```
import os
import sys
proj_dir = os.path.dirname(os.path.abspath(__file__))
proj_file = os.path.join(proj_dir, 'util.py')
sys.path.insert(0, proj_file)
import paramiko
from util import *

class controlHost:

    def __init__(self, ipaddr, username, password, port,
key_file='/root/.ssh/id_rsa'):
        self.today = ControlTime().date_today()[0]
        self.ipaddr = ipaddr
        self.username = username
        self.password = password
        self.port = port
        self.key_file = key_file
        self.ssh = paramiko.SSHClient()

self.ssh.set_missing_host_key_policy(paramiko.AutoAddPolicy())
        self.pkey =
paramiko.RSAKey.from_private_key_file(self.key_file)
        self.t = ''

    def sshConn(self):
        try:
            self.ssh.connect(hostname=self.ipaddr,
port=self.port, username=self.username, pkey=self.pkey)
            return {"status": 1, "message": 'sshed login %s
method %s' % (self.ipaddr, 'public_key')}
        except Exception as e:
            try:
                self.ssh.connect(hostname=self.ipaddr,
port=self.port, username=self.username,
```

```
password=self.password)
                return {"status":1, "message": 'sshed login %s
method %s' % (self.ipaddr, 'password')}
            except Exception as e:
                return {"status:":0, "message": 'sshed login
%s %s %s author failed! error log %s' %(self.ipaddr,
self.username, self.password, e)}


    def exeCommand(self, cmd, timeout=300):
        try:
            stdin, stdout, stderr = self.ssh.exec_command(cmd,
timeout=timeout)
            output = stdout.read()
            outerr = stderr.read()
            if output:
                return {"status": 1, "message": output}
            else:
                return {"status": 0, "message": ""}
        except Exception as e:
            print e
            return {"status": 0, "message": ""}


    def sftpConn(self):
        try:
            t = paramiko.Transport((self.ipaddr, self.port))
            self.t = t
        except:
            return {"status": 0, "message": "ip or port
error"}

        try:
            self.t.connect(username=self.username,
pkey=self.pkey)
```

```
            return {"status": 1, "message":'%s connect
success! method public key' % self.ipaddr}
        except:
            t = paramiko.Transport((self.ipaddr, self.port))
            self.t = t
            try:
                self.t.connect(username=self.username,
password=self.password)
                return {"status": 1, "message": '%s connect
success! method password' % self.ipaddr}
            except:
                return {"status": 0, "message": 'connect
failed password or key error!'}


    def sftpFile(self, src, dest, fileName, action):
        sftp = paramiko.SFTPClient.from_transport(self.t)
        if action == 'pull':
            if not os.path.exists(dest):
                os.makedirs(dest)
            sftp.get(src, os.path.join(dest, fileName))
            return {"status": 1, "message":'sftp success get
%s to local' %(':'.join([self.ipaddr, src]))}
        elif action == 'push':
            sftp.put(src, os.path.join(dest, fileName))
##push file
            return {"status": 1, "message": 'sftp success push
%s to remote %s' % (src, ':'.join([self.ipaddr,
os.path.join(dest, fileName)]))}
        else:
            raise TypeError("sftp action must be pull or
push")

    def close(self):
        self.t.close()
```

```
        self.ssh.close()




if __name__  == '__main__':
    import time
    beg = time.time()
    x = controlHost('10.20.7.36', 'sliu', 'XxvnzP', 22)
    x.sftpConn()
    print 1
    x.sshConn()
    print 2
    x.exeCommand('mkdir -p /data/sliu/2018-03-06/
gvtest7.vivo.com.cn')
    print 3
    x.sftpFile('/data3/bs/2018-03-06/gvtest7.vivo.com.cn/
201803061100_new', '/data/sliu/2018-03-06/
gvtest7.vivo.com.cn', '201803061100_new', 'push')
    print 4
    end = time.time()
    print end - beg
```