# python-ssh help

lichun.william@gmail.com

December 24, 2011

## Contents

# 1 python-ssh.py

```
#!/usr/bin/env python
#-*- coding = utf-8 -*-
''' the Parallel batch command running environment'''
'''
    Parallel LHCK utility

    Author       : Wei Lichun<lichun.william@gmail.com>
    Create Date : Thu Sep  8 21:58:04 CST 2011
    Version      : 0.9
    Recent Changes:
        support max parallel thread argument
        support login_name argument
        support regex pattern filter, invert match
'''

'''


This is free software; see the source for copying conditions.
There is NO warranty; not even for MERCHANTABILITY or FITNESS
FOR A PARTICULAR PURPOSE.

'''
import subprocess
import optparse
import sys
import getpass
import copy
import re
import signal

import process_thread
import get_host_list


parser=optparse.OptionParser(
    usage="%prog [-p <max parallel thread number>] -f filename \
    [-l login_name] command",version='%prog 1.2',
    epilog="Report any bugs to lichun.william@gmail.com", prog='python-ssh')
parser.add_option("-p","--parallel"    ,action="store",type="int",
    dest="parallel",default=10,
```

```
    help="max number of parallel threads , default is 10")
parser.add_option("-f","--file"     ,action="store",type="string",
    dest="filename",help="the host file which stores the host list")
parser.add_option("-l","--login_name",action="store",type="string",
    dest="login_name",
    help="Specifies the user to log in as on the remote machine.\
    \n\n\n\nThis also may be specified on a per-host basis in the configuration file
parser.add_option("-X","--extra-arg",action="store",type="string",
    dest="extra_argument",
    help="Extra command-line argument. for example: -o ConnectTimeOut=10")
parser.add_option("-e","--regexp",metavar="PATTERN",action="store",
    type="string",dest="pattern", help="Use PATTERN as the pattern;\
    \n\n\n\nuseful to protect patterns beginning with -.")
parser.add_option("-v","--invert-match",
    action="store_false",dest="invert",default=True,
    help="Invert the sense of matching, to select non-matching lines.")


(options,command)=parser.parse_args()

filename=options.filename
login_name=options.login_name
pattern=options.pattern
parallel=options.parallel
extra_argument=options.extra_argument
sshpass_cmd='sshpass'

if (not filename):
    parser.error ('filename argument is required')

if (not command):
    parser.error ('command argument is required')

hostnames=get_host_list.list_host_from_file(filename)

if(login_name):
    password=getpass.getpass()

def signal_handler(signal, frame):
        print 'Ctrl+C Caught, Exiting..'
        sys.exit(1)

if __name__ == '__main__':
    tasks=[]
```

```
if(not hostnames):
    print ('No Host found from lh util')
    sys.exit(-3)
command.insert(0,'ssh')
host_insert_index=1
if (extra_argument):
    command.insert(1,extra_argument)
    host_insert_index=host_insert_index+1
if(login_name):
    command.insert(1,login_name)
    command.insert(1,'-l')
    command.insert(0,password)
    command.insert(0,'-p')
    command.insert(0,sshpass_cmd)
    host_insert_index=host_insert_index+3
task_group=process_thread.TaskGroup(parallel)
for host in hostnames:
    ssh_command=copy.copy(command)
    ssh_command.insert(host_insert_index,host)
    task_group.add_task(host,ssh_command)

signal.signal(signal.SIGINT, signal_handler)
task_group.start()
size=len(hostnames)
index=0
if(pattern):
    compiled_pattern=re.compile(pattern)
while(index<size):
    index=index+1
    task=task_group.done_queue.get()
    if(pattern):
        if(compiled_pattern.search(task.stdout)):
            matched=True
        else:
            matched=False
        if(not(matched ^ options.invert)):
            print task.key
    else:
            print "\033[0;36;40m",index,"of",size,\
                "\033[0;32;40m: ==========\033[0;33;40m",\
                task.key,"\033[0;32;40m==========\033[0m"
            if(task.stdout):
                    print task.stdout,
```