**Googlesheet-Polarion 工具说明**

Shaobo Lv<shalv@redhat.com>, Wangyang Li<wanli@redhat.com>

2017 年 3 月 16 日

Goolesheet Job 工具主要包括以下三个脚本:

1. /kernel/networking/common/tools/initial.sh: 安装第三方Python 库(不用每次执行),

2. /kernel/networking/common/tools/tools\_polarion/gen\_report.py: 将 Beaker 中 Job 结果解析导入Googlesheets,

3. /kernel/networking/common/tools/tools\_polarion/xunit\_to\_polarion.py: 将Googlesheets中数据结果导入到 polarion.

1 环境安装与配置

该工具需在 **Python2.7** 版本下安装运行

1.1 环境安装

通过 **bash initial.sh** 完成, 其主要功能如下:

Firstly, we have to check that python-pip, gspread and oauth2client have already been installed,

Secondly, we should install pylarion.

1.2 环境配置

配置文件.pylarion 位置和内容如下所示:

**cat ∼/.pylarion**

[ webservice ]

url = https :// polarion . engineering . redhat .com/ polarion

svn\_repo = https :// polarion . engineering . redhat .com/repo

user = <kerberos\_id >

password = <kerberos\_password >

default\_project = <RHEL6 | OtherProject >

logstash\_url = ops -qe -logstash -2. rhev -ci -vms.eng.rdu2. redhat .com

logstash\_port = 9911

2 使用方法

2.1 gen\_report.py

2.1.1 创建新文档

1. 新建一个 Googlesheet 文档,并起名

2. 分享给 testreport-1258@appspot.gserviceaccount.com

3. 添加一个 sheet(第一行 header 需要包括 CASE, TOPO, ROLE, NIC, DRIVER, Beaker RESULT, Final RESULT, JOB, COMMENT, BZ, OWNER, VIEWER)

2.1.2 脚本使用

Usage ([] 表 示 该 参 数 为 可 选 ):  
gen\_report .py [-a] -d <doc name > -s <sheet name > [-c <columnX =valueX , columnY =valueY, ... >] -- <options for bkr job -list >  
样 例:  
**python gen\_report .py [-a] -d test\_spreadsheet -s sheet1 -c " OWNER = qding " -- --owner=qding --whiteboard "RHEL -6.8 OVS SS3"**  
参 数 说 明:  
[-a]会 将 结 果 为 Aborted 的 结 果 导 入 , 并 在 Googlesheet 中 Job 栏 用 Hyperlink 函数 表 示 , 可 以 通 "= ISFORMULA ()" 条 件 进 行 过 滤 或 添 加 条 件 格 式 。 未 设 置 则 跳 过 Aborted 的 结果。  
-d <doc name > 指 定 导 入 到 的 Googlesheet 的 文 档 名  
-s <sheet name > 指 定 导 入 到 的 Googlesheet 的 表 名  
[-c <columnX =valueX , columnY =valueY ,... >] -c 是 在 导 入 过 程 中 设 置 某 一 列 的 值 , 例 如 '-c "OWNER=qding, VIEWER = qding "' 会 覆 盖 默 认 设 置 的 OWNER 和 VIEWER 值 , 替 换 为 指 定 值  
-- <option for bkr job -list > -- 参 数 一 定 要 在 所 有 参 数 的 最 后 , -- 后 的 所 有 内 容 会 直 接 传 递 给 bkr job -list 来 筛 选 从 Beaker 中 需 要 导 入 的 Job , 具 体 参 数 参 照 "bkr job -list --help" ,常 用 的 有 :

-o OWNER , --owner = OWNER # 属 于 OWNER 的 Job

-w STRING , --whiteboard = STRING # whiteboard 内 包 含 STRING 的 Job

--min -id= MIN\_ID #Job id 最 小 值  
 --max -id= MAX\_ID #Job id 最 大 值

2.2 xunit\_to\_polarion.py

2.2.1 脚本使用

Usage ([] 表 示 该 参 数 为 可 选 ):

从 googlesheet 导 入 :

xunit\_to\_polarion .py -u <user> [--password=<kerberos password>] -d <doc\_name > -s <sheet name > -p <polarion project > -i <plannedin > -t <title prefix > [-n]

从 csv 文 件 导 入 :

xnuit\_to\_polarion.py -u <user> [--password=<kerberos password>] -f <file\_name > -p <polarion project> -i <planned in> [-n]

样 例:

**python xunit\_to\_polarion.py -u shalv -d RHEL6.9\_Test\_Report -s RC -p RHEL6 -i 6\_9\_RC -t Test\_KN-RUN-RHEL-69-RC**

python xunit\_to\_polarion.py -u shalv -f fail.csv -p RHEL6 -i 6\_9\_RC

参 数 说 明:

-u kerberos user

-d 指 定 导 出 的 Googlesheet 的 文 档 名

-s 指 定 导 出 的 Googlesheet 的 表 名

-f 使 用 本 地 csv 作 为 输 入 , 导 出 结 果 到 polarion

-p 指 定 导 入 到 的 polarion 的 project 名

-i 指 定 本 次 导 入 所 对 应 的 planned\_in 参 数 , 会 传 入 新 创 建 的 test run

-t 指 定 命 名 的 前 缀 , 创 建 test run 时 使 用

[-n] 设 置 则 dry\_run , 只 读 取 并 在 polarion 中 查 询 对 应 case\_id , 不 写 入 polarion

[--password=kerberos password] 指定用户的密码，不设置该参数时默认为～/.pylarion文件中对应密码

*2.3 特别注意*

1. Pylarion配置文件必须放在~/.pylarion处，程序才能正确提取出password，否则需要通过--password=XXXXX参数进行输入，

2. Pylarion配置文件中请填写one and only one default\_project，例如default\_project = RHEL6，

3. 因为该脚本需要借助于Polarion Xunit Importer，在使用xunit\_to\_polarion.py之前，必须手动删除https://polarion.engineering.redhat.com/polarion/网站中对应的TestRun(确保删除对应该plannedin的, 不要删除其他plannedin的)，

Screenshot from 2017-03-16 13-54-47.png

4. 在该脚本中，默认设置的case名称前缀为 "KN-TC New-Test case Pylarion"，考虑到参数太多，没有再设置此项。如果有需要，可以在testcase.py文件中的第94行进行设置，例如：case\_title\_prefix = 'XXXXXX，

5. log文件夹下的test.log文件内容不会自动删除，如有必要，可以手动删除，

6. 如果脚本运行过程提示Find more than one case错误，说明之前case建立存在错误。在测试过程中还未有出现过该情况，如果有出现，可以手动删除所有的case，然后重新执行脚本，

7. 如果系统运行过程提示SSL认证错误(ssl.SSLError)，为urlllib2.py存在问题，可以通过执行**bash initial.py**解决(用tools中的urllib2.py覆盖/usr/lib64/python2.7/urllib2.py).

3 开发者文档

3.1 gen\_report.py

3.1.1 设计思路

Beaker 中 Job 的结果都是以 XML 形式进行存储(可以使用"bkr job-result JOB\_ID --prettyxml" 指令来查看), 脚本的主要功能就是解析 XML 中内容取得相应结果,所以,脚本能获得的所有数据都只能在 XML 中,进行扩展时请注意。

3.1.2 使用库

主要使用库有 xml.etree.ElementTree(用于解析 XML 结构), gspread(用于 googlesheet 通讯)。

3.1.3 手动维护项目

代码 275 行左右(若更新代码,请及时更新该文档),需要手动维护一个 list,这个 list 表示的是需要解析Task/TOPO 的 Case 名,list 里面的会解析每一个 task,并对应生成多条记录。而不在该 list 内的 case,永远只会有一个结果,并生成一条记录,其中 Task/TOPO 列为空。

3.2 xunit\_to\_polarion.py

3.2.1 设计思路

填写 Polarion 数据的关键在于：

1. 对于每一个 case, 确定其ID(对应 get\_cases\_ids 函数), 运行结果(对应 get\_cases\_results 函数), comment(对应get\_cases\_comments 函数), 及其所属TestRun的ID(对应gen\_runs\_ids\_for\_cases函数).

2. 记录case信息，在本地生成html格式的TestRuns文件，借助Polarion Xunit Importer(<https://mojo.redhat.com/docs/DOC-1073077>)将信息上传到Polarion.

3.2.2 规范/约定

*1. TestRun\_Title*，约定 TestRun\_Title 为: "Test\_KN-RUN-RHEL-[version]-[test\_stage]-[component]-[driver]-[model]"

(其中"Test\_KN-RUN-REHL-[version]-[test\_stage]" 由传入脚本参数 prefix 给定)

1. version: 测试对应的 RHEL 版本,如 69 对应 RHEL6.9

2. test\_stage: 测试对应阶段,如 Alaph

3. component: 测试对应的组件,如 SCTP,NIC,SRIOV 等

4. driver: 可选项,如果 googlesheet 中有对应数据则追加在尾部

5. model: 可选项,如果 googlesheet 中有对应数据则追加在尾部

例如:Test\_KN-RUN-RHEL-69-Alpha-NIC-be2net-Emulex-OneConnect\_10Gb-be3

*2. 运行结果*: 由 googlesheet 中 Beaker Result 以及 Final Result 共同决定

*3. Case\_ID*，约定 TestCase\_Title 为"KN-TC xxx-Test case: /path/to/case TOPO/FUNC=task\_name",其中

1. xxx 是 Case 所属类型,如 SCTP,NIC,SRIOV 等

2. /path/to/case 是 case 在 git 中路径,推荐不含"/kernel/networking"

3. TOPO/FUNC=task\_name 是指 Case 对应的小 task 的标示

例如:KN-TC NIC-Test case: nic/sanity\_check TOPO=broadcast-test

在上面的约定下,查找 case\_ID 的方法为使用构造 Title 进行 SQL 查询。(具体参考 testcase.get\_case\_id\_by\_sql函数,该函数摘自文件系统组 polarion 工具, 如有更新，可以参考跟进维护)

*4. Comment*，代码中默认使用 COMMENT 列和 BZ 列进行组合,得到 comment.