Compass CI: a developer friendly OSS test service
Compass CI: from openEuler, for the OSS world

• testing OS
  • rootfs can be openEuler or Debian/CentOS/Archlinux/...
  • rpmbuild
  • PKGBUILD

• testing OSS
  • tests “git push” for 20000+ upstream OSS repos
Vision: test matrix

- coverage: build/functional/performance tests
- coverage: full OSS stack
- coverage: docker/VM/HW @ arm, x86, riscv

components*parameters explosion
The matrix

workloads

X

OS

X

machine

packages
+ overlayfs
+ ro rootfs

job

build test benchmark borrow bisect

mysql, ceph, nginx, fluentd, ...

coreutils, bash, vim, git, curl, ...

gcc llvm rust go crystal ruby python

glibc

musl libc

kernel

linux POSIX kernel

docker testbox

container

VM testbox

qemu

HW testbox

x86 arm risc-v

job.yaml

script_name:
  param1: ...
  param2: ...

os: ...
kernal: ...
testbox: ...

workloads

X

OS

X

machine

packages
+ overlayfs
+ ro rootfs

job

build test benchmark borrow bisect

mysql, ceph, nginx, fluentd, ...

coreutils, bash, vim, git, curl, ...

gcc llvm rust go crystal ruby python

glibc

musl libc

kernel

linux POSIX kernel

docker testbox

container

VM testbox

qemu

HW testbox

x86 arm risc-v

job.yaml

script_name:
  param1: ...
  param2: ...

os: ...
kernal: ...
testbox: ...
## Current status

<table>
<thead>
<tr>
<th>OS distro</th>
<th>openEuler, CentOS, Debian, Archlinux</th>
</tr>
</thead>
<tbody>
<tr>
<td>arch</td>
<td>ARM, x86, RISC-V</td>
</tr>
<tr>
<td>testbox</td>
<td>HW, VM, container</td>
</tr>
<tr>
<td>build tests</td>
<td>20000+ PKGBUILD</td>
</tr>
<tr>
<td>kernel tests</td>
<td>randconfig builds</td>
</tr>
<tr>
<td>benchmarks</td>
<td>reuse/extend lkp-tests</td>
</tr>
<tr>
<td>os</td>
<td>os_arch</td>
</tr>
<tr>
<td>----------</td>
<td>---------</td>
</tr>
<tr>
<td>opensuse</td>
<td>aarch64</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>opensuse</td>
<td>x86_64</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>centos</td>
<td>aarch64</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>opensuse</td>
<td>x86_64</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>debian</td>
<td>aarch64</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>archlinux</td>
<td>aarch64</td>
</tr>
</tbody>
</table>
testbox providers

server: ipxe boot
others: provider scripts

VM testbox

provider: qemu
template: kvm
nr_node: 1
nr_cpu: 1
memory: 1G
Vision: help developers get bug fixed

• auto bisect
• submit jobs for reproducing/debugging
• ssh login to test environment
Auto detect/bisect/report regressions

Compass CI runs common bisect flow for

• any project
• any error id

boot.fail: 1
ltp.open13.fail: 1
kvm-unit-tests.vmx.fail: 1
xfstests.generic.452.fail: 1
dmesg.BUG:scheduling_while_atomic: 1
dmesg.WARNING:at_mm/page_alloc.c:free_area_init_node(): 1
stderr.mkfs.xfs.pwrite_failed:Invalid_argument: 1

Ingo Molnar when asked how much time it saves him:

_a lot_.

With Git bisect it's a breeze

it's invaluable because there are bugs I would never even _try_ to debug if it wasn't for git bisect.

git bisect is unconditional goodness - and feel free to quote that.
Vision: help developers get bug fixed

• auto bisect
• submit jobs for reproducing/debugging
• ssh login to test environment
QA farm developers

• Problem: reproduce bug
  ➔ resubmit job to re-produce

• Problem: local arch ≠ remote arch
  ➔ one job to borrow machine
  ➔ access various archs in QA farm
single command to borrow a machine + deploy OS + ssh login

$ submit -m -c testbox=... os=openeuler ... borrow-1h.yaml

hi1003@account-vm ~% submit -m -c borrow-1h.yaml testbox=vm-2p16g
per-job kernel+OS deployment

```
job.yaml

```

```
# ipxe
initrd http://$IP:$PORT/os/debian/aarch64/sid/initrd.lkp
initrd http://$IP:$PORT/initrd/lkp/latest/lkp-aarch64.cgz
initrd http://$IP:$PORT/job_initrd_tmpfs/65110/job.cgz
kernel http://$IP:$PORT/os/debian/aarch64/sid/vmlinuz \
    ro root=\$IP:/os/debian/aarch64/sid

```

```
overlayfs + NFS
```

POST

GET

http://$sched/boot.ipxe/mac/$mac

sched
[queues]
**rootfs access types**

```
job.yaml
kernel:
  os: ...
  os_mount: initramfs|nfs|cifs|local
```

- **boot stage1**
  - initramfs
    - kernel
    - `initramfs initrd`
  - nfs
    - kernel
    - `dracut initrd`

- **boot stage2**
  - overlayfs
    - NFS
  - overlayfs
    - CIFS
  - local disk
debug: modify job, re-submit

```
job.yaml
suite: unixbench
category: benchmark
runtime: 300s
unixbench:
  test: syscall
  nr_task: 1
commit: xxxxxx

testbox: vm-1p2g
os: openeuler
os_arch: aarch64
os_version: 20.03
```

+ testbox: vm-2p8g = try another machine
+ os_version: 20.09 = try another rootfs
+ unixbench.nr_task: 2 = try another parameter
+ commit: yyyyyy = try another commit
+ sshd: = let me login
Vision: ecosystem

• create mutual beneficial test alliance: OSV + IHV + ISV/OSS projects

• standardizing on
  • parameterized jobs
  • normalized result stats
  • ES db index

• build common test case pool and tooling
Reusing test workloads

One httpd benchmark could be reused by
- upstream nginx/apache/... web server projects
- gcc/llvm, qemu/kernel projects
- OSV, HW vendors

$ compare os=openeuler --dimension os_version

<table>
<thead>
<tr>
<th>os_version</th>
<th>metric</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>fails:runs</td>
</tr>
<tr>
<td>0:2</td>
<td>+25.0%</td>
</tr>
<tr>
<td>2:2</td>
<td>-100.0%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>os_version</th>
<th>metric</th>
</tr>
</thead>
<tbody>
<tr>
<td>20.03</td>
<td>last_state.test.deploy-csi.exit_code.127</td>
</tr>
<tr>
<td></td>
<td>last_state.test.sleep.exit_code.1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>os_version</th>
<th>metric</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>%stdev</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>503056.00</td>
<td>-22.3%</td>
</tr>
<tr>
<td>222398.00</td>
<td>-25.3%</td>
</tr>
<tr>
<td>3840.00</td>
<td>+5254.2%</td>
</tr>
<tr>
<td>116.00</td>
<td>+4.4e+5%</td>
</tr>
</tbody>
</table>
Reusing the tools

unified data format ➞ unified tool chain
- bisect
- report
- compare
- visualize, accounting ➞

ISV, OSV, IHV come together, share and improve the same tool chain

original output (iostat)

```
avg-cpu: %user %nice %system %iowait %steal %idle
          0.85  5.45  2.00  0.36  0.00  91.35
```

normalized metrics

```
iostat.cpu.user: 0.85
iostat.cpu.nice: 5.45
...
```

add to job.yaml ➔ save to ES db ➔ standardized tool chain
Easy to deploy

- micro-services in
- 60+ containers

- minimal deploy in 1 VM
  - client
  - db + services
  - docker testboxes
Compass CI resources

Homepage
https://compass-ci.openeuler.org

Code
https://gitee.com/wu_fengguang/compass-ci
https://gitee.com/wu_fengguang/lkp-tests
https://gitee.com/wu_fengguang/lab-z9
https://gitee.com/wu_fengguang/upstream/repos
Thank you!