

中科院计算机网络信息中心超级计算中心

SCE Client 用户手册

SCGrid

SCE Client 用户手册

1 简介

SCE Client 是科学计算网络的命令行客户端软件，通过客户端软件可以访问网格环境内的计算资源，完成提交作业、下载结果文件等工作。

2 SCE 命令

SCE 命令参数中，[]包含的参数是可选参数，|代表从多个参数中选出一个。类似 ujid[,ujid]格式的参数，“ujid1 ujid2”代表 ujid1 和 ujid2 号作业，“ujid1, ujid2”代表 ujid1 至 ujid2 号作业。

sce 命令后加 --help 可以显示命令帮助信息。

2.1 作业号

作业号分为两种:gid(Global Job ID)和 ujid(User Job ID)。其中 gid 是网格系统为用户提交的作业分配的全局作业号,它是唯一的,每个 gid 都是 19 位。ujid 则是为了方便用户使用而设置的简单的短作业号,用户直接使用 ujid 即可对相应作业进行操作。简单起见,以下命令在用到作业号时均使用 ujid。

2.2 HOST 和 DIR

在 sce 中,可以通过环境变量 HOST 来设置默认的远程主机的名字。用户可以将使用频率最多的主机设置成为默认远程主机。实现命令如下: set HOST=hpcname。

在设置之前,用户可以运行 set HOST 查看是否已经设置了默认远程主机,也可以运行 unset HOST 来撤销远程主机的设置。

用户可以通过--dir 来指定远程目录,当用户没有指定时,缺省的远程目录为远程的 \${HOME} 目录加上当前目录去除 \${HOME} 之后的目录。

2.3 Linux 常用命令

在 sce 环境中,在 sce 环境中按 Tab 键可以自动列出当前支持的所有命令列表,支持常用的 Linux 命令,用法不变,主要包括: cat, cd, chmod, clear, cp, dos2unix, date, df, du, echo, grep, head, id, ls, less, ln, mkdir, man, more, mv, passwd, pwd, rm, sleep, tail, tar, top, touch, unzip, vi, vim。其中诸如 vi, cat 等文件编辑类命令,只可以对本地文件进行编辑。

2.4 作业管理命令

bsub 命令 提交作业

格式: bsub [-n cpucore] [-q queueName] [-W walltime] [-h hpcName] [-g gridnode] [-d dirName] [-o outFile] [-e errfile] [-v version] [--host hpcName] [--dir workingdir] [--si stageInFile] [--so stageOutFile] [--jn jobName] [other_options_for_LSF/PBS] [-f multipleSubfile] app_name [argument]...

参数:

[-n cpucore]	指定需要的 CPU 核数
[-q queueName]	指定队列名
[-W walltime]	指定作业完成需要的大概时间, 单位为分钟, 注意 W 为大写
[-h hpcName]	指定 HPC 集群
[-g gridnode]	指定网格节点
[-d dirName]	指定作业运行时的工作目录 (在 HPC 上的目录)
[-o outFile]	指定作业的标准输出文件名
[-e errfile]	指定作业的标准错误输出文件名
[-v version]	指定网格应用版本
[--host hpcName]	指定 HPC 集群, 如不指定则按 2.2 选取 hpcName
[--dir workingdir]	指定工作目录, 如不指定则按 2.2 选取 workingdir
[--si stageInFile]	提交作业所需要的输入文件 (有多个文件用", " 隔开)
[--so stageOutFile]	提交作业的输出文件名 (有多个文件用", " 隔开)
[--jn jobName]	指定作业名
["other_options_for_LSF/PBS"]	指定其他 LSF/PBS 接受的相关命令
app_name	指定网格应用名称
[argument]...	指定应用本身所需的参数
[-f multipleSubfile]	用于批量提交作业, 将作业提交命令写入 multipleSubfile

选择集群和队列有四种方式:

- -q queueName 指定默认 HPC 的队列, 或使用 -q hpcName@queueName 指定任一 HPC 的队列
- -h hpcName 指定某个 HPC, 系统自动选取队列; 可以跟 "any" 作参数, 系统自动选取 HPC 和队列
- -g gridnode 指定某个分中心, 由分中心网格来自动分配集群和队列
- 如果什么都不指定, 系统将去读取环境变量里的 HOST 参数作为默认的远程主机, 将作业提交到该主机上, 如果该环境变量没有被设置, 命令将不被执行。

用户可以使 -d 参数来指定远程工作目录, 如果没有指定, 缺省情况下使用 \${CURRENT_DIR} - \${HOME} 得到用户当前的相对路径, 作为远程的相对路径来使用。

bkill 命令 终止作业

格式: bkill all|ujid[,ujid]...

参数:

all	用户所有未完成的作业
ujid	作业的 ujid 号, 可以指定多个 ujid

<code>[-g gridnode]</code>	查询指定网格节点上所有队列
<code>[-h hpcname]</code>	查询指定 HPC 集群上所有队列
<code>[-u]</code>	查询本用户可以使用的队列
<code>[-a]</code>	查询所有的队列

如果不加任何参数，系统将去读取环境变量里面的 HOST 参数作为默认的远程主机，显示该主机上的队列，如果该环境变量没有被设置，命令将不能执行。

listres 命令 查询资源列表

格式: `listres [-g gridnode] [-h hpcname] [--host hpcname] [-a] [appname [-v version] [-W walltime] [-n cpucore] [-u]]`

参数:

<code>[-g gridnode]</code>	查询指定网格节点上的应用
<code>[-h hpcname]</code>	查询指定 HPC 上的应用
<code> [--host hpcname]</code>	查询指定 HPC 上的应用
<code>[-a]</code>	查询所有应用
<code>appname</code>	需要查询的应用名
<code>[-v version]</code>	根据指定的版本查询应用
<code>[-W walltime]</code>	根据指定 walltime 查询带有某应用的队列
<code>[-n cpucore]</code>	根据指定核数查询带有某应用的队列
<code>[-u]</code>	查询用户可以使用的带有某应用的队列

如果不加任何参数，系统将去读取环境变量里面的 HOST 参数作为默认的远程主机，显示该主机上的应用，如果该环境变量没有被设置，命令将不能执行。

listapps 命令 查询所有资源列表

格式: `listapps`
查看所有应用，相当于 `listres -a`。

sceapp 命令 查询应用相关信息

格式: `sceapp [--host hpcname] [--dir workingdir] appname command [-d] [-v] [-q] [-e] [command_args]`

参数:

<code> [--host hpcname]</code>	指定 HPC 集群
<code> [--dir workingdir]</code>	指定 HPC 端的工作目录
<code>appname</code>	指定应用名
<code>command</code>	应用自带的命令
<code>-d</code>	显示应用的描述信息
<code>-v</code>	显示应用的版本信息
<code>-q</code>	显示应用相关的队列信息
<code>-e</code>	显示应用相关的环境变量
<code>[command_args]</code>	应用相关的命令参数

2.6 文件传输命令

sceput2 命令

上传文件或目录到远程目录

格式: `sceput2 [-j ujid] filename ... [-d des_dir]`

参数:

<code>[-j ujid]</code>	指定作业号, 文件上传至该号作业的 HPC 端工作目录
<code>filename</code>	指定需要上传的目录或文件名, 多个文件以空格分隔
<code>[-d des_dir]</code>	指定上传文件的目录

`sceput2` 可以上传文件或者目录到远程指定目录, 缺省的远程目录为远程的 `{HOME}` 目录加上当前目录去除 `{HOME}` 之后的目录。如果进到一个目录里面, 想要上传当前目录所有文件到远程目录的话, 请使用 “`sceput2.`”, “.” 意味着上传当前目录下所有文件, 也可用 `*` 来代表所有文件。同名的文件和目录不能相互覆盖, 建议在上传之前先用 `scels` 查看远程是否已经存在同名的文件或目录, 如果存在请使用 `scerm` 命令先删除再上传。

sceget2 命令

下载远程文件或目录到当前目录

格式: `sceget2 [-c|-f|-l] [-u username] [-j ujid] filename ... [-d des_dir]`

参数:

<code>[-c]</code>	指定作业的缓存目录
<code>[-f]</code>	指定作业的系统目录
<code>[-l]</code>	指定作业的 HPC 端工作目录
<code>[-u username]</code>	指定用户名
<code>[-j ujid]</code>	指定作业号
<code>filename</code>	指定需要下载的目录或文件名, 多个文件以空格分隔
<code>[-d des_dir]</code>	指定下载文件的目录

`sceget2` 可以下载远程指定文件或者目录到当前目录, 缺省的远程目录为远程的 `{HOME}` 目录加上当前目录去除 `{HOME}` 之后的目录。如果想下载远程目录下所有文件到当前目录的话, 请使用 “`sceget2.`”, “.” 意味着下载远程目录下所有文件 (也可用 `*` 来代表所有文件)。同名的文件和目录不能相互覆盖, 建议在下载之前先查看本地是否已经存在同名的文件或目录, 如果存在请先删除再下载。

2.7 远程文件管理命令

远程文件管理命令与本地文件管理命令相对应, 在本地文件命令前面加上 `sce` 即为远程文件命令, 主要支持的命令有 `scels`, `scerm`, `scmkdir`, `scecat`, `sceln`, `scevi`, `scetar`, `scetail`。与在本地执行 `linux` 命令相比, 远程执行需要增加 `--host` 参数, 使用方式请参照 `scels` 命令。如果用户没有指定 `--host` 参数, 将去读取环境变量里面的 `HOST` 参数作为默认的远程主机, 如果该环境变量没有被设置, 命令将不能执行。

scels 命令

列出远程目录

格式: `scels [linux_ls_options_or_args]... [--host hpcname] [--dir workingdir]`

参数:

<code>[linux_ls_options_or_args]</code>	对应 <code>ls</code> 的各项参数
<code>[-- host hpcname]</code>	指定要操作的远程主机的名字

`[--dir workingdir]` 指定要操作的远程主机上的目录
`scels` 命令对应于传统 linux 命令 `ls`, 其作用相同, 只是在前面加上 `sce`。

scerm 命令 删除远程目录或文件

格式: `scerm [linux_rm_options_or_args]... [--host hpcname] [--dir workingdir]`

参数:

`[linux_rm_options_or_args]` 对应 `rm` 的各项参数
`[-- host hpcname]` 指定要操作的远程主机的名字
`[--dir workingdir]` 指定要操作的远程主机上的目录

`scerm` 命令对应于传统 linux 命令 `rm`, 其作用相同, 只是在前面加上 `sce`。

scemkdir 命令 创建远程目录

格式: `scemkdir [linux_mkdir_options_or_args]... [--host hpcname] [--dir workingdir]`

参数:

`[linux_mkdir_options_or_args]` 对应 `mkdir` 的各项参数
`[-- host hpcname]` 指定要操作的远程主机的名字
`[--dir workingdir]` 指定要操作的远程主机上的目录

`scemkdir` 命令对应于传统 linux 命令 `mkdir`, 其作用相同, 只是在前面加上 `sce`。

scecat 命令 查看远程文件内容

格式: `scecat [linux_cat_options_or_args]... [--host hpcname] [--dir workingdir]`

`scecat ujid filename`

参数:

`[linux_cat_options_or_args]` 对应 `cat` 的各项参数
`[-- host hpcname]` 指定要操作的远程主机的名字
`[--dir workingdir]` 指定要操作的远程主机上的目录
`ujid` 指定作业 `ujid` 号
`filename` 指定要查看的文件名

`scecat` 命令对应于传统 linux 中 `cat` 命令, 其作用相同, 只是在前面加上 `sce`。

`scecat ujid filename` 可以查看 `scesh2 -c ujid` 列出的文件。

scecat2 命令 查看远程文件内容

格式: `scecat2 [-h|-t|-m] [-u username] [-c|-f|-l] ujid filename [num1[, num2]]`

参数:

`[-h]` 从开始显示文件的前 `num1` 行, 不指定 `num1` 则默认为 10 行
`[-t]` 从末尾显示文件的后 `num1` 行, 不指定 `num1` 则默认为 10 行
`[-m]` 显示文件的 `num1` 至 `num2` 行, 不指定默认显示开头 10 行
`[-u username]` 指定用户名
`[-c]` 指定作业的缓存目录
`[-f]` 指定作业的系统目录
`[-l]` 指定作业的 HPC 目录
`ujid` 指定作业号
`filename` 指定文件名

scetail 命令

查看远程文件的后若干行

格式: `scetail [linux_tail_options_or_args]... [--host hpcname] [--dir workingdir]`

参数:

<code>[linux_tail_options_or_args]</code>	对应 tail 的各项参数
<code>[-- host hpcname]</code>	指定要操作的远程主机的名字
<code>[--dir workingdir]</code>	指定要操作的远程主机上的目录

scetail 命令对应于传统 linux 中 tail 命令，其作用相同，只是在前面加上 sce。

scetar 命令

对远程文件或目录进行打包

格式: `scetar [linux_tar_options_or_args]... [--host hpcname] [--dir workingdir]`

参数:

<code>[linux_tar_options_or_args]</code>	对应 tar 的各项参数
<code>[-- host hpcname]</code>	指定要操作的远程主机的名字
<code>[--dir workingdir]</code>	指定要操作的远程主机上的目录

scetar 命令对应于传统 linux 中 tar 命令，其作用相同，只是在前面加上 sce。

2.8 编译命令

SCE 客户端中的编译命令由软件包 GCAide (Grid Compiler Aide) 提供。GCAide 旨在提供网格编译环境的说明和配置，辅助用户完成程序编译。GCAide 提供了两个命令，scelib 和 scemake。scelib 命令可以查看到当前网格环境中各节点的编译环境安装情况，包括编译器和程序库的软件包名称、版本、安装路径以及编译命令/链接选项说明。scemake 命令实现程序的编译过程，可以保持程序原有的编译方式，包括 make 方式、configure+make 方式、shell 脚本方式，以及直接的编译命令方式。通过这两个命令，GCAide 可以从三个不同层次辅助用户完成程序编译，从而满足不同层次用户的需求：

1、手工设置编译环境。

高级用户可以通过 scelib 命令查看所需的编译器和程序库等的在目标编译节点的安装路径，然后根据这些路径自行设置编译环境，或者相应的修改编译脚本文件，最后运行 scemake 命令启动编译。这种方式的优点是可控性强，编译脚本的编写方式保持不变，只是执行编译的命令略有不同；缺点是对用户要求比较高，要求用户对编译环境的设置和调试比较有经验。

2、自动设置编译环境。

普通用户可以通过 scelib 命令查看目标编译节点上是否安装了自己需要的编译器和程序库，在执行 scemake 命令的时候提供一个网格编译描述文件（gcd 文件，具体说明在后面），其中定义了程序编译所需的软件包列表，系统会根据用户指定的软件包列表为用户设置编译环境，然后再执行编译脚本。这种方式的优点是操作简单，用户不需要根据目标节点单独设置环境，也不需要修改编译脚本文件；但是要求用户对程序编译所需的特定软件包版本比较明确，比如只能用 GNU 的 C 编译器，另外同样要求用户需要自己处理编译的各种选项，包括程序库的链接选项。

3、自动选择并设置编译环境。

对于初级用户，GCAide 提供了网格编译宏，使用网格编译宏编写编译脚本文件，用户可以不关心编译器的选择，头文件库文件路径的设置，以及程序库链接选项的编写，这些都由系统自动替换完成。这种方式的优点是对用户要求比较低，用户只需要知道程序编译需要使用了 C 编译器还是 Fortran 编译器，是否使用了 MPI，是否用到了 BLAS 等等类似的接口；缺点是目前替换规则还不够完善，有些情况会处理不了。

在 SCE 网格环境中，编译程序的步骤：

第一步，选择目标编译节点；

第二步，修改相关的环境设置脚本或者编译脚本文件；

第三步，上传所有程序文件，如果已经上传过，只需要上传本地有更新的文件即可；

第四步，运行 `scemake` 命令启动编译，如果程序使用 `make` 方式编译，直接带 `make` 原本的参数即可，如果是其他方式，可通过 `-e` 指定需要执行的脚本；如果编译出错，查错，转第二步。

第五步，执行 `bsub` 提交作业试运行，如果作业执行出错，查错，如果是编译问题，转第二步。

当然，整个过程中，用户随时可以运行 `scelib` 命令查看当前编译环境信息。

下面介绍 `scemake` 和 `scelib` 两个命令的使用格式。

scelib 查看网格环境中编译器和程序库信息

格式： `scelib [-h hpcname] [-n softname] [-k keyword] [-L] [-1]`

参数：

<code>[-h hpcname]</code>	指定查询的网格节点名字，ALL 为全部。
<code>[-n softname]</code>	指定查询的软件名字。
<code>[-k keyword]</code>	指定查询的网格编译宏，比如 BLAS, CC。
<code>[-L]</code>	查看软件的安装路径，包括命令路径、头文件路径和库文件路径。
<code>[-1]</code>	查看软件的使用帮助，可用的编译命令/可用的链接选项。

如果不使用 `-L` 或者 `-1` 选项，则只显示软件的基本信息，包括所在网格节点、软件名字、软件版本和软件支持的网格编译宏。

使用 `scelib -h ALL` 可以查看当前网格环境中所有的软件信息。

scemake 编译程序

格式： `scemake [make_opt_arg | -E gcexecfile] [-F gscriptFile] [-c GCDFile] [-h hpcname] [--host hpcname] [--dir workingdir]`

参数：

<code>[make_opt_arg -E gcexecfile]</code>	<code>make_opt_arg</code> 是 Linux C 编译环境中常用的 <code>make</code> 命令可处理的任意参数；如果程序编译不是使用 <code>make</code> 的方式，可以通过 <code>-E</code> 定义执行的编译命令。
<code>[-F gscriptFile]</code>	网格编译脚本文件，以 <code>.t</code> 为文件后缀，是使用网格编译宏编写的编译脚本文件；不设置该选项表示无网格编译脚本需要处理。
<code>[-c GCDFile]</code>	网格编译描述文件，以 <code>.gcd</code> 为文件后缀，定义了编译所需的特定软件项；如果对使用的编译器或者软件库没有特殊需求，该选项可省略。
<code>[-h hpcname]</code>	指定的节点完成编译，如果不指定，则为当前工作节点。
<code>[-host hpcname]</code>	指定的节点完成编译，如果不指定，则为当前工作节点。
<code>[-dir workingdir]</code>	指定目标机的工作目录，如果不指定，则为当前工作目录。

该命令的使用可以认为是扩展 `make` 命令，如果程序不是使用 `make` 方式编译，可以通过 `-e` 选项指定启动编译过程的脚本。

网格编译脚本

程序编译脚本定义了程序编译执行过程，并且是在软件移植过程中需要根据新的编译环境修改的文件，常见的包括 `shell` 脚本，`Makefile`，`Makefile.INC`，`Makefile.in`，

Configure.in。

网格编译脚本是使用网格编译变量替代了程序编译脚本中的编译命令、路径定义以及链接选项，从而产生的新的文件，并且文件名变更为在原有名字基础上增加.t 后缀。

网格编译变量用@@符号括起来，比如@CC@，@BLAS_DIR@，它由最多三部分字符串组成，以下划线符号“_”作为分割，所有构成字符串均为大写字母和数字。第一块字符串称为“网格编译宏”，比如CC，BLAS，即 scelib 命令查看到的 keyword。最后一块字符串可以为 DIR，INC，或者 LIB，分别代表软件包的根路径、头文件路径和链接库使用选项（包括-L 指定的库文件路径和-l 指定的链接库名称的一个完整的库使用选项）。中间部分用来表示接口支持的语言类型，目前有两个值，C 和 F，比如 BLAS_F_LIB，表示使用 fortran 接口的 blas 库所需要的编译选项。举例如下：

网格编译宏	网格编译变量替换值举例
@CC@	icc
@MPI_INC@	/home_soft/soft/x86_64/mpi/impi/3.2.0.011/include64/
@BLAS_C_LIB@	-L /home_soft/soft/x86_64/lib/mkl/10.1.1.019/lib/64 -lmkl -liomp5 -lpthread
@BLAS@	-L /home_soft/soft/x86_64/lib/mkl/10.1.1.019/lib/64 -lmkl -liomp5 -lpthread

如果 GCAide 在当前记录信息中找到网格编译变量相应的替换项，则替换之，否则将保持原有网格变量字符串不变（这样可以保证不破坏 Makefile.in 文件的变量替换机制）

网格编译描述文件

网格编译描述文件用来定义程序编译所需的特定的软件包，网格编译描述文件的格式定义如下：

```

编译描述：(软件定义项)+
;
软件定义项：网格编译宏 左方括号 软件名项 (逗号 软件名项)* 右方括号
;
网格编译宏：CC | FC | MPI | BLAS | 其他待定义
;
软件名项：软件包名字 [ 左小括号 [等于| 大于| 大于等于] 版本号 右小括号]
;

```

网格编译描述文件格式 (BNF)

其中软件定义项中方括号之间的软件包部分根据顺序存在使用优先级，并且使用优先级逐渐降低。比如 HPL 程序如果对 C 编译器和 BLAS 库的选择有特殊需求，可以形成网格编译描述文件 hpl.gcd，其内容如下：

```

CC[intel]
MPI[openmpi,mpich2]
BLAS[mkl(10.0.1)]

```

2.9sce shell 命令

alias 命令

设定别名

格式: `alias [aliasname[= "aliasvalue"]]`

参数:

`aliasname` 变量名

`aliasvalue` 变量值

例如: `alias list= "bjobs -l"`

执行 `list` 可以实现 `bjobs -l` 的功能。

如果不跟任何参数, `alias` 将会显示全部已经设置的别名。

unalias 命令 **取消设定的别名**

格式: `unalias [name] ...`

参数:

`name` 变量名

set 命令 **设定系统变量**

格式: `set [name[=value]]`

参数:

`name` 变量名(一般全都大写)

`value` 变量值

例如: `set FS_HOME=/usr/local/sce/fs`

可以通过 `${FS_HOME}` 来引用代表 `/usr/local/sce/fs`。

如果不跟任何参数, `set` 将会显示全部已经设置的系统变量。

unset 命令 **取消设定的系统变量**

格式: `unset name`

参数:

`name` 变量名

version 命令 **查询当前 SCE 的版本**

格式: `version [cs|fs]`

查询当前 `cs` 或者 `fs` 的版本信息。

history 命令 **查看历史命令**

exit , quit, bye 命令 **退出 SCE 客户端**

? , help 命令 **SCE 使用帮助**


```
[era@sce testgcc]$ listres
APPLICATION      VERSION
-----
ABACUS
ABACUS           1.0.0
Abinit
Abinit           7.10.2
AMBER
AMBER            14
```

APPLICATION 代表应用名，VERSION 代表版本。

3.4 查看含有 AMBER 应用的集群及队列名

```
[era@sce ~]$ listres -a AMBER
```

```
[era@sce ~]$ listres -a AMBER
APPNAME  VERSION  GRID      HPC      QUEUE     WALLTIME  MAXCPUS  MINCPUS  NJOBS  PEND  RUN
-----
AMBER    14       HuaiRou   era      cpu        370       1100     20       3584   1420  2144
AMBER    14       HuaiRou   era      cpuII      370       13470    24       4812   312   4500
AMBER    14       HuaiRou   era      cpuII      370       13470    24       4812   312   4500
AMBER    14       HuaiRou   era      cpu        370       1100     20       3584   1420  2144
```

APPNAME 代表具体的应用，VERSION 代表应用的版本，GRID 代表网格节点，HPC 代表集群名称，QUEUE 代表该集群的队列，WALLTIME 代表作业在该队列内允许运行的最长时间（以分钟为单位），MAXCPUS 代表队列中可以使用的最大 CPU 核数，MINCPUS 代表队列中可以使用的最小 CPU 核数，NJOBS 代表队列中所有的核数，PEND 代表正在等待的核数，RUN 代表正在运行的核数。

3.5 利用 bqueues 查询可用的队列信息

```
[era@sce ~]$ bqueues
```

```
[era@sce ~]$ bqueues
GRID      HPC      QUEUE     WALLTIME  MAXCPUS  MINCPUS  NJOBS  PEND  RUN
-----
HuaiRou   era      c kevnYang 14400     120      8        0      0      0
HuaiRou   era      cpu        370       1100     20       3504   1388   2064
HuaiRou   era      cpuII      370       13470    24       4812   312   4500
HuaiRou   era      mic        360       160      20       128    128    0
```

GRID 代表网格节点，HPC 代表集群名称，QUEUE 代表该集群的队列，WALLTIME 代表作业在该队列内允许运行的最长时间（以分钟为单位），MAXCPUS 代表队列中可以使用的最大 CPU 核数，MINCPUS 代表队列中可以使用的最小 CPU 核数，NJOBS 代表队列中所有的核数，PEND 代表正在等待的核数，RUN 代表正在运行的核数。红色队列代表当前用户不可用队列。

3.6 提交一个作业到 era 集群上。

```
[era@sce ~]$ scephut2 lammps.in
```

```
[era@sce ~]$ bsub -n 20 -q cpu -W 30 Lammps -in lammps.in -log lammps.log
```

```
[era@sce ~]$ bsub -n 20 -q cpu -W 30 Lammps -in lammps.in -log lammps.log
gid is: 1460913353210977321, ujid is :480
Success!
```

先用 `sceput2` 上传 `lammps.in` 文件，然后用 `bsub` 提交作业。提交成功之后，会返回一个具体的 `gid` 号和 `ujid` 号，并显示 “Success!”。

3.7 设置一个 alias 别名来查看 480 号作业的信息。

```
[era@sce ~]$ alias list="bjobs 480"
[era@sce ~]$ list
```

```
[era@sce ~]$ alias list="bjobs 480"
[era@sce ~]$ list
-----
UJID   STAT   EXEC_HOST   QUEUE   NCORE JOB_NAME   SUBMIT   UPDATE
-----
480    DONE   era         cpu     20    *n lammps Apr 07 14:08 Apr 07 14:10
```

设置此别名之后，可以利用 `list` 来查询 480 号作业的信息。如果想删除此别名，可以利用 `unalias list` 来完成。

3.8 利用 `scesh2` 查看第 480 号作业的文件信息

```
[era@sce ~]$ scesh2 -l 480
```

```
[era@sce ~]$ scesh2 -l 480
total 16K
-rw-r--r-- 1 user083 SCGRID 1.3K Apr 7 14:09 480.err
-rw-r--r-- 1 user083 SCGRID 1.2K Apr 7 14:09 480.out
-rw-r--r-- 1 user083 SCGRID 410 Apr 7 14:08 Info.out
-rw-r--r-- 1 user083 SCGRID 0 Apr 7 14:09 lammps.log
-rw-r--r-- 1 user083 SCGRID 8 Apr 7 14:08 localuser
drwxr-xr-x 3 user083 SCGRID 0 Mar 24 18:12 sce
```

查看 480 号作业 HPC 端的文件，`480.out` 文件包含运行作业之后的输出结果，`480.err` 文件包含运行作业之后的出错信息。如果提交的作业有上传其它文件，其它文件也会保存在这层目录内。

```
[era@sce ~]$ scesh2 -c 480
```

```
[era@sce ~]$ scesh2 -c 480
total 40K
-rw-r--r-- 1 sce sce 1.1K Apr 7 14:09 1460913353210977321.xml
-rw-r--r-- 1 sce sce 1.3K Apr 7 14:09 480.err
-rw-r--r-- 1 sce sce 1.2K Apr 7 14:09 480.out
-rw-r--r-- 1 sce sce 259 Apr 7 14:09 Info.out
-rw-r--r-- 1 sce sce 43 Apr 7 14:11 job.sub
```

查看 480 号作业缓存目录的文件，`*.xml` 是提交作业之后自动生成的系统文件。`job.sub` 包含了作业提交的信息，如果作业状态为 “SUB_ERR”，可以使用 `scecat2 -c ujid job.sub` 查看出错信息。

3.9 利用 `scecat2` 查看文件信息

```
[era@sce ~]$ scecat -h -l 480 480.out
```

```
[era@sce ~]$ sccat2 -h -l 480 480.out
Sender: jhscheduler System <jhadmin@c1612>
Subject: Job 1289042: </home/SCGRID/user083/.JSDL/1460913353210977321/1460913353210977321.job> Done

Job </home/SCGRID/user083/.JSDL/1460913353210977321/1460913353210977321.job> was submitted from host <login5> by user <user083>.
Job was executed on host(s) <20*c1612>, in queue <cpu>, as user <user083>.
</home/SCGRID/user083> was used as the home directory.
</home/SCGRID/user083> was used as the working directory.
Started at Thu Apr 7 14:09:26 2016
Results reported at Thu Apr 7 14:09:32 2016
```

查看 480 号作业在 HPC 端的 480.out 文件，-h 表示从开头显示，默认显示 10 行内容。

3.10 编译一个 cparam.c 的作业(路径为~/testgcc)到 era, 其中 gcc 为 sceapp gnu_compiler gcc 的别名。

```
[era@sce ~]$ cd ~/testgcc
[era@sce testgcc]$ ls
```

```
[era@sce testgcc]$ ls
cparam.c
```

```
[era@sce testgcc]$ cat cparam.c
```

```
[era@sce testgcc]$ cat cparam.c
#include "stdio.h"
main(int argc, char* argv[])
{
    if(argc>0){
        printf("%s:%s\n",argv[0],argv[1]);
    }else{
        printf("no paragram");
    }
}
```

```
[era@sce testgcc]$ sccput2 cparam.c
```

```
[era@sce testgcc]$ sccput2 cparam.c
begin put files to remote hpc...
Uploading cparam.c to cparam.c
cparam.c 100% 185 0.2KB/s 00:00
```

```
[era@sce testgcc]$ gcc -o cparam cparam.c
```

```
[era@sce testgcc]$ scels
```

```
[era@sce testgcc]$ gcc -o cparam cparam.c
[era@sce testgcc]$ scels
Connecting, please wait.....(Press Ctrl+C to abort)
cparam
cparam.c
```

3.11 使用 generic 应用提交一个使用 3.10 编译结果的作业, 并利用 sccget2 获取结果

```
[era@sce testgcc]$ bsub -q cpu generic ./cparam helloworld
```

```
[era@sce testgcc]$ bsub -q cpu -n 20 generic ./cparam helloworld
gid is: 1460974277857875062, ujid is :481
Success!
```

```
[era@sce testgcc]$ bjobs
```

```
[era@sce testgcc]$ bjobs
UJID   STAT   EXEC_HOST   QUEUE   NCORE JOB_NAME   SUBMIT   UPDATE
-----
481    DONE   era         cpu     20    ./cparam*  Apr 07 15:07  Apr 07 15:08
480    DONE   era         cpu     20    *n lammgs  Apr 07 14:08  Apr 07 14:10
479    DONE   era         cpuII   96    *24181608  Mar 24 18:16  Mar 24 18:18
478    DONE   era         cpuII   24    *24181203  Mar 24 18:12  Mar 24 18:13
477    SUB_ERR* casnw      normal  12    47.0.252*  Mar 01 11:58  Mar 01 11:58
476    DONE   dicp       dgrid_qu* 8    vasp5.2    Oct 08 2014  Oct 08 2014
475    FAILED dicp       dgrid_qu* 8    vasp5.2    Oct 08 2014  Oct 08 2014
474    DONE   deepcomp7000 scgrid   8    *-p -ec 0  Sep 12 2014  Sep 12 2014
473    DONE   imr        intel    4    methano-*  Sep 11 2014  Sep 11 2014
472    DONE   nscctj     cngrid   8    NAMD2.9    Apr 11 2014  Apr 11 2014
471    DONE   nscctj     cngrid   8    NAMD2.9    Apr 11 2014  Apr 11 2014
470    DONE   nscctj     cngrid   8    NAMD2.9    Apr 11 2014  Apr 11 2014
469    DONE   nscctj     cngrid   8    NAMD2.8    Apr 11 2014  Apr 11 2014
468    DONE   nscctj     cngrid   8    Lammgs     Apr 11 2014  Apr 11 2014
467    DONE   nscctj     cngrid   8    Gromacs.*  Apr 11 2014  Apr 11 2014
466    DONE   nscctj     cngrid   8    Gromacs.*  Apr 11 2014  Apr 11 2014
465    DONE   nscctj     cngrid   8    DOCK       Apr 11 2014  Apr 11 2014
464    DONE   nscctj     cngrid   8    CPMD       Apr 11 2014  Apr 11 2014
463    DONE   nscctj     cngrid   8    Abinit     Apr 11 2014  Apr 11 2014
462    DONE   nscctj     cngrid   8    Abinit     Apr 11 2014  Apr 11 2014
Press 'q' to exit, press any key to continue...
```

```
[era@sce testgcc]$ scels
```

```
[era@sce testgcc]$ scels
Connecting, please wait.....(Press Ctrl+C to abort)
481.err
481.out
cparam
cparam.c
Info.out
localuser
```

```
[era@sce testgcc]$ scejget2 481.out
```

```
[era@sce testgcc]$ scejget2 481.out
begin get files from remote hpc...
Fetching 481.out to 481.out
481.out 100% 1122 1.1KB/s 00:00
```

```
[era@sce ~]$ ls
```

```
[era@sce testgcc]$ ls
481.out cparam.c
```

```
[era@sce ~]$ cat 481.out
```

```
[era@sce testgcc]$ cat 481.out
Sender: jhscheduler System <jhadmin@cl1425>
Subject: Job 1289095: </home/SCGRID/user083/.JSDL/1460974277857875062/1460974277857875062.job> Done

Job </home/SCGRID/user083/.JSDL/1460974277857875062/1460974277857875062.job> was submitted from host <login5> by user <user083>.
Job was executed on host(s) <20*c1425>, in queue <cpu>, as user <user083>.
</home/SCGRID/user083> was used as the home directory.
</home/SCGRID/user083/testgcc> was used as the working directory.
Started at Thu Apr 7 15:07:56 2016
Results reported at Thu Apr 7 15:07:56 2016

Your job looked like:

-----
# LSBATCH: User input
/home/SCGRID/user083/.JSDL/1460974277857875062/1460974277857875062.job
-----

Successfully completed.

Resource usage summary:

    CPU time      :      0.01 sec.
    Max Memory    :         2 MB
    Max Swap      :        19 MB

    Max Processes :          1

The output (if any) follows:

./cparam:helloworld
Total cpucount is:20
Total runtime is:0d0h0m0s.

PS:

Read file <481.err> for stderr output of this job.
```

其中，./cparam:helloworld 为 481 号作业的输出。

SCGrid