《devkit测试平台安装使用以及与Jenkins集成部署指导手册》

```
一、lkp test 添加测试用例介绍
   样例
   概试
   添加meta.yaml描述文件
   添加job YAML
   添加程序
   添加依赖
  示例-云测工具 (compatibility-test)
二、Jenkins Pipeline 中集成kp test (以云测工具(compatibility-test)为示例)
   2 创建流水线
  3. 执行任务
   4. 查看任务执行状态
   5. 查看报告
   6. lkp test报告内容(以云测工具(compatibility-test)为示例)
二、FAO
1. 联网安装指导
   权限
   安装教程
   证书问题
   make install 遇到的问题
   lkp install 遇到的问题
2. lkp test 任务创建指导
   1. 文件介绍
   2. 必须的文件
3. 离线安装指导
   1.yum源配置
   2.gem 配置
   3.环境变量 配置
   4.测试是否安装成功
```

测试平台使用的是lkp test 工具,以下均已lkp test描述测试平台

一、Ikp test 添加测试用例介绍

样例

如下目录中的文件,完整的添加了一个典型的测试用例memtier:

```
programs/memtier/jobs/memtier-dcpmm.yaml # 在job YAML里指定想跑的programs/params
programs/memtier/jobs/memtier.yaml # 可以预定义很多的jobs
programs/memtier/meta.yaml # memtier描述文件
programs/memtier/PKGBUILD # memtier下载编译
programs/memtier/run # memtier运行脚本
programs/memtier/parse # memtier结果解析
```

如果加的program type属于monitor/setup脚本,则需要放到对应的monitors/, setup/目录下,而非programs/目录。 集中存放monitor/setup脚本,有利于他人查找和复用。

其中jobs/下的YAML文件,定义了memtier的各种常见运行参数、及与其它脚本的组合。 用户要跑其中的一个测试组合,典型步骤如下

```
# 把job YAML从矩阵描述形式分解为一系列原子任务
$ lkp split memtier-dcpmm.yaml
jobs/memtier-dcpmm.yaml => ./memtier-dcpmm-1-cs-localhost-0-8-1-1-65535-never-never.yaml
jobs/memtier-dcpmm.yaml => ./memtier-dcpmm-1-cs-localhost-0-24-1-1-65535-never-never.yaml
# 安装依赖,包括安装meta.yaml里depends字段描述的软件包,以及调用PKGBUILD
$ lkp install ./memtier-dcpmm-1-cs-localhost-0-8-1-1-65535-never-never.yaml
# 运行任务,会调用其中指定的各run脚本,结果保存到/lkp/result/下一个新建的目录里
# 结束后自动运行各parse脚本,提取各结果指标并汇集到stats.json
$ lkp run ./memtier-dcpmm-1-cs-localhost-0-8-1-1-65535-never-never.yaml
```

概述

一个测试用例一般涉及如下部分

为了实现最大的灵活性、可复用性,我们以job-program-param三层模型来组织测试用例。 一个job YAML的典型内容为

```
monitor_program1:
monitor_program2:
```

```
setup_program1:
    param1:
    param2:
setup_program2:
    param1:
...
workload_program1:
    param1:
    workload_program2:
    param1:
    param2:
```

其中每个脚本只做一件事,这样组合起来会很灵活和强大。monitor/setup programs的可复用性就很好。

用户跑一个用例的入口是job,可以自己书写job,也可以使用jobs/目录下预定义的job。 当运行一个job时,kp会找到job中指定的各类programs,以指定的params key/val为环境变量,执行各program。 确切的规则如下

添加meta.yaml描述文件

一个meta.yaml文件描述一个program, 其结构如下

```
metadata:
                       # 程序名
     name:
     name: # 程序名
summary: # 単行描述
description: # 多行/多段详细描述
homepage: # 脚本所调用程序的上游项目的主页URL
e: # monitor|setup|daemon|workload
type:
monitorType:
                             # one-shot|no-stdout|plain
depends:
                             # ruby gem 依赖
     gem:
gem: # ruby gem 依赖
pip: # python pip 依赖
ubuntu@22.04: # ubuntu 22.04的DEB包依赖
openeuler@22.03: # openeuler 22.03的RPM包依赖
pkgmap: # 各OS之间的包名映射,这样我们可以在depends里指定一个OS的完整依赖列表,通过少量包名映射来支持其它OS
archlinux..debian@10:
debian@10.openeuler@22.03: # 以下为两个样例
           dnsutils: bind-utils
cron: cronie
params: # run脚本可以接受的环境变量参数,以下为样例
     runtime:
           type: timedelta
           doc: length of time, with optional human readable time unit suffix
           example: 1d/1h/10m/600s
     ioengine:
           type: str
values: sync libaio posixaio mmap rdma results: # parse脚本可以从结果中提取的metrics,以下为样例
     write_bw_MBps:
           doc: average write bandwidth kpi: 1 # weight for computing performance index; negative means the larger the worse
```

添加job YAML

一般我们需要主要跑一个type=workload的program,同时再跑一些type=monitor/setup/daemon的programs,加上它们的参数,构成一个完整的测试用例。 我们用一个个的job YAML来描述这些测试用例。

所以预定义job YAML大体上可以按workload来组织,放在路径下

```
programs/$workload/jobs/xxx.yaml
```

当然也可以按更大粒度来组织,比如场景、测试类型等分类,此时可以放在路径下

```
jobs/$test_scene/xxx.yaml
jobs/$test_class/xxx.yaml
```

以上预定义jobs的搜索路径,lkp框架代码都支持。具体path glob pattern是

```
programs/*/jobs/*.yaml
jobs/**/*.yaml
```

添加程序

Job YAML中引用的programs,需要您预先写好, kp会在如下路径搜索其文信息/脚本:

```
1st search path 2nd search path programs/$program/meta.yaml programs/$program/{run,parse} programs/$programs/$package/PKGBUILD programs/**/PKGBUILD-$package
```

程序一般添加到 programs/\$program/ 目录下,具体添加以下几个脚本

```
programs/$program/meta.yaml # 描述文件
programs/$program/run # 接收/转换环境变量传过来的参数,运行目标程序
programs/$program/parse # 解析结果(一般是run的stdout),输出metrics (YAML key/val)
programs/$program/PKGBUILD # 下载编译安装run调用的目标程序
tests/$program => .../programs/$program/run # 创建符号链接 保持兼容
```

其中PKGBUILD仅必要时添加。parse一般在program type=monitor/workload时才需要。

一般一个program一个目录。但有时候client/server类型的测试,把workload+daemon programs放在一起比较方便。 此时可以参照sockperf,把 sockperf-server daemon以如下方式添加到sockperf workload目录下:

```
programs/sockperf/meta-sockperf-server.yaml
programs/sockperf/run-sockperf-server
```

添加依赖

一个program的依赖表述为

这里定义了两类依赖 1) OS自带的包 2) 需要从源码下载编译的包 当OS包含package1/package2时,kp框架可自动安装对应的rpm/deb; 如果没有,再使用PKGBUILD-xxx构建出包。

例如,在debian 10中,lkp install会执行

```
apt-get install $package1 $package2
```

在在centos 8中,lkp install会执行

```
yum install $package1
makepkg PKGBUILD-$package2 # 从源码下载编译
```

如您希望强制从源码编译下载,无论所在OS是否包含RPM/DEB包,那么可以通过指定PKGBUILD依赖

```
depends:
PKGBUILD:
- $package1
```

那么lkp install会无条件编译\$package1

```
注意,PKGBUILD语义上对应一个package,而不是对应
program。 这两者语义上不同,虽然很多时候两者内容是一样的。当内容一样时,比如
```

```
programs/$program/PKGBUILD-$package
```

也可以写为简化形式

```
programs/$program/PKGBUILD # when $package=$program
```

注意,PKGBUILD文件名及其内部depends/makedepends字段里的\$package使用的是archlinux包名。 所以其它OS缺失此包,或者有此包,但 是名字不一样的话,需要配置对应的pkgmap包名映射,或者加上OS后缀,比如

```
makedepends_debian_11=(lam4-dev libopenmpi-dev libmpich-dev pvm-dev)
```

示例-云测工具 (compatibility-test)

1. 在programs 文件夹下创建compatibility-test文件夹,里面至少要包含以下几个文件,其余文件可以根据需求自行决定是否添加 programs/compatibility-test/jobs/compatibility-test.yaml# 预定义compatibility-test的job,需要与文件夹名字一致 programs/compatibility-test/meta.yaml# compatibility-test描述文件 programs/compatibility-test/run# compatibility-test运行脚本

2. 文件内容详情

programs/compatibility-test/jobs/compatibility-test.yaml:

```
suite: compatibility-test # 项目介绍 category: functional # 项目类型(functional是只跑run脚本不会同时拉monitor监控应用,如果想同时使用monitor请写 benchmark) file_path: /home/lj/test/compatibility_testing/Chinese # run 脚本的输入参数,如果有多个请写在下面 compatibility-test:后 compatibility-test: # run 脚本的输入参数, # 如果同一参数有多个不同值需要运行,可以按照以下方式参考 xxx:
    parameter1:
        - value1
        - value2

parameter2:
        - value1
        - value2
```

programs/compatibility-test/meta.yaml:

```
metadata:
    name: compatibility-test # 名字
    summary: A program can run some basic tests # 这个项目的总结
    description: run compatinility test and generate the report # 这个项目的介绍
    homepage: https://gitee.com/openeuler/devkit-pipeline # 项目的网址
    type: workload # 项目类型
    depens: # 项目依赖
    params: # 需要的参数
    results: # 需要对结果进行处理
```

programs/compatibility-test/run:

```
# 这个文件是shell脚本,当使用lkp test命令去运行指定yaml的时候会把yaml的参数传入到run 文件中,请根据各自项目自行写此文件
#!/bin/bash
set -e
ct_sh_path=${HOME}/.local/compatibility_testing/Chinese/compatibility_testing.sh
cloud_jar=${HOME}/.local/compatibility_testing/cloudTest.jar
cd $(HOME)/.local/compatibility_testing/Chinese/
#sh $ct_sh_path
#java -jar $cloud_jar &
#sleep 15
#jar_pid=$!
#curl --location --request GET 'http://127.0.0.1:10037/api/v1/report?
savePath=/'${HOME}'/.local/compatibility_testing/Chinese/log.json&file=/'${HOME}'/.local/compatibility_testing/Chinese/log.json&file=/'${HOME}'/.local/compatibility_testing/template.html.bak
/${HOME}/.local/compatibility_testing/template.html
cd $(HOME)/.local/compatibility_testing/femplate.html
cd $(HOME)/.local/compatibility_testing/femplate.html
python3 ${HOME}/.local/compatibility_testing/json2html.py
```

3. 必要步骤

在完成此文件夹的创建后,依然还需要两步操作去让lkp命令找到指定的运行文件

```
# 第一步 运行lkp slpit 命令去分隔jobs里面写的yaml文件,他会根据run文件以来的每个参数不同的输入值分成多个可执行的 yaml文件,例如 lkp split programs/compatibility-test/jobs/compatibility-test.yaml # 云测工具会得到输出 programs/compatibility-test/jobs/compatibility-test.yaml => ./compatibility-test-defaults.yaml, 当我们每次更新jobs下面的yaml文件的输入参数后都需要重新运行 lkp split命令 # 当我们lkp run的时候就要运行这个分隔后的yaml文件(在云测工具就是compatibility-test-defaults.yaml) # 第二步 需要增加一个软连接 ln -s xxx/lkp-tests/programs/compatibility-test/run xxx/lkp-tests/compatibility-test
```

二、Jenkins Pipeline 中集成lkp test (以云测工具(compatibility-test)为示例)

1 groovy 代码

```
stage('lkp test') {
                                                                                                                                                                                                                    steps {
                                                                                                                                                                                                                                                    script{
                                                                                                                                                                                                                                                                                                                       '===== lkp test ======'
                                                                                                                                                                                                                                                                                  sh '
cp -rf /home/lj/test/compatibility_testing/template.html.bak /home/lj/test/compatibility_testing/template.html # 用于最终生成报告
                                                                                                                                                                                                                                                                                                               \verb|sudo|/home/lj/lkp-tests-master/bin/lkp| run /home/lj/lkp-tests-master/bin/lkp| run /home/lj/lkp| run /home/
master/programs/compatibility-test/compatibility-test-defaults.yaml
                                                                                                                                                                                                                                                                                                               ср
 /home/lj/test/compatibility\_testing/compatibility\_report.html \ ./compatibility\_report.html \ 
                                                                                                                                                                                                                                                                         sh(script: "sudo bash
  /home/lj/test/compatibility_testing/Chinese/test_result.sh", returnStdout:true).trim() # 用来根据报告结果设置流水线结果
                                                                                                                                                                                                                                                    always
                                                                                                                                                                                                                                                                                  publishHTML(target: [allowMissing: false,
                                                                                                                                                                                                                                                                                                                                                                                                                                                                        alwaysLinkToLastBuild: false,
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     : true,
: '.',
                                                                                                                                                                                                                                                                                                                                                                                                                                                                        keepA11
                                                                                                                                                                                                                                                                                                                                                                                                                                                                       reportDir
                                                                                                                                                                                                                                                                                                                                                                                                                                                                        reportFiles
```

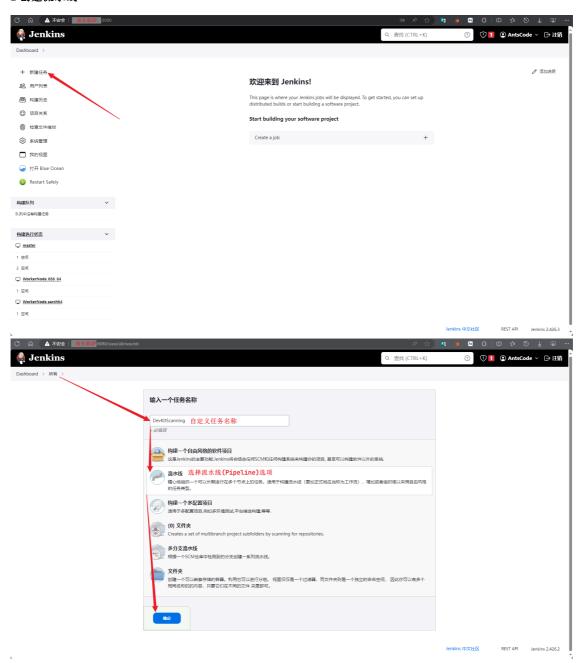
```
'compatibility_report.html',
Report']

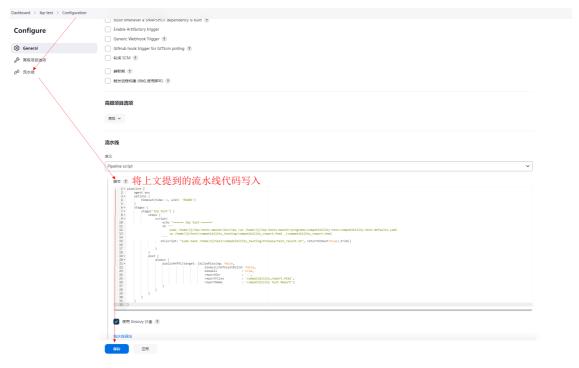
}

reportName : 'compatibility test

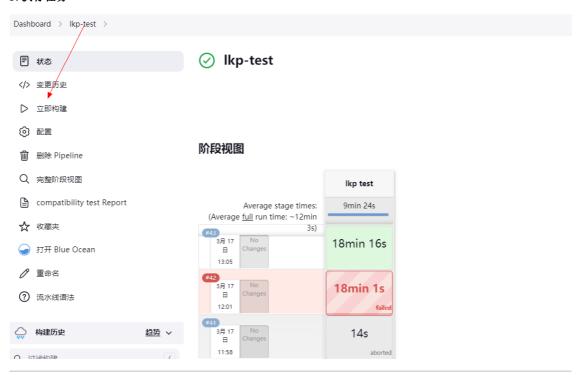
}
```

2 创建流水线

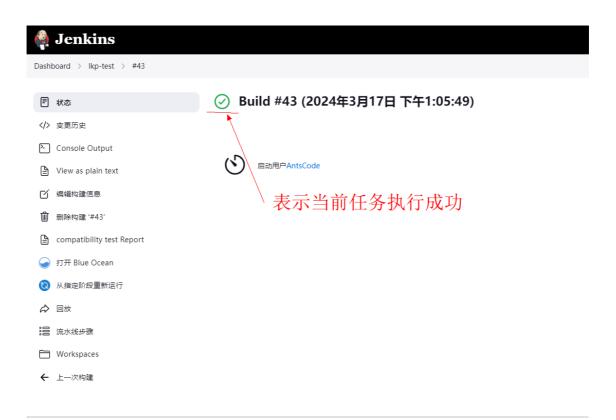




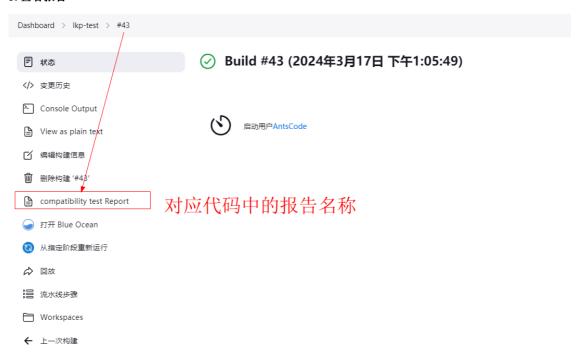
3. 执行任务



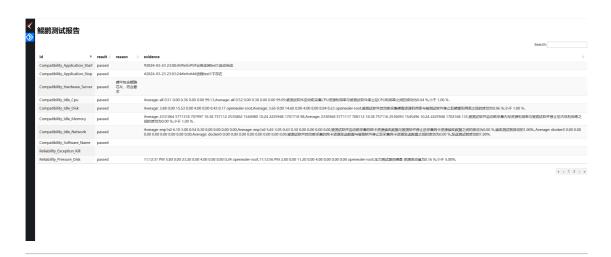
4. 查看任务执行状态



5. 查看报告



6. lkp test报告内容(以云测工具(compatibility-test)为示例)



二、FAQ

1. 联网安装指导

权限

lkp test 运行需要root用户运行,所以在自动化的过程中有一个切root的操作

安装教程

证书问题

[解决方式]: https://developer.baidu.com/article/details/2821747

make install 遇到的问题

1. 未从更新后的镜像源拉取资源

```
bash sbin/install-dependencies.sh
Installing packages ruby rubygems make gcc diffutils util-linux lftp hostname sudo
gzip git rubygem-json rubygem-bundler gcc-c++ ruby-devel rubygem-rake rpm-build
Installing ruby gems...
source https://rubygems.org/ not present in cache
```

[解决方式]:

将 lib/install.sh里第8行替换为可用的镜像源,以及可以屏蔽掉10-12行中不可用的镜像源

2. 卡死报错

```
[Potlins:H]
[rootimaster0] lkp-tests]# make install
bash sbin/install-dependencies.sh
Installing packages ruby rubygems make gcc diffutils util-linux lftp hostname sudo gzip git rubygem-json rubygem-bundler gcc-c++ ruby-devel rubygem-rake rpm-build
Installing ruby gems..

source <a href="http://mirrors.aliyun.com/rubygems/">http://mirrors.aliyun.com/rubygems/</a> already present in the cache
```

```
Error fetching https://mirrors.tuna.tsinghua.edu.cn/rubygems/:
too many connection resets (https://mirrors.tuna.tsinghua.edu.cn/rubygems/specs.4.8.gz)
https://mirrors.ustc.edu.cn/rubygems/ added to sources
Error fetching https://gems.ruby_china.com/:
SSL_connect returned=1 erron=0 state=error: wrong version number (https://gems.ruby-china.com/specs.4.8.gz)
Traceback (most recent call last):
2: from /usr/bin/bundle:23:in `main>'
1: from /usr/share/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/rubygems/ru
```

[解决方式]: 安装bundler 首先运行

```
[root@master01 lkp-tests]# cat Gemfile.lock | grep -A 1 "BUNDLED WITH"
BUNDLED WITH
2.2.33
[root@master01 lkp-tests]# yum install bunder -v '2.2.33'
Loaded plugins: bullddep, changelog, config-manager, copr, debug, debuginfo-install, download, generate_completion_cache, needs-restarting, playground, repoclosure, repodiff, repograph, repomenage, reposync
manage, reposync
VIVI version: 4.2.23
```

安装相同版本

gem install bundler -v 2.2.33

3. 为更新gem缓存导致更新的镜像源未生效

```
Trootsteamer'st lay-testif mide install
bash biblin'stall dependences sh
Installing peckages ruby rubygems make gcc diffutils util-linux lftp hostname sudo grip git rubygem-json rubygem-bundler gcc-c++ ruby-devel rubygem-rake rpm-build
Installing peckages ruby rubygems are good into the cache
Onot run Bundler as root. Bundler can ask for sudo if it is needed, and installing your bundle as root will break this application for all non-root users on this machine.

//usr/ahar/gems/gems/psych-3-0.2/ll/bysych.rb:223: warning: alteredy initialized constant Psych::IBIAML_VENSION
//usr/ahar/grobypsych.bi2-3-0.2/ll/bysych.rb:234: suming: alteredy initialized constant Psych::IBIAML_VENSION
//usr/ahar/rubypsych.rb:234: warning: previous definition of FALLBACK
//usr/ahar/ruby/psych.rb:234: warning: previous definition of FALLBACK was neer
fetching source index from https://rubygems.org/
//usr/ahar/rubygems.org/specs.4.8.gz)>

Retrying fetcher due to error (3/4): Bundler::HTTPError Could not fetch specs from https://rubygems.org/ due to underlying error <SL_connect returned-1 error=0 state=error: wrong version number (https://rubygems.org/specs.4.8.gz)>

Retrying fetcher due to error (3/4): Bundler::HTTPError Could not fetch specs from https://rubygems.org/ due to underlying error <SL_connect returned-1 error=0 state=error: wrong version number (https://rubygems.org/specs.4.8.gz)>

Retrying fetcher due to error (4/4): Bundler::HTTPError Could not fetch specs from https://rubygems.org/ due to underlying error <SL_connect returned-1 error=0 state=error: wrong version number (https://rubygems.org/specs.4.8.gz)>
```

[解决方式]:

```
# 运行命令,删除无效镜像源
gem source
# 在更新镜像源后需要更新gem缓存,让更新的镜像源生效
gem source -u
```

4. bundle镜像源配置问题

```
[rost@master01 lkp-tests]# make install
bash shin/install-dependencies.sh
Installing pendencies.sh
Installing pendencies.
```

[解决方式]:

替换bundle镜像源

bundle config mirror.https://rubygems.org https://mirrors.aliyun.com/rubygems

5. 安装超时报错

```
Bandar::HTPTror: Could not download gem from https://mirors.aliyun.com/rubygems/ due to underlying error <Error::ECONNEFUSED: Failed to open TCP connection to 90.253.6.207:8888 Bandar::HTPTror: Could not download gem from https://mirors.aliyun.com/rubygems/gems/zeltwerk-2.6.5.gem) / User/local/share/gems/gems/bundler-2.2.33/lib/bundler/rubygems_integration.pts945 in: rescue in download_gem? / User/local/share/gems/gems/bundler-2.2.33/lib/bundler/rubygems_integration.pts945 in: rescue in download_gem? / User/local/share/gems/gems/bundler-2.2.33/lib/bundler/rubygems_integration.pts945 in: rescue in download_gem? / User/local/share/gems/gems/bundler-2.2.33/lib/bundler/source/rubygems_integration.pts945 in: rescue in download_gem? / User/local/share/gems/gems/bundler-2.2.33/lib/bundler/source/rubygems_integration.pts945 in: rescue in download_gem? / User/local/share/gems/gems/bundler-2.2.33/lib/bundler/source/rubygems_integration.pts945 in: rescue in download_gem? / User/local/share/gems/gems/bundler-2.2.33/lib/bundler/installer/gem_installer_installer_installer.pts16: in: rescue in downstaller_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installer_installe
```

[解决方式]:

安装时间过长,连接冲断,重新运行make install即可解决

lkp install 遇到的问题

1. 报错,系统不支持

```
[root@master01 lkp-tests]# lkp install
Not a supported system, cannot install packages.
```

[解决方式]:

```
环境变量中增加 LKP_SRC,路径和LKPPATH一样exportPATH=LKP_PATH 一样 export PATH=LKP ATH—样exportPATH =PATH/home/lj/lkp-tests/sbin/home/lj/lkp-tests/bin/home/lj/lkp-tests/sbin/home/lj/lkp-tests/bin export LKP_PATH=/home/lj/lkp-tests export LKP_SRC=/home/lj/lkp-tests
```

2. lkp test 任务创建指导

1. 文件介绍

 $doc/add\text{-}testcase.\textit{zh}.md\cdot Fengguang/lkp\text{-}tests\text{-}Gitee.com$

2. 必须的文件

run (可执行脚本)

jobs 文件夹以及文件夹内需要包含一个与program同名的yaml文件

lkp split xxx.yaml # 这个yaml是jobs文件夹里的,在哪里执行这个命令,分割出来的任务就会在哪lkp run xxxx.yaml # 这个yaml是上一步分割完后生成的yaml

3. 离线安装指导

1.yum源配置

请配置everything的yum源

https://repo.huaweicloud.com/openeuler/openEuler-20.03-LTS/ISO/aarch64/

2.gem 配置

去https://gems.ruby-china.com/ 下载以下gem依赖 bundeler 2.2.33, diff-ks 1.5.0, minitest 5.15.0 concurrent-ruby 1.1.10, docile 1.4.0, rehardet 1.8.0, gnuplot 2.6.2, parallel 1.22.1, public_suffix 4.0.7, regexp_parser 2.6.0, rexml 3.2.5, ast 2.4.2, rainbow 3.1.1, rspec-support 3.12.0, ruby-progressbar 1.11.0, unicode-display_width 2.3.0, git 1.7.0, simplecov_json_formatter 0.1.4, simplecov-html 0.12.3, rspec-core 3.12.0, rspec-expectations 3.12.0, rspec-mocks 3.12.0, il 8n 1.12.0, builder 3.2.4, sync 0.5.0, tzinfo 2.0.5, rspec 3.12.0, ci_reporter 2.0.0, ci_reporter_rspec 1.0.0, parser 3.1.2.1, tins 1.31.1, term-ansicolor 1.7.1, rubocop 1.12.1, simplecov 0.21.2, simplecov-rcov 0.3.1

将以上依赖放到 /usr/share/gems/gems
并执行gem install --local 安装以上依赖

3.环境变量 配置

export PATH=\$PATH:kptest路径/kp-tests/sbin:kptest路径/kp-tests/bin:kptestBin:kptest

4.测试是否安装成功

lkp help lkp install