交数

10K+

GitHub访问/访问者\*

61K+/13K+

下载\*

5K+

Jan 31, 2016 – Mar 17, 2021

Contributions to master, excluding merge commits

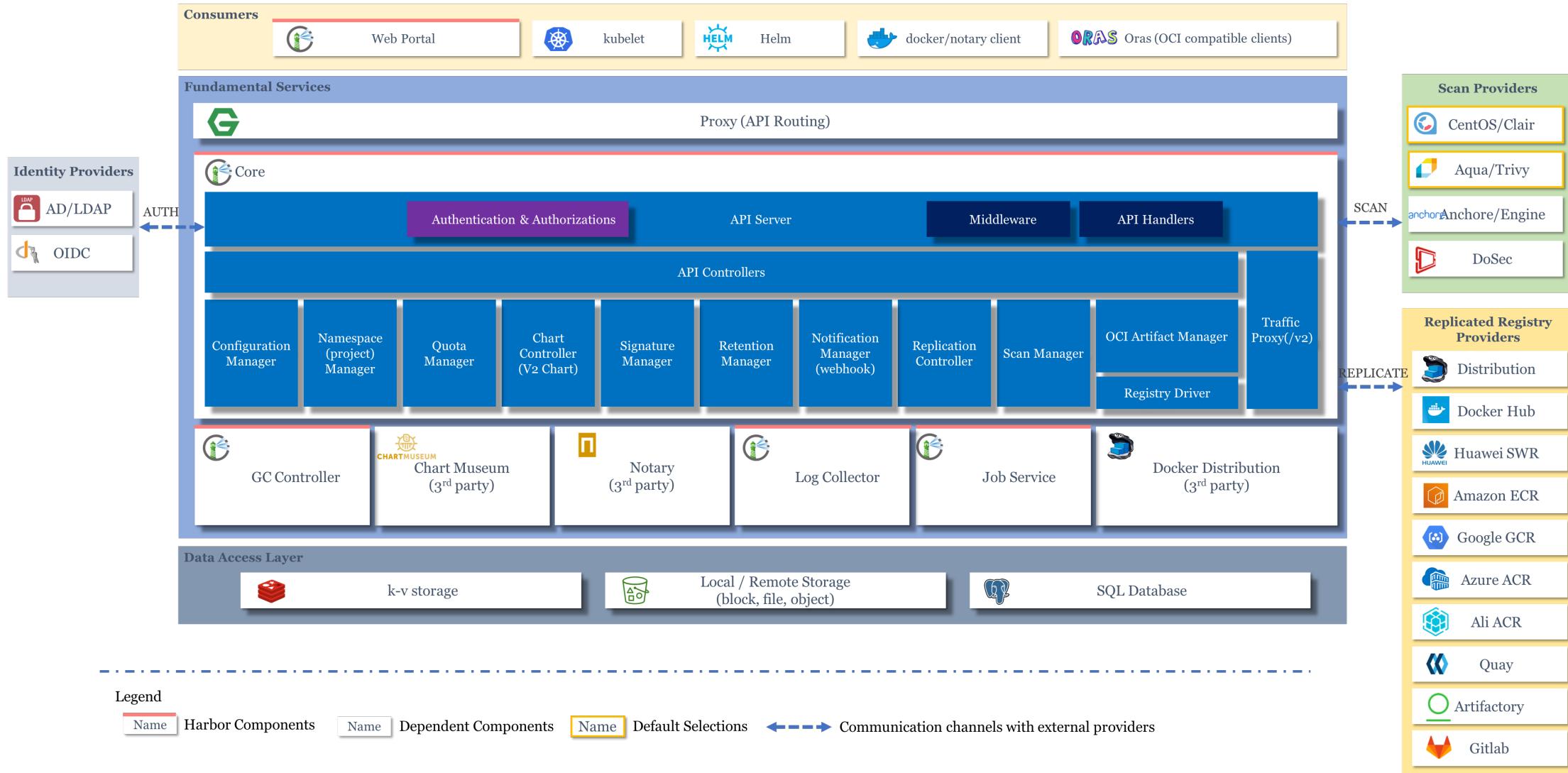


Harbor社区

# 初识Harbor [3] – 整体架构



Virtual



# 初识Harbor [4] – 功能

Harbor

系统

系统设置 (鉴权模式等)

垃圾回收(GC)

内容复制

配额管理

扫描管理

用户管理

P2P预热管理

系统标签管理

系统级日志

系统监控

项目1

制品管理

访问控制(RBAC)

Tag清理策略

Tag不可变策略

P2P预热策略

缓存策略

机器人账户

Webhooks

项目配置

项目标签管理

项目扫描器设置

项目级日志

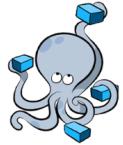
...

项目N

# 搭建Harbor仓库服务



Virtual



## 离线安装包

- 通过Docker-compose编排运行
- 所需镜像皆打包在离线包内

1



## 在线安装包

- 通过Docker-compose编排运行
- 所需镜像从Dockerhub来拉取

2



## Helm Chart

- 通过Helm来安装
- 目标为K8s集群
- 仅聚焦Harbor组件安装
- goharbor/harbor-helm

3

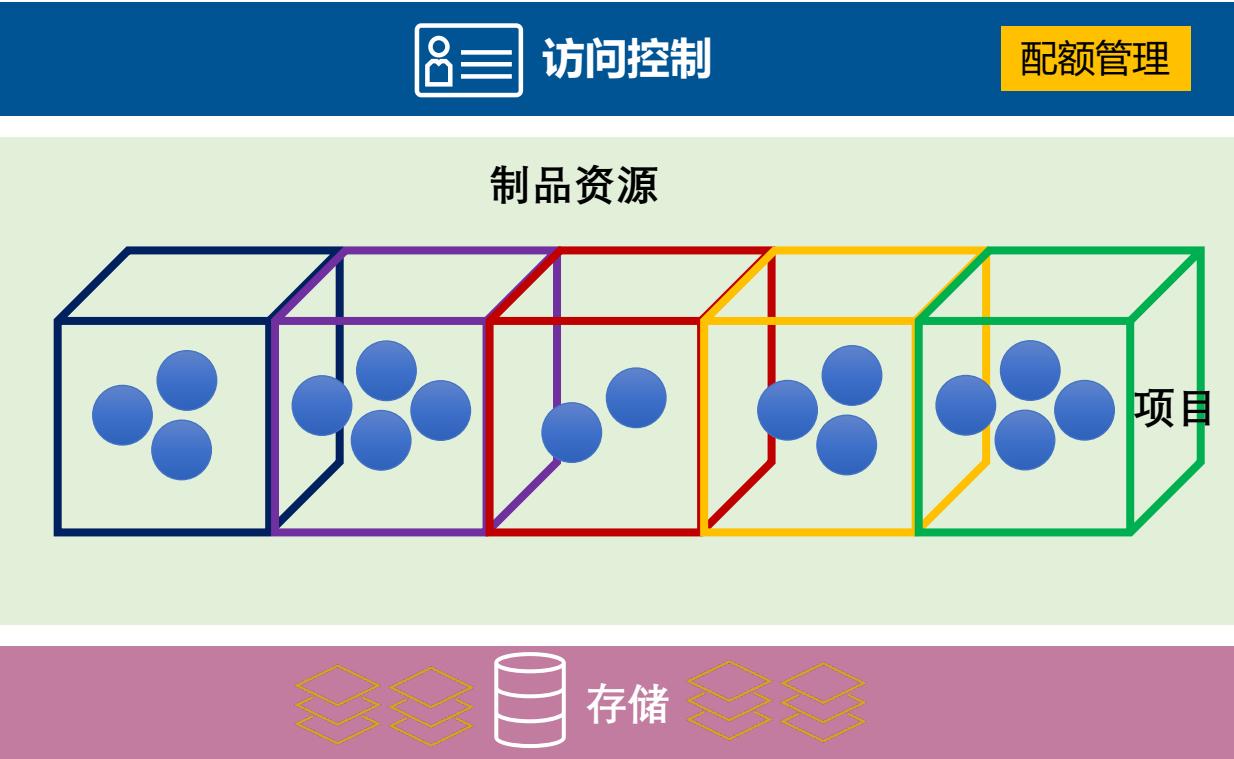


## K8s Operator

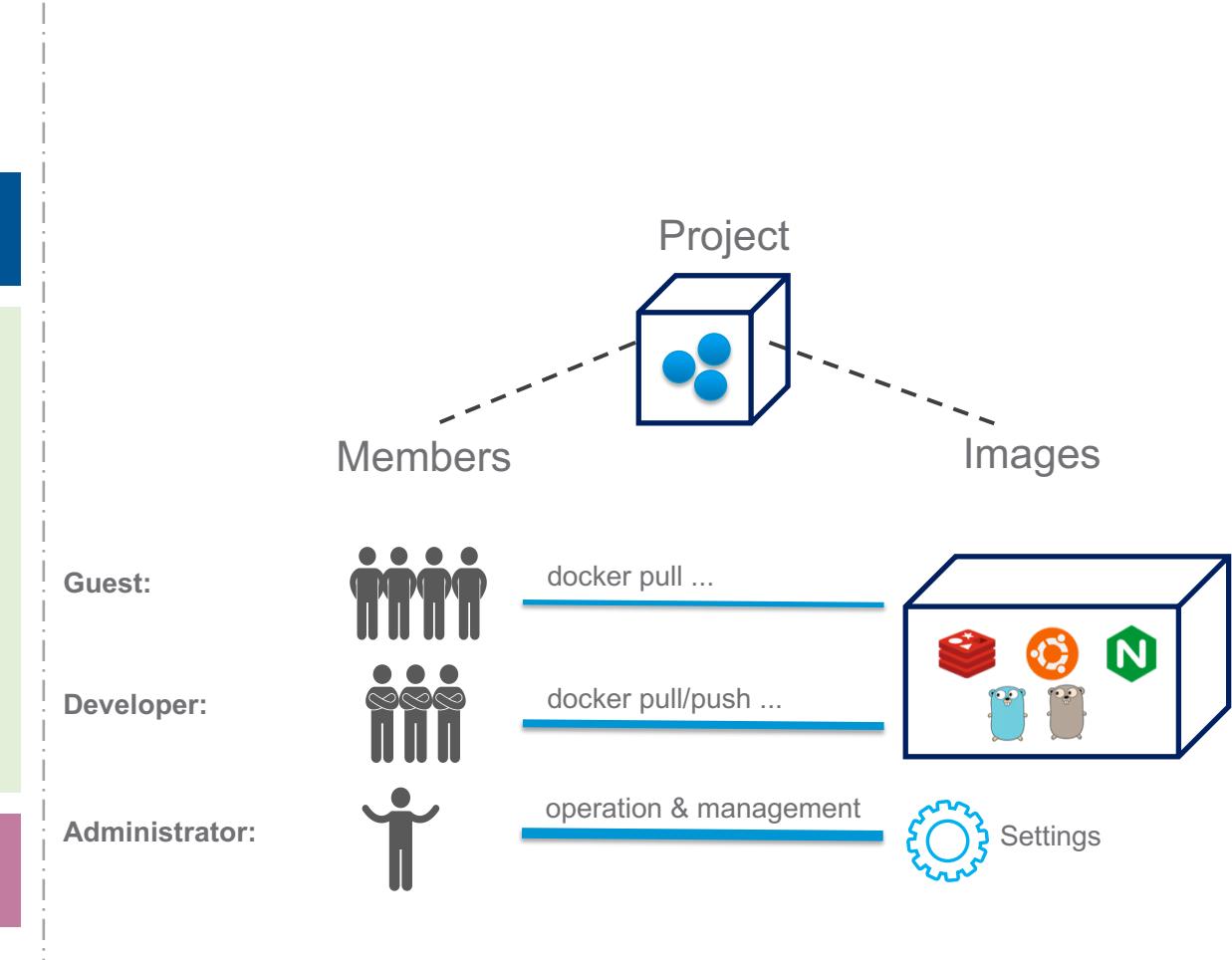
- 通过K8s CRD实现编排
- 目标为K8s集群
- 专注于HA模式支持
- goharbor/harbor-operator (开发中)

4

# 资源隔离与多租户管理

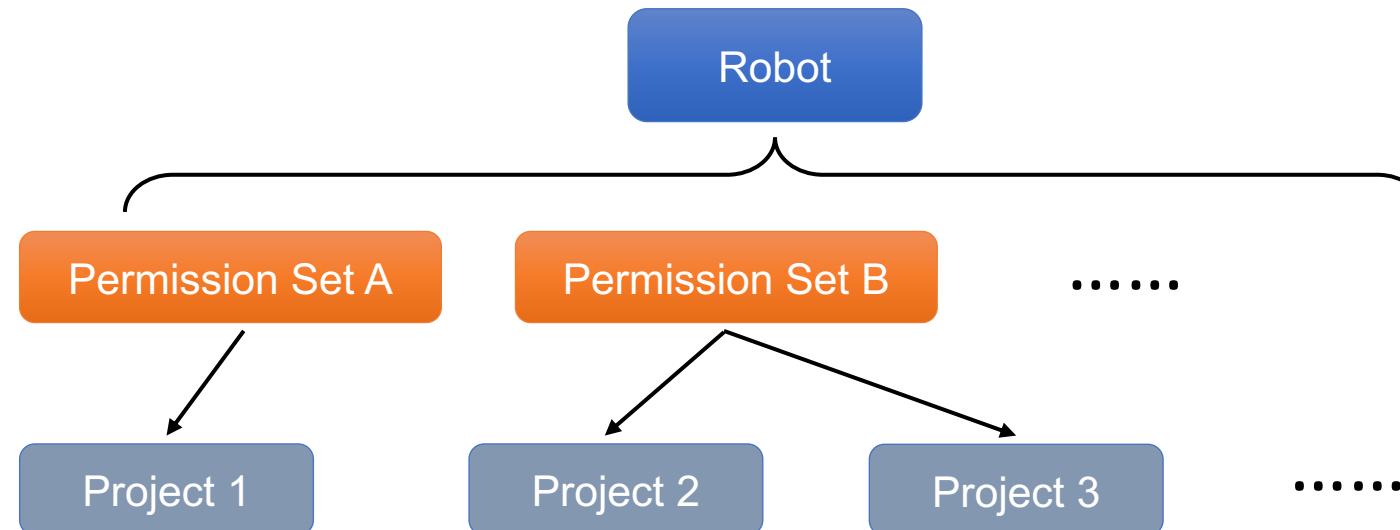


提供以项目为单位的逻辑隔离，存储共享



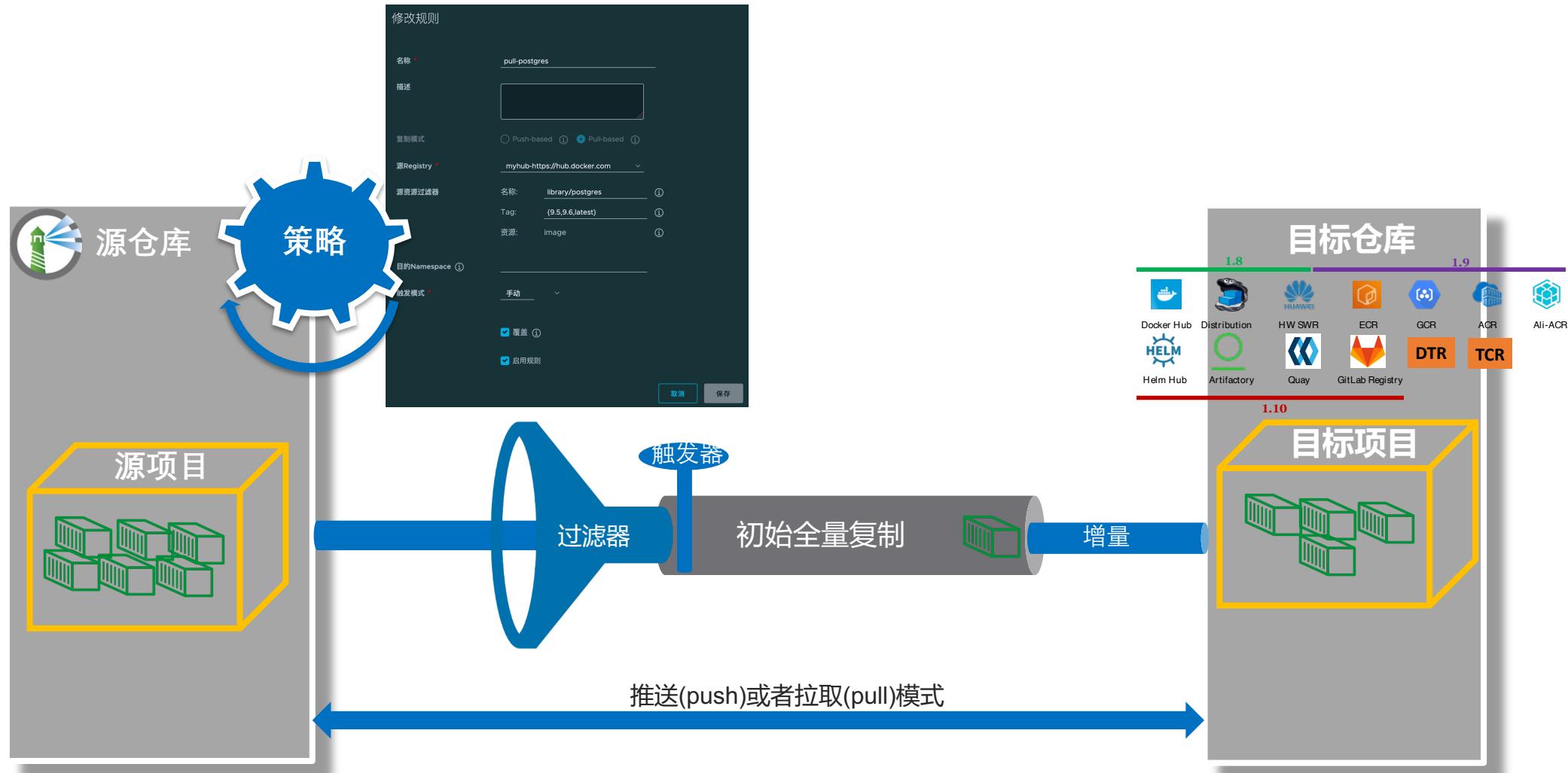
不同角色具有不同的访问权限，可以与其它用户系统集成

- 实现系统级别的机器人账户以支持通过一组授权来覆盖多个项目
- 支持更多类型的权限来授权不同的操作和API调用
- 相对于之前的jwt令牌方式，新的方式带来更大的灵活性(即使授权范围或者过期时间发生变化，密码依然有效)

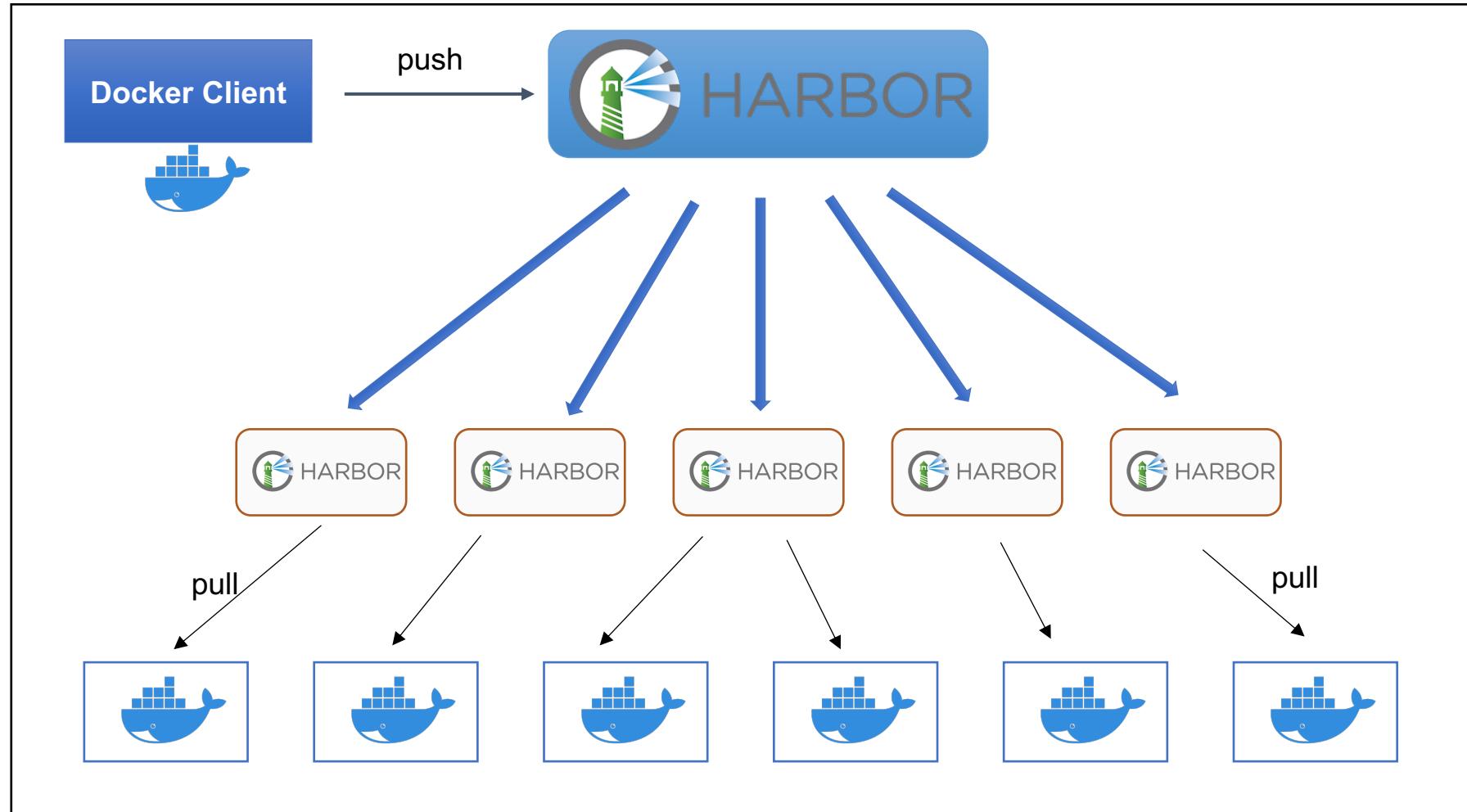


# 制品的高效分发-复制 [1]

基于策略的内容复制机制：支持多种过滤器(镜像库、 tag和标签) 与多种触发模式 (手动， 基于时间以及定时) 且实现对推送和拉取模式的支持



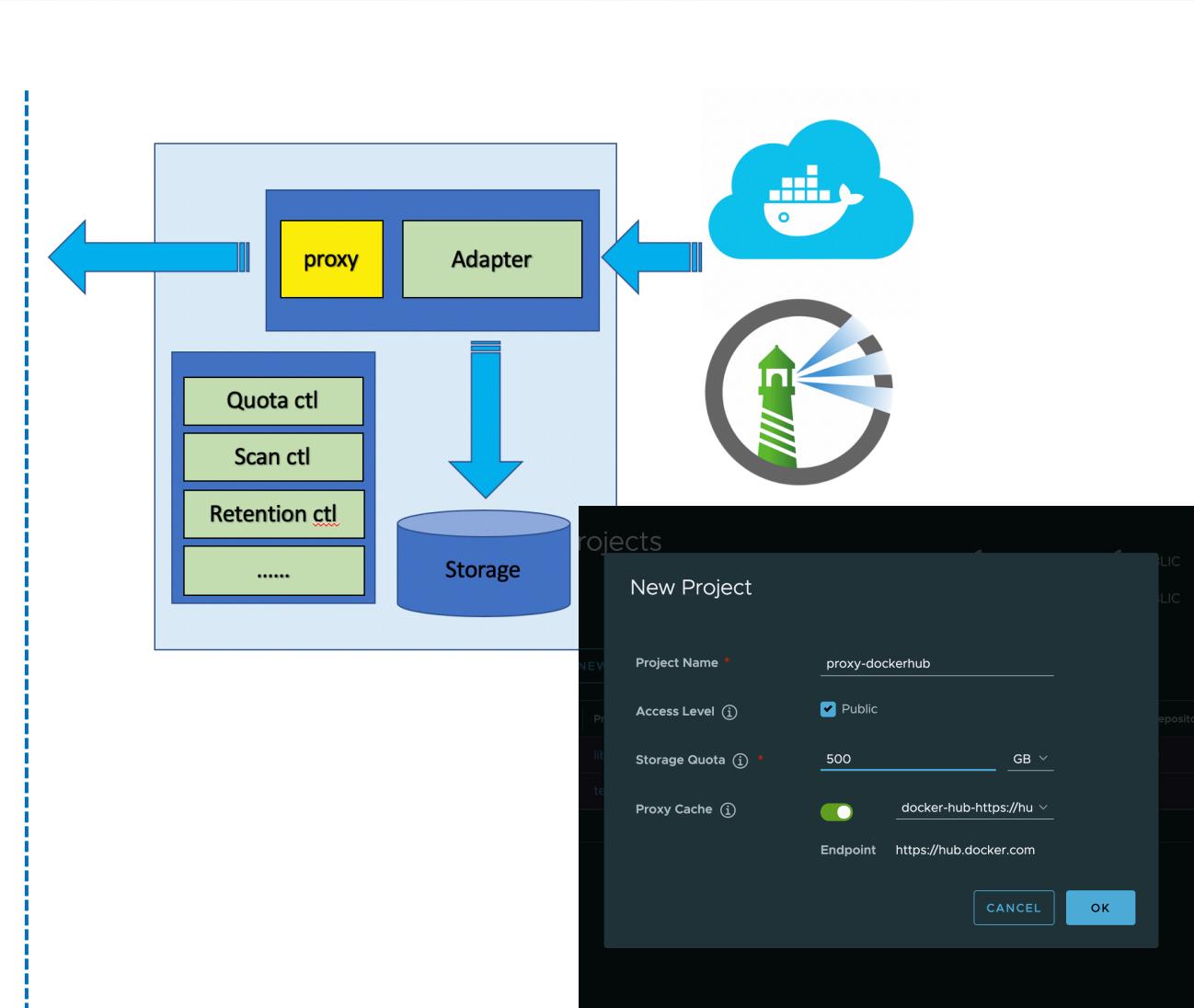
# 制品的高效分发-复制 [2]



主从模式(中心-边缘模式)

# 制品的高效分发-缓存

- 在项目级别提供“缓存”能力
- 已缓存下来的制品与“本地”制品无异
- 相关的管理策略可以应用到缓存的镜像上，比如配额、扫描等
- 目前仅支持上游Dockerhub\*和其它Harbor，更多上游仓库的支持正在进行中（与复制共享同样的仓库适配器）
- 注意：在创建项目时选择启用缓存功能则新建项目为缓存项目，不可推送；以创建的普通项目无法直接转为缓存项目
- Pull路径: docker pull <harbor-host>/[cache\_project\_name]/<repository\_path> ([docker pull goharbor.io/my\\_cache\\_pro/library/nginx:latest](https://goharbor.io/my_cache_pro/library/nginx:latest))



注: 可有效解决Dockerhub 限速的问题

# 制品的高效分发-P2P预热

- 将所选镜像提前分发到(加热)P2P网络以便客户端拉取内容时从P2P网络直接获得
- 基于策略实现自动化
  - Repository过滤器
  - Tag过滤器
  - 标签 (label) 过滤器
  - 漏洞状态条件
  - 签名状态条件
- 基于事件触发或者定时触发
- 支持P2P引擎：
  - Dragonfly
  - Kraken

Edit P2P Provider Policy

Provider: dragonfly mydragonfly-<http://47.252.100.100>

Name: my-preheat

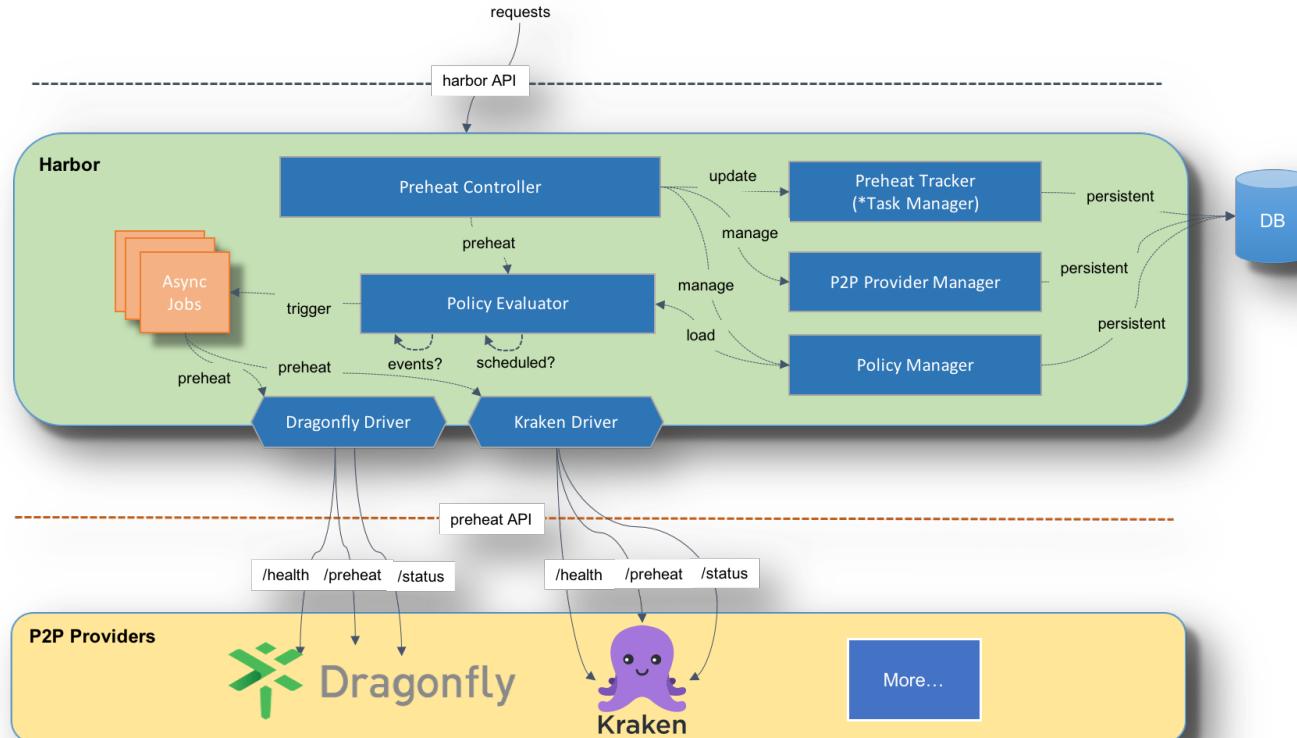
Description:

Filters: Repositories: Enter multiple comma separated repos,repo\*, or \*\* Tags: 5\* Enter multiple comma separated tags,tag\*, or \*\*

Criteria:  Only signed Images No vulnerability severity of Critical and above Labels: approved

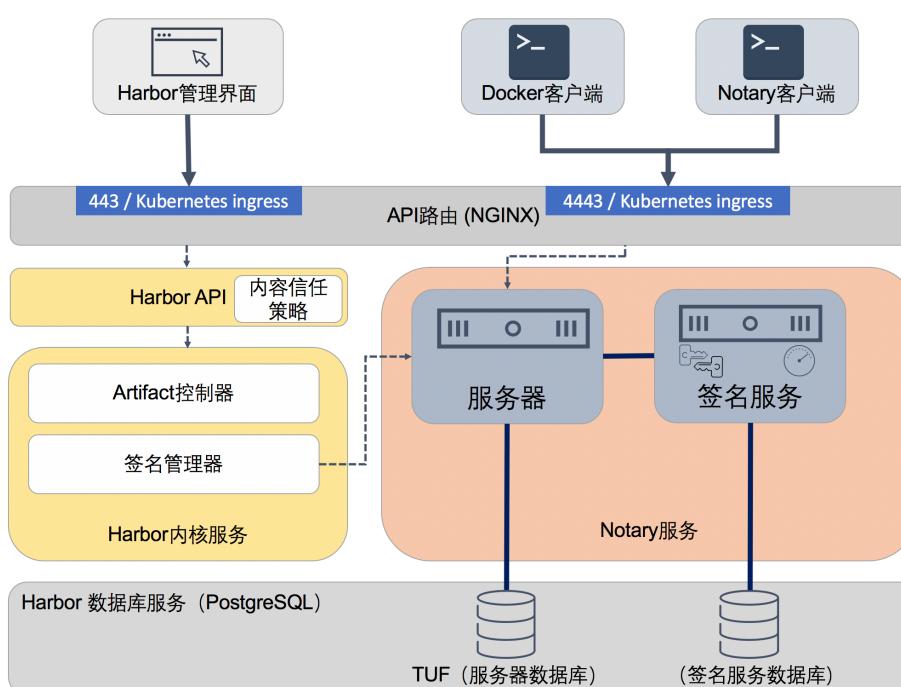
Trigger: Manual

CANCEL SAVE



# 制品安全分发-签名 [1]

Harbor基于开源的Notary项目实现镜像的签名（兼容DTR）



项目 < 镜像仓库 < nginx  
sha256:d9002da0

Tags

+ 添加 TAG

- 删除 TAG

名称 已签名

latest ✓

```
$ export DOCKER_CONTENT_TRUST=1
$ export DOCKER_CONTENT_TRUST_SERVER=https://<harbor主机地址>:4443
```

# 制品安全分发-签名 [2]

Harbor基于Helm社区支持的GPG实现对Helm V2 chart的签名支持

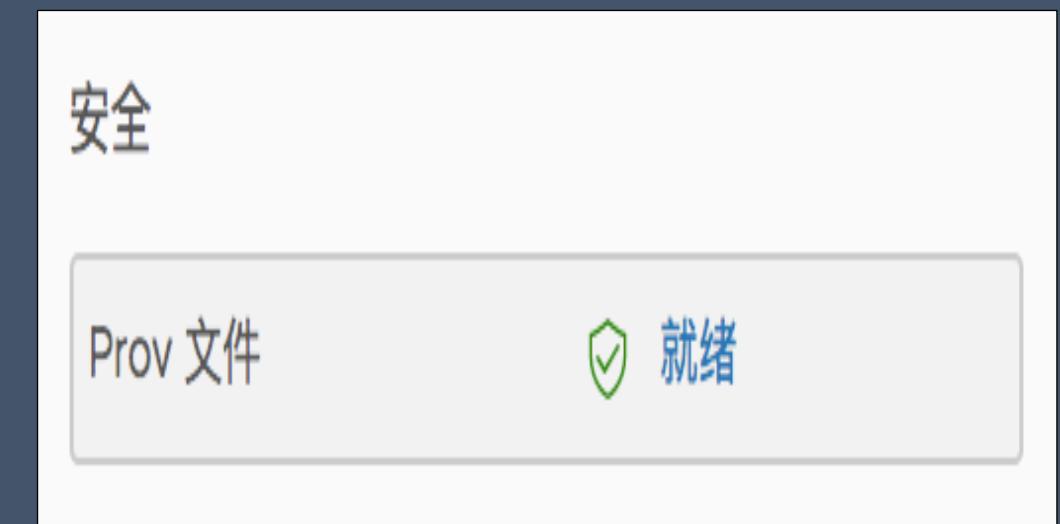
```
helm package --sign --key 'my signing key' --keyring path/to/keyring.secret mychart
```

上传Chart文件

Chart 文件  选择文件

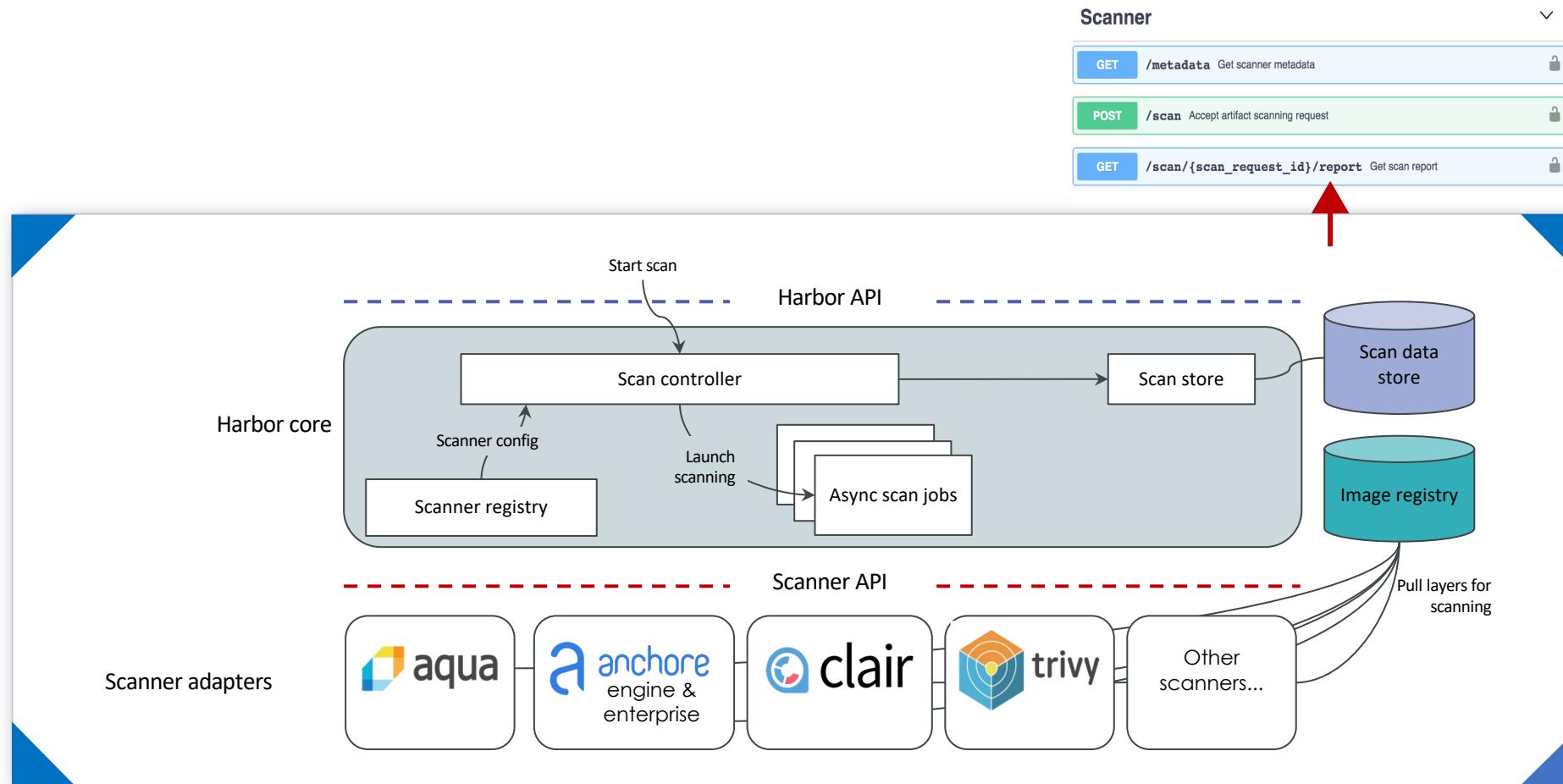
Prov 文件  选择文件

取消 上传



# 制品安全分发-扫描 [1]

通过插件化的扫描器接入使得用户可以选择自己偏向的扫描器来进行漏洞扫描



来自中国厂商：  
• 小佑科技  
• 探针科技

# 制品安全分发-扫描 [2]

扫描报告有助于实时了解所管理镜像的相关漏洞信息和安全威胁程度

审查服务

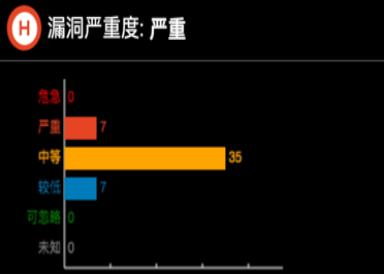
扫描器 漏洞

镜像扫描器 ①

+ 新建扫描器 设为默认 其他操作 ▾

名称	地址	健康	启用	认证模型
Trivy	http://trivy-adapter:8080	健康	true	None
Clair	http://clair-adapter:8080	健康	true	None

1 - 2 共计 2 条记录

镜像	版本	大小	状态	漏洞
sha256:644fcb1a	3.9.2	17.56MB	<span>H</span> • 49 总计 • 49 可修复	 扫描用时: 578 min 8 sec 扫描完成时间: 5/16/20, 7:53 PM
sha256:8b6b8c0f	20190408	17.72MB	<span>无</span> 没有漏洞	
sha256:28ef97b8	3.9.3	17.58MB	<span>H</span> • 42 总计 • 42 可修复	
sha256:66790a2b	3.6... (2)	11.63MB	<span>无</span> 没有漏洞	
sha256:5040b958	20190508	17.73MB	<span>无</span> 没有漏洞	
sha256:ca1c944a	3.10.0	17.76MB	<span>H</span> • 42 总计 • 42 可修复	

其他

漏洞

扫描

缺陷码	严重度	组件	当前版本	修复版本
CVE-2019-14697 ①	严重	musl	1.1.20-r3	1.1.20-r5
CVE-2019-14697 ①	严重	musl	1.1.20-r3	1.1.20-r5
CVE-2019-14697 ①	严重	musl	1.1.20-r3	1.1.20-r5
CVE-2019-14697 ①	严重	musl	1.1.20-r3	1.1.20-r5
CVE-2019-14697 ①	严重	musl	1.1.20-r3	1.1.20-r5
CVE-2019-14697 ①	严重	musl	1.1.20-r3	1.1.20-r5
CVE-2019-14697 ①	严重	musl	1.1.20-r3	1.1.20-r5
CVE-2019-1543 ①	中等	openssl	1.1.1a-r1	1.1.1b-r1
CVE-2019-1549 ①	中等	openssl	1.1.1a-r1	1.1.1d-r0

# 制品安全分发-安全策略



Virtual

可在项目级别设置相关安全策略以阻止不符合安全规范的镜像的分发

基于漏洞严重程度或者签名状态

项目

library 系统管理员

概要 镜像仓库 Helm Charts 成员 标签 扫描器 策略 机器人账户 Webhooks 日志 配置管理

项目仓库  公开  
所有人都可访问公开的项目仓库。

部署安全  内容信任  
仅允许部署通过认证的镜像。  
 阻止潜在漏洞镜像  
阻止危害级别 危急 以上的镜像运行。

# 制品安全分发-不可变Tag



Virtual

通过设置不可变规则来避免特定 Tag 被覆盖或者误删除

添加不可变的 Tag 规则

为此项目指定不可变的 Tag 规则。注意：所有不可变的 Tag 规则都将首先被独立计算，合并后最终获得不可变规则的集合。

应用到仓库

匹配   
使用逗号分隔 repos,repo\* 和 \*\*。

Tags

匹配   
使用逗号分割 tags,tag\* 和 \*\*。

Tags

<input type="checkbox"/>	名称	已签名	拉取时间	推送时间
<input type="checkbox"/>	test	<span style="color: red;">×</span>		6/15/20, 5:13 PM
<input type="checkbox"/>	latest	<span style="border: 1px solid blue; border-radius: 50%; padding: 2px;">不可变的</span>	<span style="color: red;">×</span>	6/15/20, 4:59 PM

1 - 2 共计 2 条记录

# 资源清理与垃圾回收 [1]



# Virtual

通过Artifact保留策略实现资源清理：根据用户设置的保留策略计算得出需要保留的资源而删除不需要保留的资源

不释放存储空间/释放配额

项目

# library 系统管理员

概要 镜像仓库 Helm Charts 成员 标签 扫描器 **策略** 机器人账户 Webhooks 日志 配置管理

**TAG保留 不可变的TAG**

保留规则 (2/5)

操作  应用到仓库匹配\*\*, 保留最近推送的2个 artifacts基于条件tags匹配\*\*基于条件 无 Tag  
操作  应用到仓库匹配hello-world, 保留全部 artifacts基于条件tags匹配\*\*基于条件 无 Tag

**添加规则**

定时执行 无 编辑

**运行保留策略**

**立即运行 模拟运行 中止 C**

ID	状态	模拟运行	执行类型	开始时间	持续时间

暂无记录!

0 条记录

以 artifact 数量或天数为条件

Tags

✓ 保留最近推送的#个 artifacts  
保留最近拉取的#个 artifacts  
保留最近#天被推送过的 artifacts  
保留最近#天被拉取过的 artifacts  
保留全部 artifacts

以 artifact 数量或天数为条件

✓ 保留最近推送的#个 artifacts

保留最近拉取的#个 artifacts  
保留最近#天被推送过的 artifacts  
保留最近#天被拉取过的 artifacts  
保留全部 artifacts

注意：不可变Tag一定会被保留

# 资源清理与垃圾回收 [2]



Virtual

通过垃圾回收可以清理存储空间中的无用数据，V2.1之前为阻塞式GC，V2.1之后实现非阻塞式

释放并回收空间/可能释放配额

垃圾清理

垃圾清理 历史记录

删除无 Tag 的 Artifacts

无

编辑 立即清理垃圾

1

垃圾清理

垃圾清理 历史记录

历史任务

任务ID	触发类型	状态	创建时间	更新时间	日志
2	手动	已完成	Jun 15, 2020, 5:06:11 PM	Jun 15, 2020, 5:06:11 PM	<a href="#">日志</a>
1	手动	已完成	Jun 15, 2020, 5:06:05 PM	Jun 15, 2020, 5:06:10 PM	<a href="#">日志</a>

最新的 2 个任务

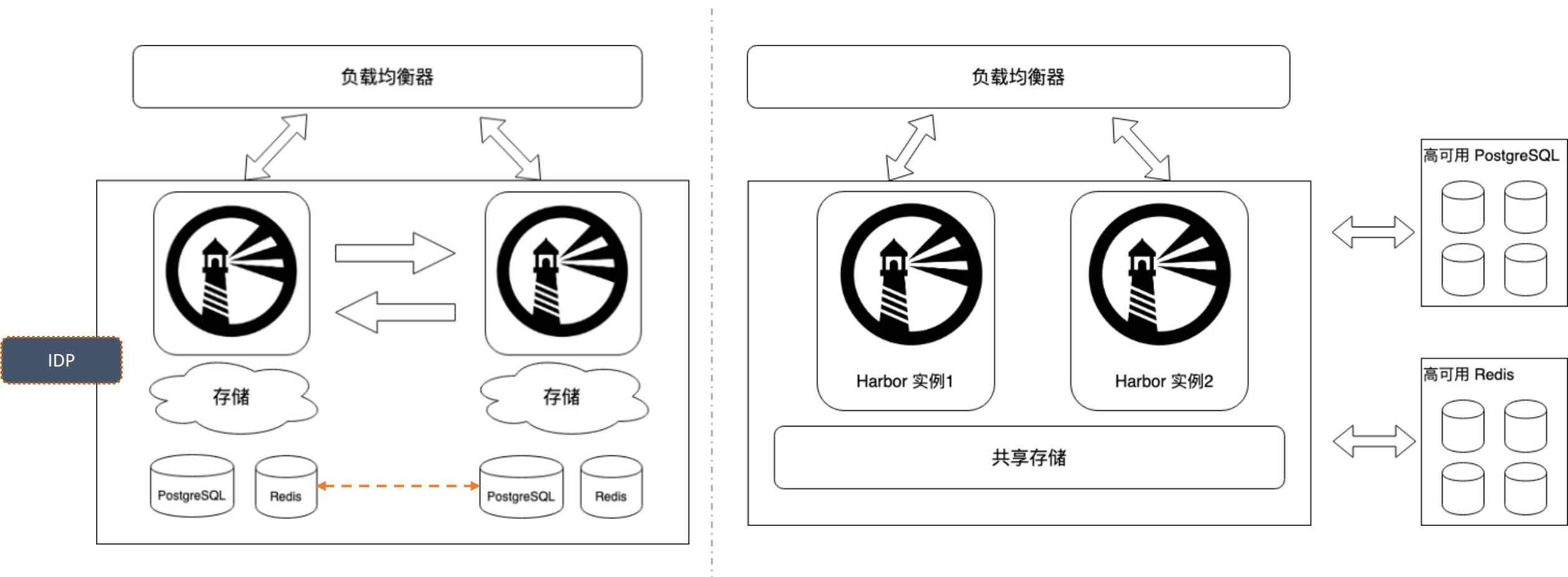
2

2020-06-15T09:06:11Z [INFO] [/jobservice/job/impl/gc/garbage\_collection.go:131]: start to run gc in job.  
2020-06-15T09:06:11Z [INFO] [/jobservice/job/impl/gc/garbage\_collection.go:271]: required candidate: \$+v[]  
2020-06-15T09:06:11Z [INFO] [/jobservice/job/impl/gc/garbage\_collection.go:279]: end to delete required artifact.  
2020-06-15T09:06:11Z [INFO] [/jobservice/job/impl/gc/garbage\_collection.go:236]: flush artifact trash  
2020-06-15T09:06:11Z [INFO] [/jobservice/job/impl/gc/garbage\_collection.go:144]: GC results: status: true, message: demo/redis  
demo/redis: marking manifest sha256:86edeb58fb97b7e941515a5f1bb571547432fedba f3513117ba521d99ef9d  
demo/redis: marking blob sha256:4cd8ec704e477aab9d24926e60b9ab8a25cbeff48f0ff23ac5eae879a98a7ebdef  
demo/redis: marking blob sha256:c499e6d256d6d4a546fc141e04b5b4951983ba7581e39deaf5c595289ee70f  
demo/redis: marking blob sha256:b1bc8a5a7e4b2529c9f8393946e0ba020b961ffbb2682807987f407e6da2f266  
demo/redis: marking blob sha256:7564fb795604548fde45efccac7008b17ac86fa01f7fd393a6b9ecfb31d0d889  
demo/redis: marking blob sha256:ec6e86f783e4327c9957aefc3ba1079ac5adf4c60bd0af15204508cadb885d9  
demo/redis: marking blob sha256:1371d6223f46d8d18bfef5f54db873d03a116e0d02e2b7887e32d50db56af58a  
demo/redis: marking blob sha256:021fd554320fb0855c449e33fe30fe134f781409c5da4f78683f9ebd80e0188c1  
8 blobs marked, 0 blobs and 0 manifests eligible for deletion  
, start: 2020-06-15 09:06:11.5387112 +0000 UTC, end: 2020-06-15 09:06:11.601464316 +0000 UTC.  
2020-06-15T09:06:11Z [INFO] [/jobservice/job/impl/gc/garbage\_collection.go:145]: success to run gc in job.

3

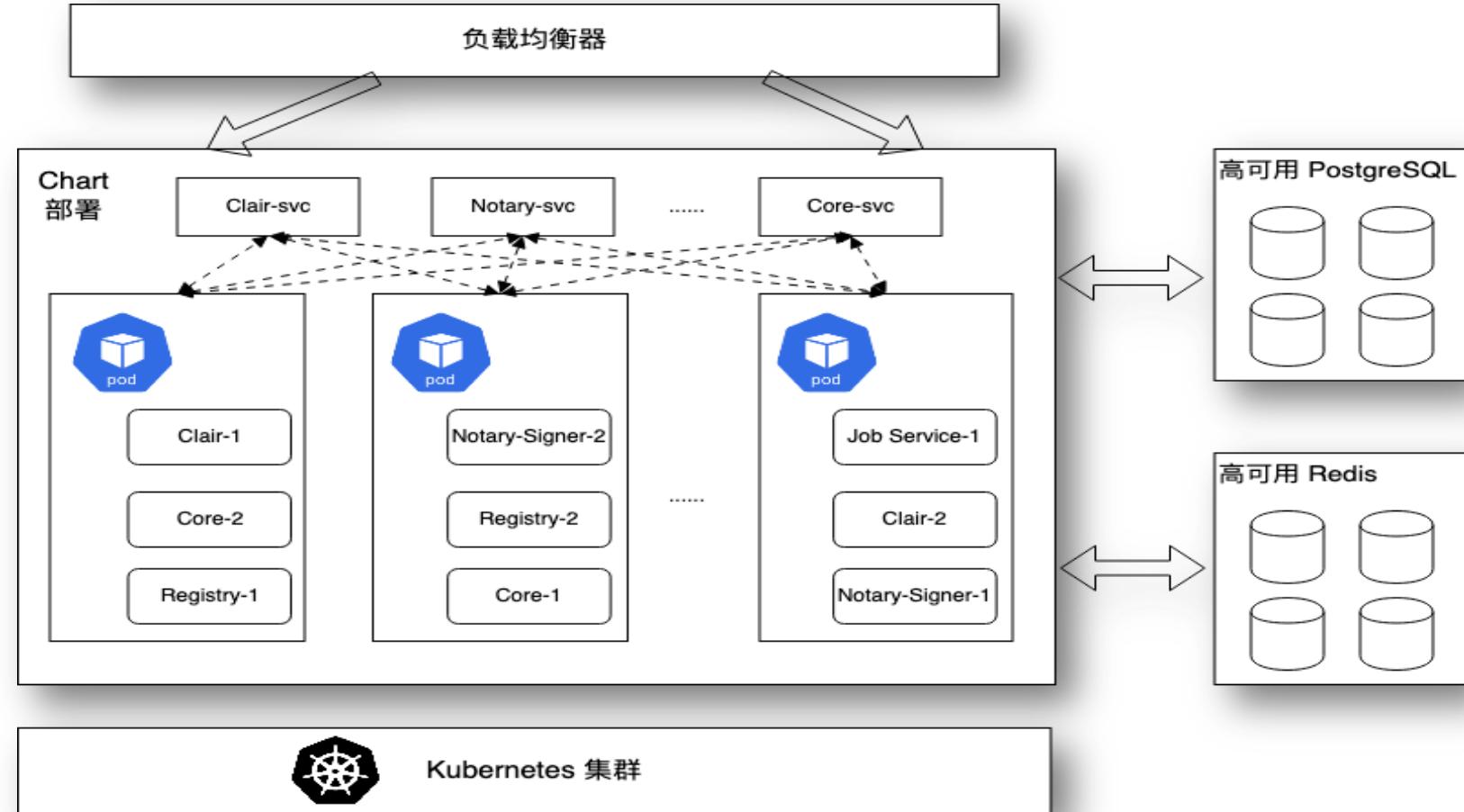
# 构建高可用(HA)仓库服务 [1]

使用离线安装包搭建HA仓库服务：基于**内容复制能力**或者基于**外部共享服务**



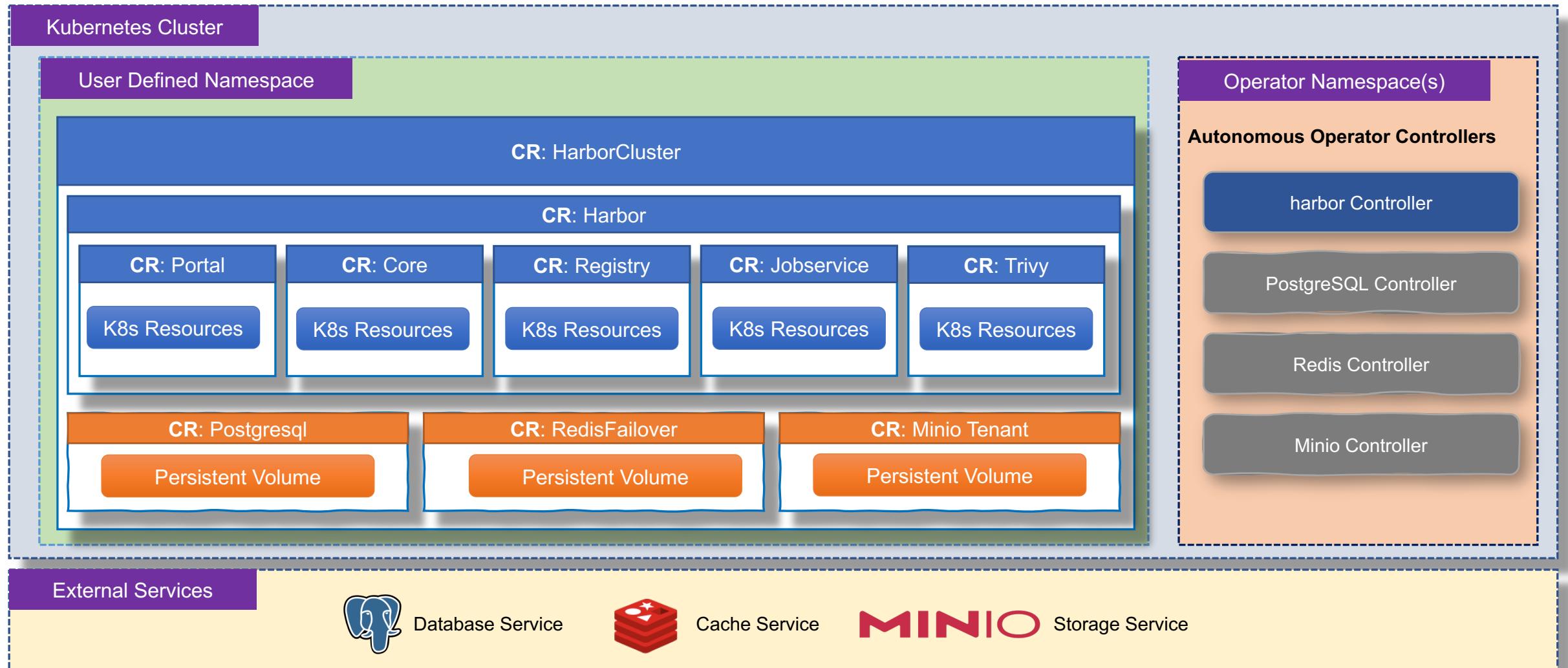
# 构建高可用(HA)仓库服务 [2]

使用Helm Chart和外部高可用服务（数据库，缓存和存储）部署HA的仓库服务



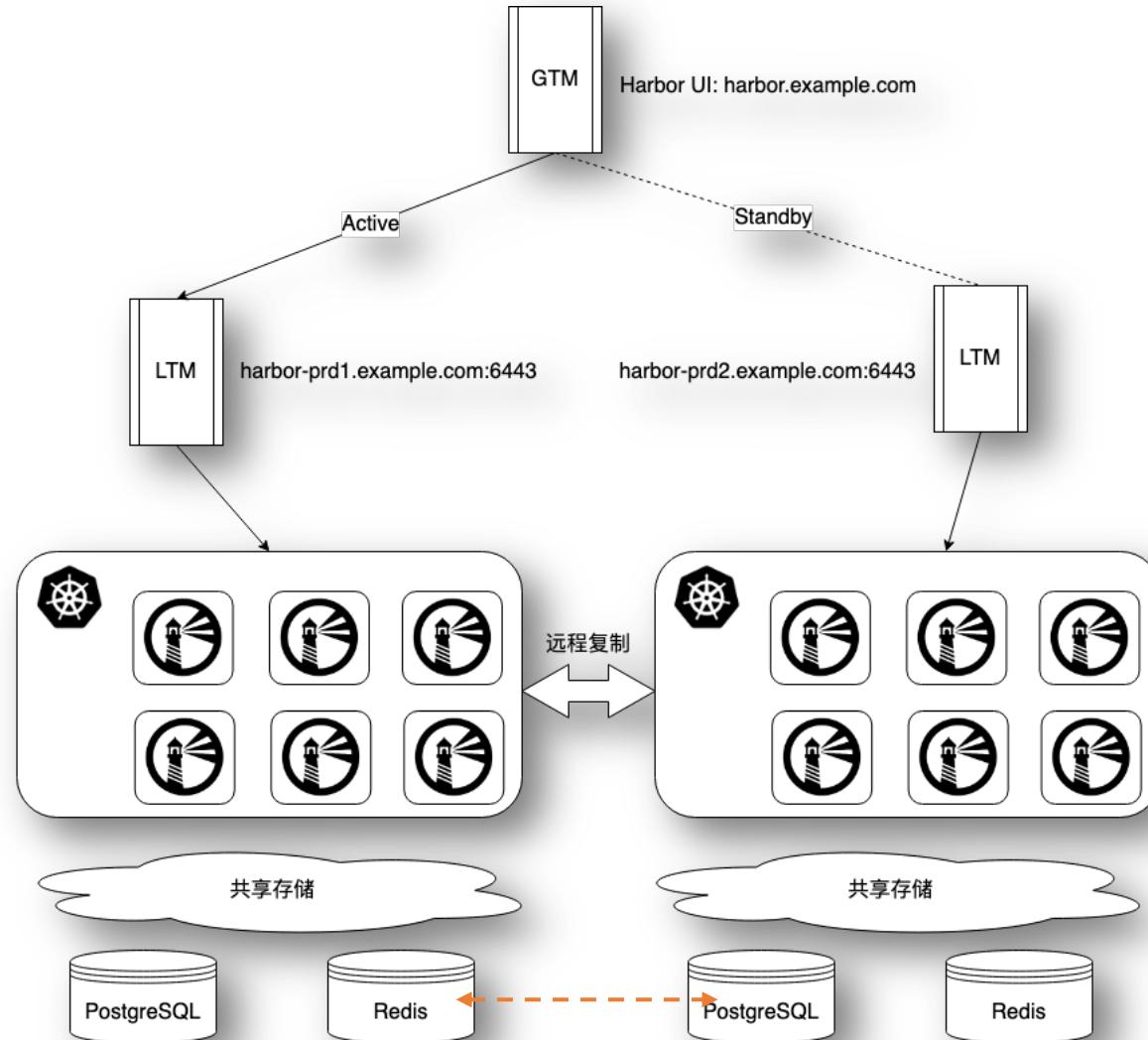
# 构建高可用(HA)仓库服务 [3]

Harbor Operator提供基于K8s集群的all-in-one HA解决方案(也支持使用外部共享服务)



# 构建高可用(HA)仓库服务 [4]

多数据中心HA部署



# 与Harbor集成



## Restful API

- 完善的API
- 遵循OpenAPI规范
- 通过Swagger生成调用代码



## AUTH集成

- AD/LDAP
- OIDC



## P2P引擎

- 标准化Adapter接口定义



## 制品类型

- 遵循OCI规范
- annotation化的数据扩展
- UI自动渲染扩展数据



## 漏洞扫描器

- 扫描器适配API规范
- 插件式扫描器框架



## 系统监控

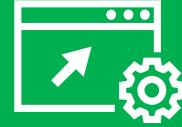
- 早发现系统问题或者性能瓶颈
- 包含系统状态和业务数据



v2.2

# 路线图

1



管理



K8s  
Operator



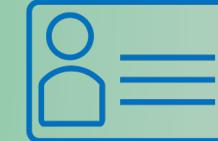
Observability



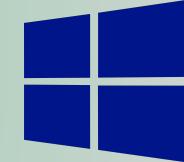
Performance



Backup & Restore



IAM&RBAC

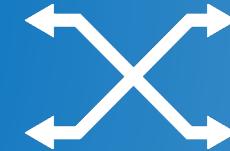


Windows Containers

2

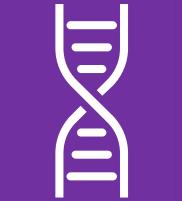


分发



Networking(IPV6)

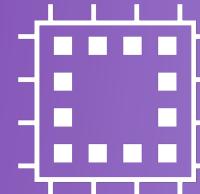
3



扩展性

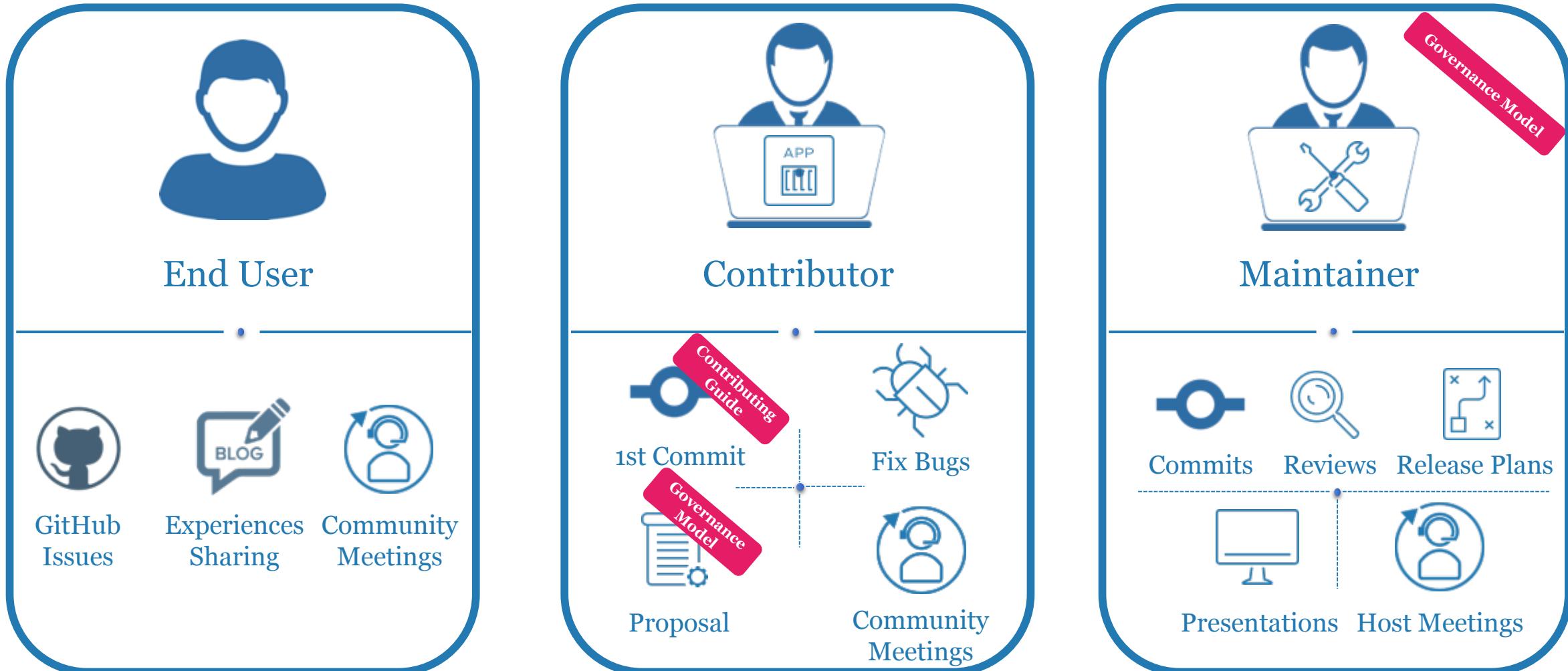


Interrogation  
Service++



Multi-CPU-arch

# 参与贡献Harbor社区 [1]



# 参与贡献Harbor社区 [2]

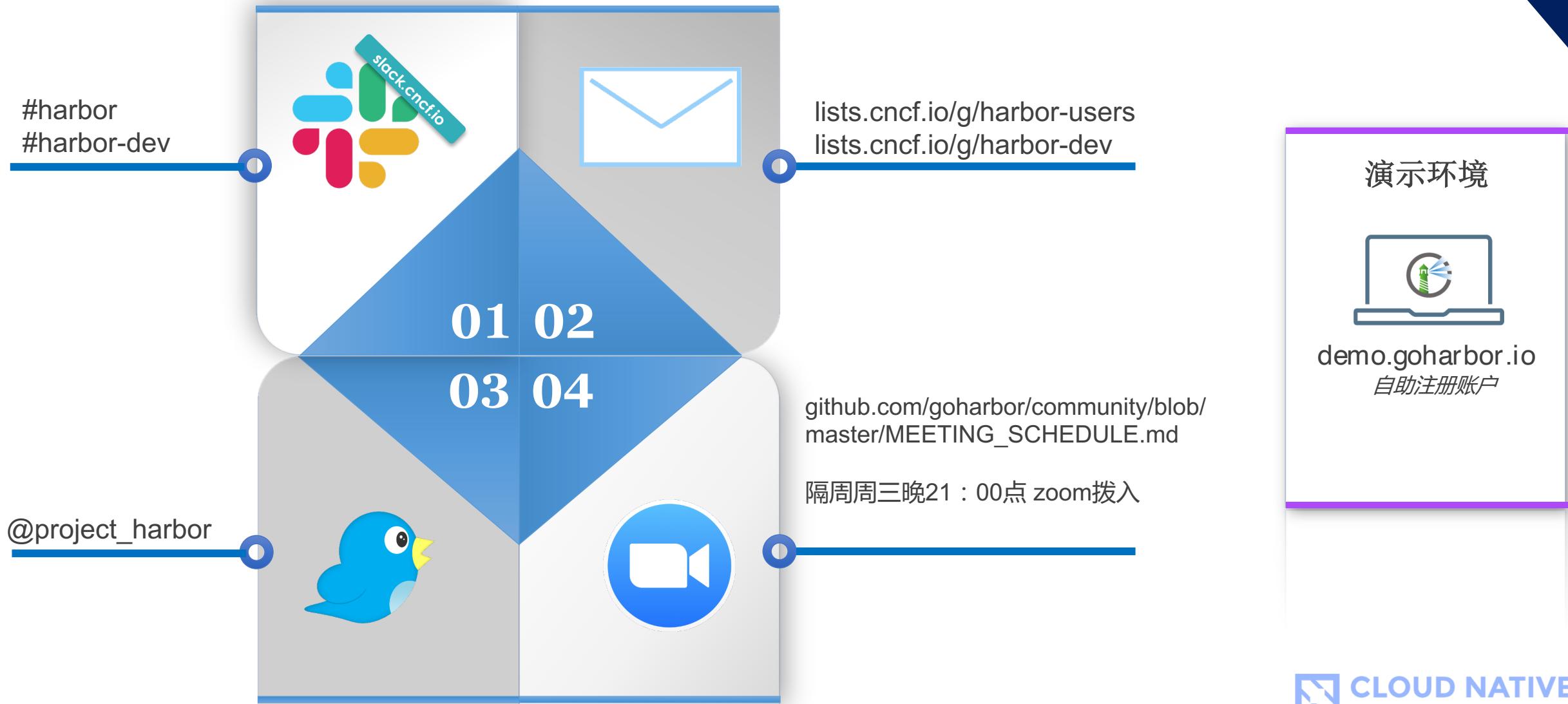


KubeCon



CloudNativeCon  
North America 2020

goharbor.io  
Virtual



# Harbor公众号



*Virtual*



欢迎关注!

