

Security Level:

华为docker测试实践

华为中央软件研究院欧拉docker实践小组、欧拉八部

2016-03

www.huawei.com

Author/ Email: 孙远/sunyuan3@huawei.com

Version: V1.0(20YYMMDD)

HUAWEI TECHNOLOGIES CO., LTD.



我的经历

2008年-2014年 风河系统公司

- 负责linux build system、analysis tools、workbench测试工作。

2015年-今 华为北京研究所

- 负责容器OS、docker测试工作
- 补充了ltp社区user namespace特性测试用例和docker社区中多个测试用例
- 参与了<<docker进阶与实战>>测试章节的编写工作

现场互动问题

- 1.现场从事代码coding的朋友
- 2.现场从事软件测试的朋友
- 3.曾经使用过docker的朋友
- 4.参与过开源社区开发的朋友

本次讲解的内容

- 1.docker测试技术的应用
- 2.docker社区的测试
- 3.docker测试实践
- 4.参与开源社区测试用例开发的实践

传统测试领域的困惑



传统软件开发的流程：

需求分析->迭代开发->迭代测试->...->产品发布->运维

痛点：

1. 开发、测试、运维环境不统一；无法准确获取客户的软件环境。
2. 开发在提交代码前未做充分的测试。
3. 开发无法复现测试报出的bug，开发与测试之间相互推诿。
4. 配置测试环境的时间较长，测试自动化成本高。

针对痛点思考的问题

- 1.如何配置一致的测试环境?
- 2.如何快速部署软件?
- 3.如何并行执行测试?
- 4.什么是复现bug的最佳方式?

Docker 技术简介

Docker三个核心技术：

Namespace, Cgroup,联合文件系统

Namespace 隔离作用

- Pid:隔离进程
- Mount:隔离文件系统挂载点
- User:隔离用户ID和组ID
- Network:隔离网络资源
- Ipc:隔离System V IPC和POSIX消息队列

Docker 技术简介

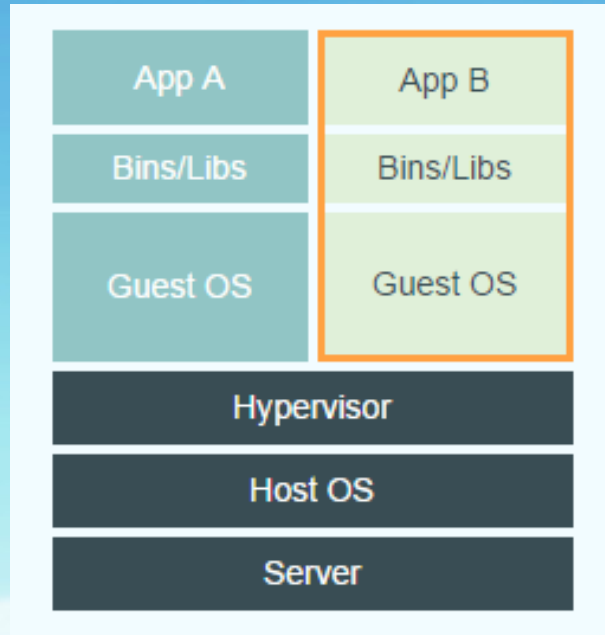
Cgroups

1. Cpuset子系统：分配指定的CPU和内存节点。
2. Cpu子系统：限制进程的CPU占用率。
3. Cpuacct子系统：统计各个Cgroup的CPU使用情况。
4. Memory子系统：限制所能使用的内存上线。
5. Blkiot子系统：限制block I/O带宽。
6. Devices子系统：控制进程对哪些设备有访问权限。

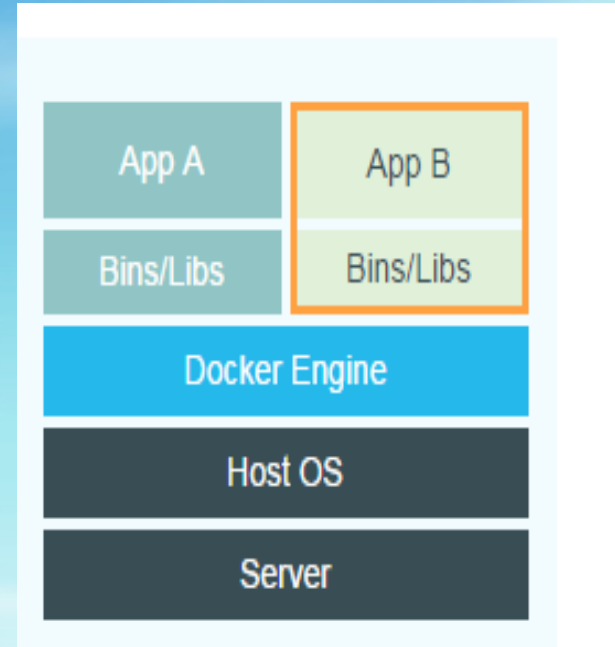
联合文件系统

- 一种分层、轻量级并且高性能的文件系统，它支持对文件系统的修改作为一次提交来一层层的叠加，同时可以将不同目录挂载到同一个虚拟文件系统下。

VM vs Docker



VM



Docker

秒级启动/快速部署/镜像分享

docker技术在测试中的应用



docker对测试的革命性影响

1. 让单元测试运行得更顺畅。(快速构建环境,测试驱动开发)
2. 让虚拟机不再困扰集成测试和功能测试。(避免多任务共享虚拟机)
3. 让测试团队和客户丢掉冗长的配置文档。
4. 可以轻松地复现客户报告的bug。
5. 通过Dockerfile可以梳理好测试镜像制作的流程。
6. 方便软件厂商将成熟的测试套或测试工具通过镜像共享。

docker技术在测试领域的适用范围

理想：Build, Ship, and Run Any App, Anywhere

现实局限：

1. 由于容器与主机共用内核，如果容器需要使用不同的内核版本就不得不更换主机内核。
2. 不能修改内核参数或者自主定制内核。
3. 对内核版本有依赖性，Docker通常需要3.10或以上版本的内核。
4. 在容器中加载或卸载内核模块会影响到主机和其他容器。
5. 跨主机容器间通信能力不足。
6. 无法像qemu一样模拟嵌入式系统运行环境。

业务名称	是否适合使用Docker
编译系统测试	是
数据库测试	是
与内核相关的网络测试	否
内核测试套LTP	否
Web应用测试	是
应用软件安装测试	是
ARM嵌入式软件模拟测试	否

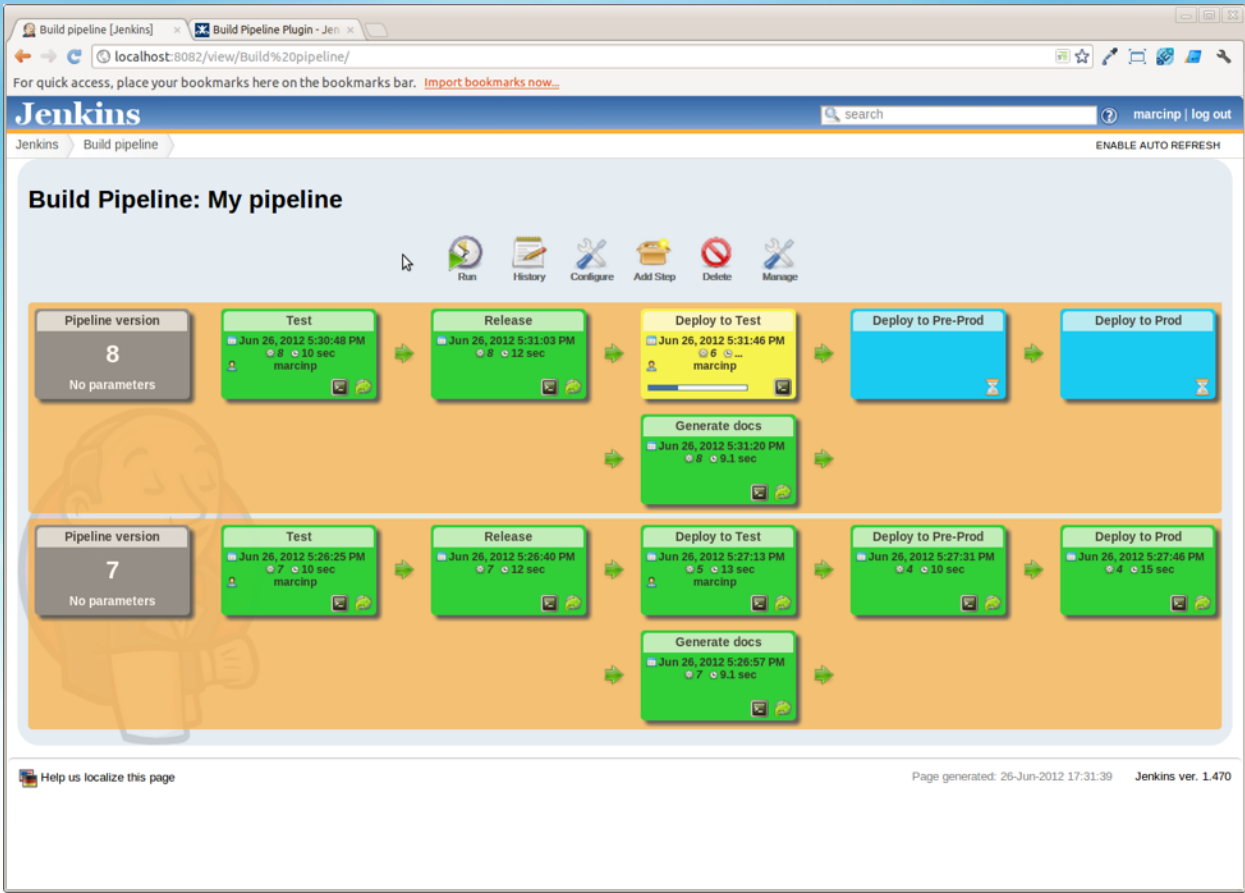
Docker与测试自动化



Build Pipeline插件

Docker build step plugin	可以添加Docker命令到构建步骤中。
CloudBees Docker Build and Publish plugin	提供通过Dockerfile构建工程的能力并将制作好的镜像发布到Docker仓库中。
Docker Plugin	可以使用Docker主机动态分配的容器作为Jenkins的从节点。
Kubernetes Plugin	通过由Kubernetes管理的多个Docker主机系统来动态分配的容器作为Jenkins的从节点。
Docker Commons Plugin	为其他与Docker相关的插件提供API。

docker pull jenkins



docker开源社区的测试



docker社区自身的测试框架

1. Go语言的测试框架。（部署简单、并发性好）
2. 通过testing包和go test命令来提供测试功能
3. 测试代码包含在文件名以“_test.go”关键字结尾的文件中
4. 可以使用go test -cover命令来获取代码测试覆盖率。

docker社区测试用例集介绍

- # git clone https://github.com/docker/docker.git
- # cd docker && make test

Target名称	作用
test	运行所有测试用例
test-unit	运行单元测试用例
test-integration-cli	运行集成测试用例

docker社区测试用例集介绍

```
root@p1:/tmp/docker/integration-cli# ls
check_test.go
docker_api_attach_test.go
docker_api_build_test.go
docker_api_containers_test.go
docker_api_create_test.go
docker_api_events_test.go
docker_api_exec_resize_test.go
docker_api_exec_test.go
docker_cli_login_test.go
docker_cli_logs_test.go
docker_cli_nat_test.go
docker_cli_netmode_test.go
docker_cli_network_unix_test.go
docker_cli_oom_killed_test.go
docker_cli_pause_test.go
docker_cli_port_test.go
```

```
PASS: docker_cli_pull_test.go:106: DockerHubPullSuite.TestPullAllTagsFromCentralRegistry
PASS: docker_cli_pull_test.go:137: DockerHubPullSuite.TestPullClientDisconnect
PASS: docker_cli_pull_test.go:20: DockerHubPullSuite.TestPullFromCentralRegistry
PASS: docker_cli_pull_test.go:69: DockerHubPullSuite.TestPullFromCentralRegistryImplicitRefParts
PASS: docker_cli_pull_test.go:46: DockerHubPullSuite.TestPullNonExistingImage
PASS: docker_cli_pull_test.go:96: DockerHubPullSuite.TestPullScratchNotAllowed
OK: 991 passed, 38 skipped
PASS
```

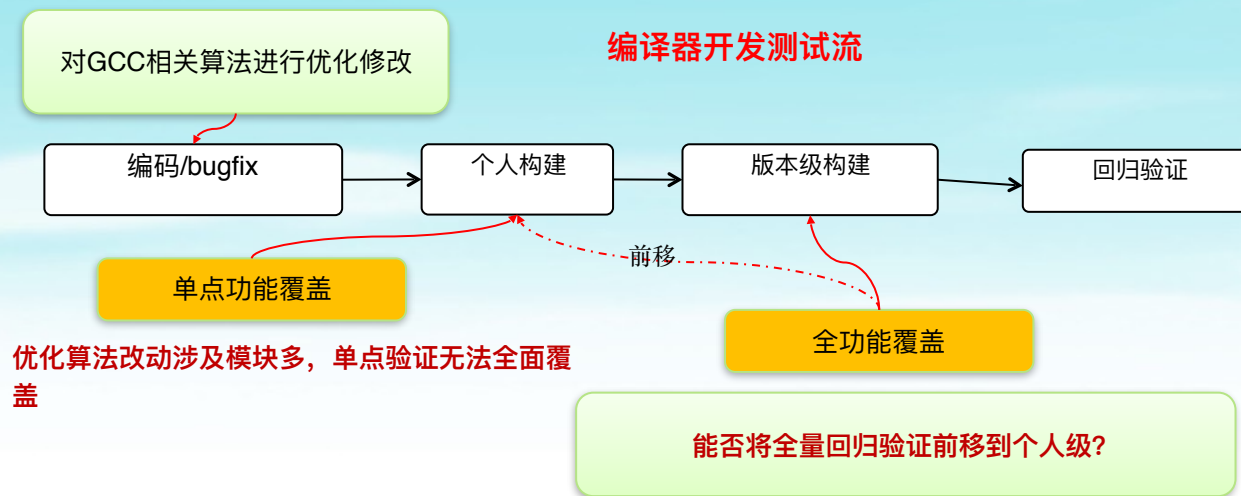
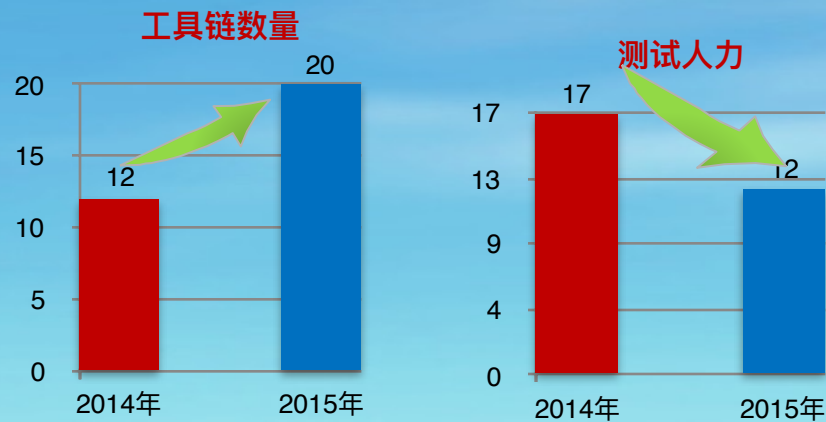
docker社区测试用例集需要改进的方面

1. 缺少性能测试用例。
2. 缺少可靠性、稳定性测试用例。
3. 缺少边界和异常测试用例。
4. 无法产生界面友好的测试报告。
5. 缺少安全测试用例。

docker测试实践

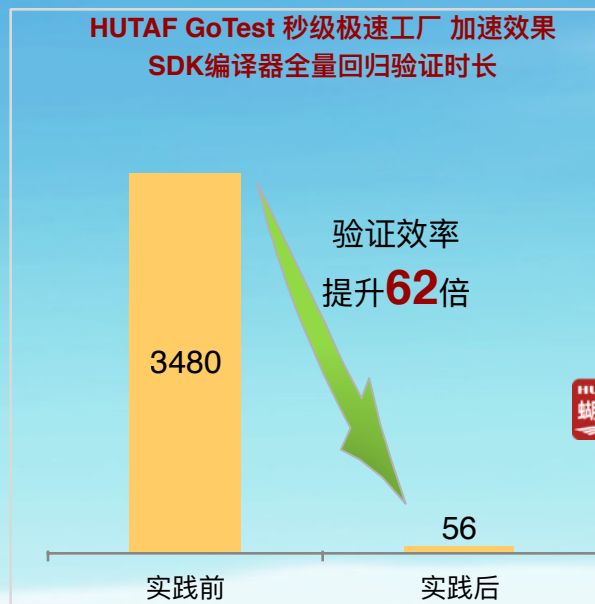
- SDK编译测试
- 外围包测试
- 网络测试
- 测试私有云

SDK编译测试



基于Docker容器化环境的秒级极速工厂，全量回归验证加速效果

基于Docker容器技术再提速，实现SDK编译器14万+自动化用例执行时间从**58分钟**缩短至**56秒**，验证效率提升**62倍**，超越业界软件类验证标杆Facebook（2~3分钟）。



- 1、Google测试验证能力：
1.1万多测试用例，全量回归验证**30分钟**完成
- 2、Facebook测试验证能力：
mobile版本有6000多测试用例，全量回归验证**2~3分钟**
- 3、HUAWEI测试验证能力：
SDK14万多测试用例，自动化率99%，全量回归验证**56秒**完成

关键
实践
与
能力

1

容器化环境能力

- 部署效率：测试套跨容器间共享，测试环境秒级Ready（**5分钟→9秒**）
- 环境使用率：提升环境使用率（**5~10倍**）

2

智能分块调度能力

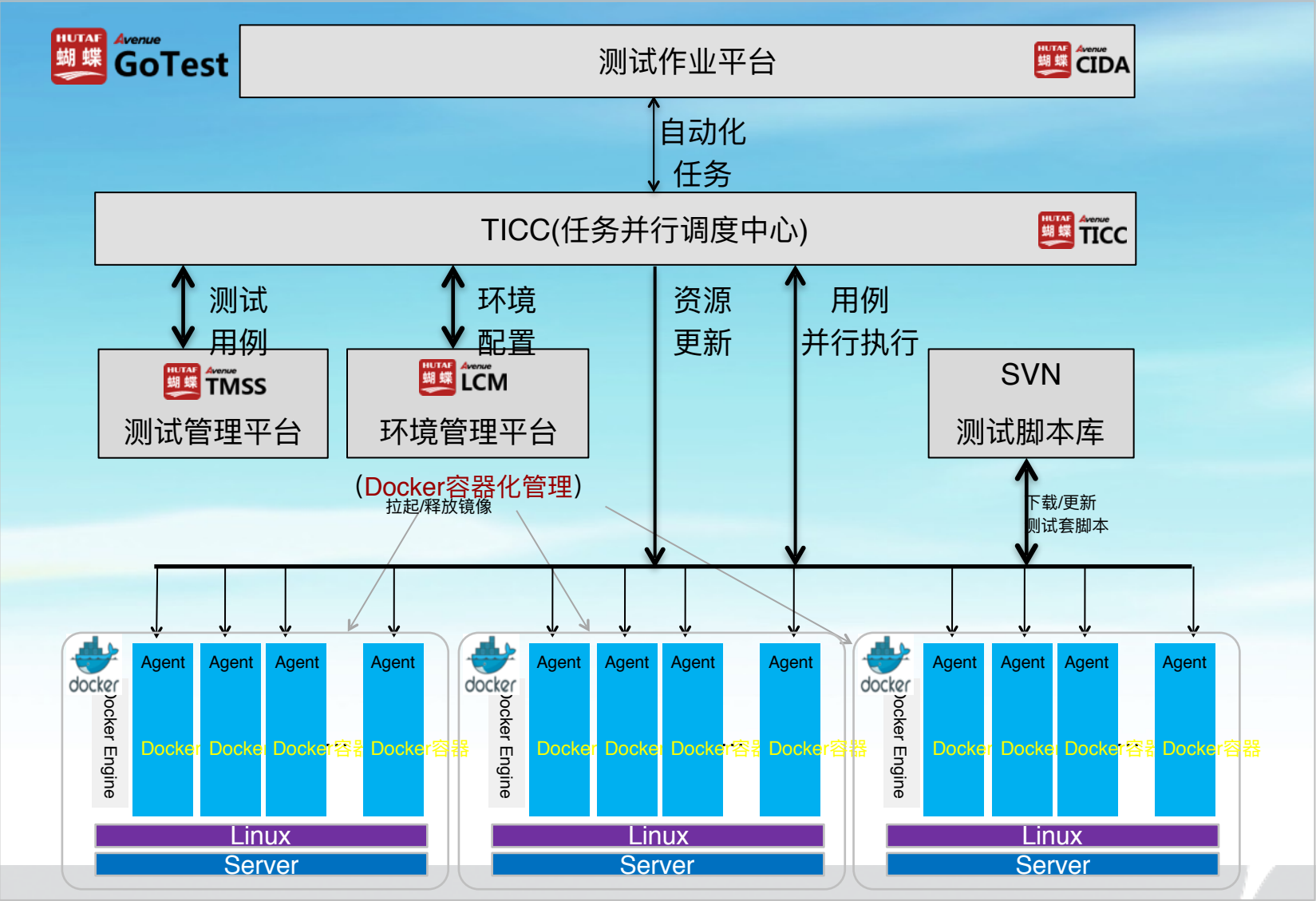
- 用例分块算法：十万级用例分块耗时（**30秒→5秒**）
- 均衡调度算法：用例块“**并发执行、同步结束**”

3

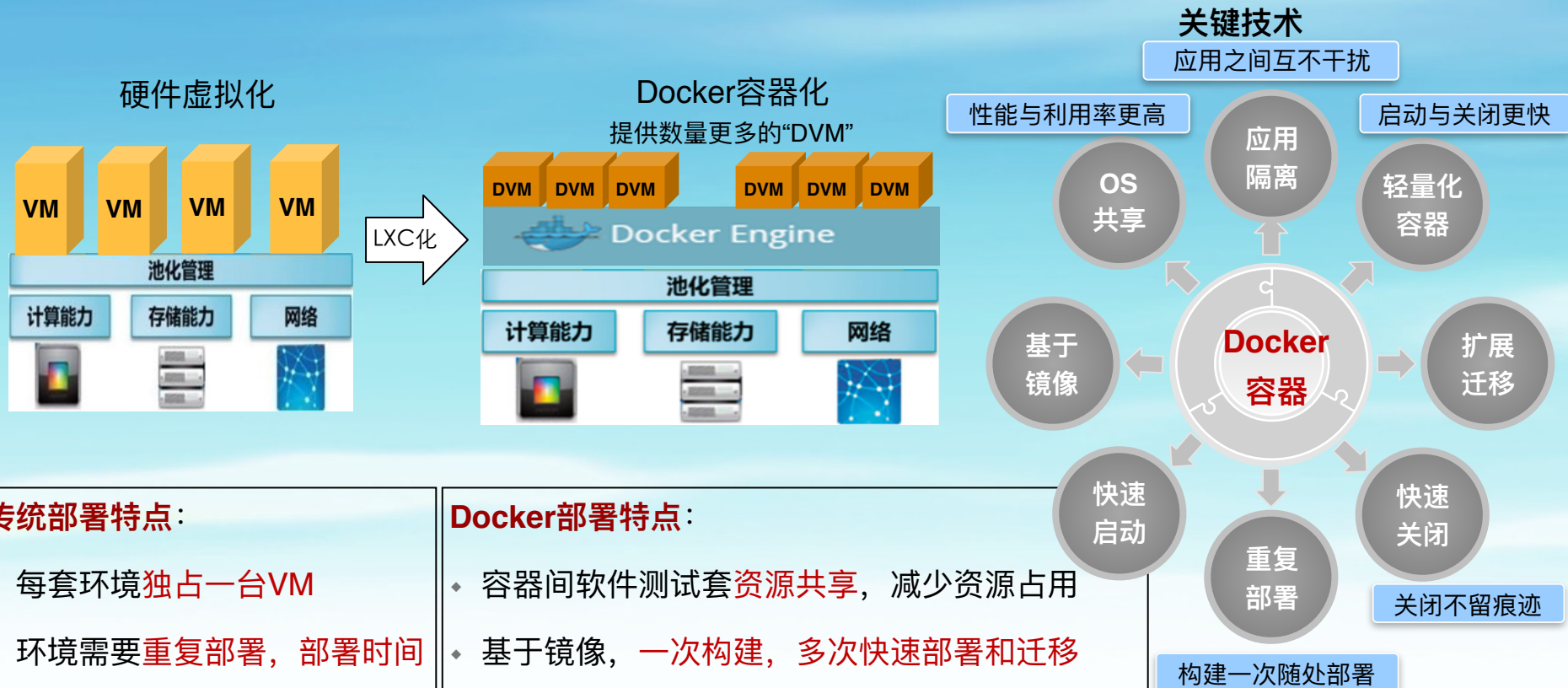
用例&脚本设计能力

- 细化大粒度用例：**10分钟→40秒内**
- 脚本用例执行并发度：单物理环境从**15→60+**

秒级自动化工厂 (基于Docker技术)



关键技术：秒级容器化环境能力，大幅降低资源消耗，提升可用环境数
5~10倍



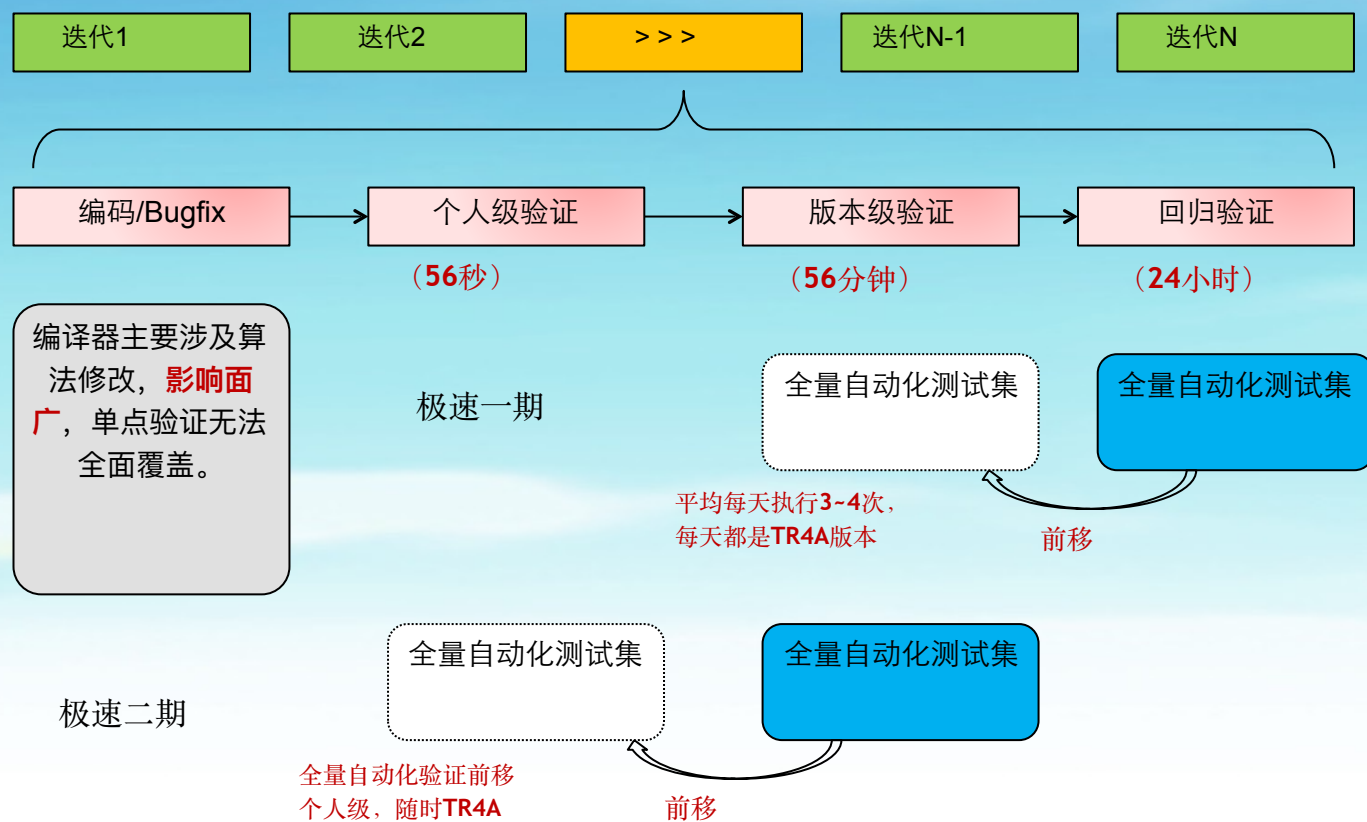
传统部署特点：

- ◆ 每套环境独占一台VM
- ◆ 环境需要重复部署，部署时间长
- ◆ 环境释放清理不干净，有残留

Docker部署特点：

- ◆ 容器间软件测试套资源共享，减少资源占用
- ◆ 基于镜像，一次构建，多次快速部署和迁移
- ◆ 容器可快速启动与关闭，每次都是干净环境
- ◆ 容器快速部署能力，实现环境资源跨产品跨团队共享

利用欧拉docker容器技术+蝴蝶GoTest构建自动化工厂，并行度达1800+规模，提升8倍。编译器全量回归验证56秒完成，实现全量回归验证前移到个人级验证，提升版本质量。



外围包测试&网络测试

coreutils, glibc, libpcrc, mktemp, passwd, sudo, tun-tap, acl, cpio,
grep, libssh2, mtd, pciutils, sysfsutils

传统部署特点:

- ◆ 每套环境独占一台主机
- ◆ 测试串行执行, 不易并发
- ◆ 环境释放时清理工作依赖于程序员的技能
- ◆ 无法解决多个外围包的环境污染问题
- ◆ 外围包编译环境不易统一
- ◆ 测试网络包时需要至少两台主机

Docker部署特点:

- ◆ 多套环境可以在同一主机上部署
- ◆ 测试并行执行, 提高了cpu利用率
- ◆ 环境释放时清理工作由docker接管
- ◆ 容器可快速启动与关闭, 每次都是干净环境
- ◆ 通过镜像保存编译环境, 确保环境统一
- ◆ 测试网络包时只需要在一台主机中启动两个容器

测试私有云（建设中）

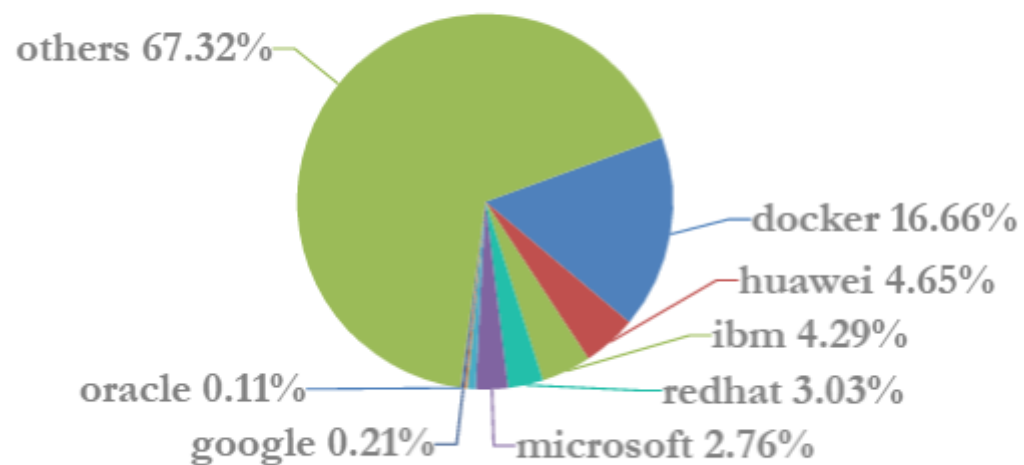
通过自定义WebUI管理界面+后台管理系统+mesos+k8s调度+测试专用仓库

目标：

- 容器OS微小化
- 磁盘限额
- 容器独立ip
- 资源管理（限定cpu、内存、io等消耗）
- 可弹性伸缩
- 测试专用docker仓库（多租户）
- 容器监控
- 测试容器热迁移
- 容器卷存储

Docker社区参与

Docker Community Engagement



Docker			
num	company	patch_num	proportion
1	docker	1441	16.66%
2	huawei	402	4.65%
3	ibm	371	4.29%
4	redhat	262	3.03%
5	microsoft	239	2.76%
6	fujitsu	63	0.72%
7	rancher	19	0.21%
8	google	19	0.21%
9	oracle	10	0.11%
10	suse	4	0.04%
11	others	5815	67.32%

参与社区开发的好处

- 对公司：提升影响力，服务于公司的产品战略。
- 对个人：
 - (1)某应届毕业生因对开源社区有突出贡献，毕业前即收到了美国google总部的offer
 - (2)某应届毕业生因在校期间参与某sdk社区开发，在风河入职后职位直升两级，跳过associate engineer和engineer，入职即为senior engineer。
 - (3)能够与世界顶级的程序员交流。

参与社区的思路

(1)根据目前自己所从事的工作定位选择恰当的社区，并在社区中找到与自己目前工作的结合点。

(2)熟悉社区参与方式，阅读userguide文档。

Ltp: 邮件或github

Docker: github

Kernel: 邮件

(3)使用Issues创建一个话题(如提出问题或报告bug)，需要提供详细的信息及bug复现方式。

(4)寻找社区中的patch点。（读懂社区中的已有用例，找到社区用例中未覆盖的功能或临界点）

(5)编写程序创建新用例并制作patch。

(6)提交patch并接受社区反馈。

(7)Patch被merge或reject，或者需要继续修改。

Docker社区添加测试用例实例

(1) Fork一个自己的repo，并在repo上进行开发。

(2) 修改代码后进行充分测试。

(3) 修改文件格式(gofmt -s -w \${some_files})

(4) 提交patch:

```
git add ${some_files}
```

```
git commit -s -m "commit note" (git commit --amend -s --no-edit)
```

```
git push -f origin master:${branch_name}
```

例子: <https://github.com/docker/docker/pull/11980>

(5) 创建pull request

https://github.com/docker/docker/issues

This repository Search Pull requests Issues Gist

docker / docker Unwatch 2,513 Unstar 29,172 Fork 8,162

Code Issues 1,297 Pull requests 64 Wiki Pulse Graphs

Filters is:issue is:open Labels Milestones New issue

1,297 Open 8,047 Closed Author Labels Milestones Assignee Sort

- docker run/create 'cannot specify 64-byte hexadecimal strings' sha256: leakage
#20972 opened 7 hours ago by WhisperingChaos
- Calling fork() exists container
#20971 opened 9 hours ago by I33
- Proposal: container startup notification socket
#20968 opened 14 hours ago by Xenopathic
- Fails to restart containers occasionally, but often. status/more-info-needed
#20964 opened 15 hours ago by chakshugoyal
- Docker daemon dies when fluent driver can not send messages to saturated fluent server
#20960 opened 17 hours ago by nikolaidershak
- Error response from daemon: open [path to file]: operation not permitted.
#20950 opened 20 hours ago by onorua

https://github.com/docker/docker/pull/11980

This repository Search Pull requests Issues Gist


docker / docker Unwatch 2,513

Code Issues 1,298 Pull requests 64 Wiki Pulse Graphs


add TestSearchCmdOptions case #11980


Merged LK4D4 merged 1 commit into docker:master from sunyuan3:TestSearchCmdOptions on 14 Apr 2015


Conversation 23 Commits 1 Files changed 1

 sunyuan3 commented on 1 Apr 2015

Signed-off-by: Yuan Sun sunyuan3@huawei.com

 GordonTheTurtle added the `status/0-needs-triage` label on 1 Apr 2015

 duglin commented on an outdated diff on 1 Apr 2015 [Show outdated diff](#)

 duglin added `status/2-needs-code-review` and removed `status/0-needs-triage` labels on 1 Apr 2015

Ltp社区添加测试用例实例

(1)配置git信息。

(2) 编写用例，添加公司版权。添加用例确保编译正确且没有warning以及成功执行用例。

(3) git add newcase.c

git commit -s -m “new case commit info”

git format-patch -1

checkpatch.pl检查patch

(4)发送patch给社区

```
git send-email --dry-run --to=jstancek@redhat.com --cc=ltp-  
list@lists.sourceforge.net --bcc=pleasuresun@sina.com --  
bcc=sunyuan3@huawei.com --bcc=peifeiyue@huawei.com 0001-  
containers-new-testcase-usersns02.patch
```

(5)以user namespace为例，讲述我的ltp社区思路。

User namespace

特性描述：将主机的uid映射到容器中，如将主机的普通用户映射为容器内的root用户。

第1步：阅读特性文档 <https://lwn.net/Articles/532593/>

第2步：掌握特性使用方法

第3步：在社区中阅读相关度大的用例，如pid namespace。

第4步：编写自己的用例。

```
uid = geteuid();
gid = getegid();

tst_resm(TINFO, "USERNS test is running in a new user namespace.");

if (uid != overflowuid || gid != overflowgid) {
    printf("Got unexpected result of uid=%d gid=%d\n", uid, gid);
    exit_val = 1;
}
```

社区参与常见问题注意事项

(1)如果不是原则问题，不要在细节问题上与maintainer过分争论

(2)与maintainer有不同意见时的策略

NO way.

I disagree with you.

In general, I agree with you. But in this case, ...

Maybe ...

Thanks && Many thanks && Thank you very much.

(3)建立社区影响力：刷patch数量很容易，但提升社区影响力并非易事。

- 改变系统架构

- 添加新的特性

- 修改bug

- 修改拼写错误或打印语句错误

(4)与maintainer搞好关系并逐步建立信任

(5)不要在社区提出低级问题，如“配置git的方式”

(6)不要在一个patch中包含多个特性的测试。要把patch细分,分别提交。

社区参与常见问题注意事项

(6)了解maintainer的个性，maintainer审查代码的严格程度不同，做事方式也不同。

(7)当patch石沉大海时：尝试在irc上联系maintainer；抄送邮件给更多maintainer。

(8)不要在社区中泄露公司的项目。

(9)没有把握时，可以先在内部review。以免出现低级问题对公司外部形象造成影响。

《Docker进阶与实战》

机械工业出版社出版

最全面
的内容



进阶与实战
的最佳选择

华为Docker实践小组 出品

Docker社区贡献全球排名前三、国内排名第一



【团队介绍】

我们致力于打造未来ICT领域的操作系统、云平台，我们致力于研究未来ICT、NFV/SDN、云服务场景下的OS、容器、虚拟化前沿技术。

这里有Linux kernel、Docker社区多名maintainer，有APCI标准提案committer，有精通Linux kernel各子系统（调度、内存管理、文件系统、网络等）的资深专家，有畅销书《Docker进阶与实战》的作者团队和您一起探讨容器虚拟化的未来技术。

如果你相信技术可以改变生活、改变世界；如果你喜欢Linux、喜欢钻研技术；如果你热爱开源、想成为技术大咖；别犹豫，加入我们！

Docker/容器虚拟化架构师/高级工程师（北京/杭州）

【职责】

- 负责x86、arm 64架构下容器虚拟化需求分析、关键技术和特性的设计；
- 负责Docker/容器关键技术研究 and 特性开发；
- 负责Docker及相关开源社区互动、运作，如推送bug fix和新特性。

【要求】

- 计算机相关专业本科及以上学历，4年以上Linux系统或内核开发经验；
- 熟悉Linux容器相关技术（namespace、cgroup）者优先；
- 熟悉Docker源码、有Docker社区开发经验者优先；
- 熟悉容器编排/调度（Kubernetes，Mesos等）技术者优先；

【招贤纳士】 【华为-Docker团队】

诚邀容器、虚拟化领域专家

【电话】 18501294585/北京、13732261657/杭州

【邮箱】 hr.kernel@huawei.com







【招贤纳士】 【华为-Docker团队】 诚邀容器、虚拟化领域专家

【Docker团队介绍】

我们致力于打造未来ICT领域的Linux操作系统平台，我们致力于研究未来ICT、NFV/SDN、云服务场景下的内核、容器虚拟化前沿技术。

这里有Linux kernel、Docker社区多名maintainer，有APCI标准提案committer，有精通Linux kernel各子系统（调度、内存管理、文件系统、网络等）的资深专家，有畅销书《Docker进阶与实战》的作者团队和您一起探讨容器虚拟化的未来技术。

如果你相信技术可以改变生活、改变世界；如果你喜欢Linux、喜欢钻研技术；如果你热爱开源、想成为技术大咖；别犹豫，加入我们！

【工作地点】

北京、杭州

【微信/手机】

13683618569、13732261657

【邮箱】

hr.kernel@huawei.com

Linux内核架构师/高级工程师（北京/杭州）

【岗位职责】

- 负责x86、arm 64架构下Linux系统需求分析、关键技术和特性的设计；
- 负责Linux内核特性、子系统开发，整体性能调试与优化；
- 负责操作系统生态链构建，如技术合作、开源运作等。

【岗位要求】

- 计算机相关专业本科及以上学历，4年以上Linux系统或内核开发经验；
- 熟悉Linux内核开发，精通一个或以上内核关键模块（如内存管理、调度、文件系统、驱动）者优先；
- 熟悉Linux系统调试调测，有数据中心运维经验者优先；
- 熟悉内核开源开发模式，有内核及相关社区补丁提交经验者优先。

Docker/容器虚拟化架构师/高级工程师（北京/杭州）

【岗位职责】

- 负责x86、arm 64架构下容器虚拟化需求分析、关键技术和特性的设计；
- 负责Docker/容器关键技术研究 and 特性开发；
- 负责Docker及相关开源社区互动、运作，如推送bug fix和新特性。

【岗位要求】

- 计算机相关专业本科及以上学历，4年以上Linux系统或内核开发经验；
- 熟悉Linux容器相关技术（namespace、cgroup）者优先；
- 熟悉Docker源码、有Docker社区开发经验者优先；
- 熟悉容器编排/调度（Kubernetes，Mesos等）技术者优先；





Thank you

www.huawei.com

Copyright©2011 Huawei Technologies Co., Ltd. All Rights Reserved.

The information in this document may contain predictive statements including, without limitation, statements regarding the future financial and operating results, future product portfolio, new technology, etc. There are a number of factors that could cause actual results and developments to differ materially from those expressed or implied in the predictive statements. Therefore, such information is provided for reference purpose only and constitutes neither an offer nor an acceptance. Huawei may change the information at any time without notice.