Paladin 用户作业操作手册 V1.1

WWW.XTAOTECH.COM

概述

极道（XTao）的 Achelous 计算系统具备强大的 ABC 融合调度能力，可以帮助用户轻松构建复杂的数据流（Flow）应用。在某些场景，用户可能不需要构建复杂数据流，需要在集群中运行一个简单的程序或者容器。Achelous 平台的 Paladin 服务提供强大的任务管理功能可以满足用户这方面的需求。Paladin 类似于传统的调度器（如 SGE、PBS、LSF 或者 Slurm），提供基本的调度和任务管理功能，包括：

1. 计算节点管理：展示集群中计算节点的资源和负载信息，包括 cpu 和 mem 的统计等
2. 任务调度：根据用户提交任务的资源需求，按照优先级、队列及其它限制，匹配合适的物理节点和硬件资源并启动运行。
3. 任务管理：任务提交、查看、删除等等
4. 队列管理：将集群中的计算节点划分为队列，设置不同的调度策略（配额、优先级等等），用户可以将不同类型的任务投递到不同的队列。

为了兼容一部分使用传统 HPC 调度平台（例如 SGE）的用户，极道的 Paladin 提供的命令行模拟 SGE 命令的行为。这些命令行（qsub，qstat，qhost，qdel，qconf，qalter）最大程度兼容原生 SGE 命令，希望帮助传统 HPC 用户或 HPC 软件透明切换到 Achelous 平台。

Paladin 是完全不同于传统调度器的实现，在兼容传统调度方式的基础上，具备强大的容器调度功能：

1. 调度 Docker 容器：在标准命令行的基础上，用户只需要增加一个参数即可运行 Docker 容器。
2. 调度 Singularity 容器：在标准命令行的基础上，用户只需要增加一个参数即可运行 Singularity 容器。

本文档将介绍如何使用 Paladin 运行任务。用户可以访问 Achelous 社区链接（www.achelous.org）获取在线帮助。

注意：极道的 Paladin 是基于最新技术实现的调度器。为了方便用户从传统调度器（如 SGE）无缝迁移，Paladin 的命令行精确模拟了 SGE 的选项和输出，但 Paladin 并不是 SGE。如果用户误认为 Paladin 就是 SGE，将会错失其丰富强大的功能。因此造成的损失，极道概不负责。

用户管理及环境初始化

添加用户

极道的 Achelous 计算平台统一采用内置的 LDAP 用户管理方式：

- 系统管理员使用 xd-compute ldap 添加一个新用户，例如“jason”。
- “jason”用户可以通过 ssh 在登录节点登陆，并在所有计算节点有效。
- “jason”用户可以通过 passwd 命令修改自己的密码，修改后对所有计算节点有效。

注意：用户在某个计算节点通过 useradd 命令添加的用户只对该节点有效，对其它节点无效。不能用来投递任务。

一键初始化环境

极道提供了专门的命令行“poroscli env set”可以帮助用户一键完成所有环境的初始化，包括 Bioflow、Paladin 和 Poros 等，用户无需了解细节，命令行自动检测服务地址，完成初始化配置。命令行使用如下例所示：

```
[root@Cc2Apc ~]# poroscli env set
success to set biocli server env: achelous.servicemgr.apc:1204
success to set biocli endpoints env: http://Cc2Apc:2380,http://Cc1Apc:2380,http://Cc3Apc:2380
success to set hermit server env: paladin-wtest1-backend.servicemgr.apc:1193
success to set hermit endpoints env: http://Cc2Apc:2380,http://Cc1Apc:2380,http://Cc3Apc:2380
```

设置环境变量

除了使用一件配置功能，用户可以手动配置环境变量。例如当系统存在多个 Paladin 服务，用户可以手动配置指向特定的服务。Paladin 常用的环境变量主要是 Paladin 的服务器的地址。设置环境变量以后，后续每一个作业请求都会发送给指定的服务器。

命令：sge env set

参数：

- 环境变量名字：例如 server
- 环境变量值：例如 paladin-backend-sge.servicemgr.apc:1141

下面是一个命令行的示例：

```
[root@Cc2Apc ~]# sge env set server paladin-backend-v2-test.servicemgr.apc:1274
sge client set key: server  value: paladin-backend-v2-test.servicemgr.apc:1274
[root@Cc2Apc ~]# sge env set etcd http://Cc1Apc:2380,http://Cc2Apc:2380,http://Cc3Apc:2380
sge client set key: etcd  value: http://Cc1Apc:2380,http://Cc2Apc:2380,http://Cc3Apc:2380
[root@Cc2Apc ~]#
```

获取环境变量

获取当前环境变量的值。
命令：sge env get
示例：

```
[root@Cc2Apc ~]# sge env get
========== XTAO SGE ENV ==========
  * backend server: paladin-backend-v2-test.servicemgr.apc:1274
  * etcd endpoints: http://Cc1Apc:2380,http://Cc2Apc:2380,http://Cc3Apc:2380
  * default cpu:    -
  * default mem:    -
  * allow max job:  35536
[root@Cc2Apc ~]#
```

查看版本信息

查看当前客户端和服务端的版本信息

命令： sge version

示例：

```
[root@Cc2Apc v2]# sge version
Client: SGE Client
  Version:
  Go version: go1.17.12
  Git commit: d93b34061c9905146ca067749d7154567bbef42a
  Built:      2022-10-11_17-12-51
  OS:         linux
  Arch:       amd64

Server: Paladin Server
  Version:
  Go version: go1.17.12
  Git commit: d93b34061c9905146ca067749d7154567bbef42a
  Built:      2022-10-11_17-11-38
  OS:         linux
  Arch:       amd64
[root@Cc2Apc v2]#
```

用户 quota

root 用户可以为所有用户配置 quota 信息，限制用户投递的任务数量和使用资源的上限
命令：

```
[root@Cc2Apc v2]# sge quota set --help
Usage: sge quota set [<flags>]

Set quota of user

Flags:
  -h, --help               Show context-sensitive help (also try --help-long and --help-man).
  -u, --user=USER          set quota for this specify user
  -n, --task_num=TASK_NUM  quota number of task, -1 is no limit(use --task_num=-1), 0 is forbidden. example: -n 1000
  -c, --cpus=CPUS          quota of cpu, -1 is no limit(use --cpus=-1), 0 is forbidden. example: -c 10
  -m, --memory=MEMORY      quota of memory, unit support M,G,T, -1 is no limit(use --memory=-1M), 0 is forbidden. example: -m 2G
      --gpus=GPUS          quota of gpus, -1 is no limit, 0 is forbidden. example: --gpus=5
      --gpumem=GPUMEM      quota of gpu mem, unit support: M,G,T, -1 is no limit(use --gpumem=-1M), 0 is forbidden. example: --gpumem=1G

[root@Cc2Apc v2]# sge quota get --help
Usage: sge quota get [<flags>]

Get quota info of user

Flags:
  -h, --help       Show context-sensitive help (also try --help-long and --help-man).
  -u, --user=USER  get quota info of specify user

[root@Cc2Apc v2]#
```

示例：为某个用户配置可投递任务上限为 100，cpu 资源上限为 20，内存资源上限为 50G

```
[root@Cc2Apc v2]# sge quota get -u wenjie
no quota information for the specified user: wenjie
[root@Cc2Apc v2]#
[root@Cc2Apc v2]# sge quota set -u wenjie -n 100 -c 20 -m 50G
succeed to set quota of wenjie by root
[root@Cc2Apc v2]#
[root@Cc2Apc v2]# sge quota get -u wenjie
User          QuotaTask     CurrentTask    QuotaCPUs      CurrentCPUs    QuotaMem       CurrentMem     QuotaGPUs      CurrentGPUs    QuotaGPUMem    CurrentGPUMem  LastSetTime
wenjie        100           0              20.0           0.0            50.0 G         0.0 M          -              0.0            -              0.0 M          2022-10-11 17:59:09
[root@Cc2Apc v2]#
```

用户软件安装

Paladin 既支持主机运行方式，也支持容器运行方式，因此有两种不同的软件安装方式。

主机运行模式

主机运行模式指的是用户将程序安装到计算节点的操作系统上，通过 qsub 投递任务运行程序的命令行。这种模式必须确保所有的计算节点可以运行该程序，并能够访问程序依赖的数据。有三种部署方式：

1. 所有节点安装同样的用户程序，数据放在共享存储上。
2. 程序安装到共享存储上的某个路径下，所有计算节点挂载共享存储，所有用户都有权限访问该路径并运行该程序。
3. 所有计算节点都挂载共享存储，每个用户将 home 目录放在共享存储上。所有用户在自己的 home 目录下编译和安装软件，通过 qsub 运行自己 home 目录下的程序。

容器运行模式

Paladin 支持用户以容器方式运行程序。Paladin 支持 Docker 和 Singularity 两种容器，运行方式也有差别：

- 如果用户使用 Docker 容器，需要将程序及其依赖打包到 Docker 镜像，通过 Poros 系统或者 imgcli 命令 push 到 Achelous 平台的私有镜像仓库。Paladin 在运行任务时自动在运行节点 pull 需要的镜像到本地运行，运行结束后自动回收镜像。
- 如果用户使用 Singularity 容器，需要将程序打包成 Singularity 的镜像，作为一个 SIF 文件存放在共享存储或者自己的 home 目录。Paladin 支持用户直接运行共享存储上的一个 SIF 镜像文件。用户也可以直接运行 Docker 仓库中的镜像，Paladin 在启动容器时会自动从私有镜像仓库 pull 镜像，转换成 Singularity 的 SIF 镜像格式运行，结束后自动回收镜像文件。

提交作业（qsub）

用户可以使用 qsub 命令提交作业，作业提交到 Paladin 服务后返回，并返回给用户作业 ID。用户使用该 ID 进行后续的作业管理。Paladin 既支持用户投递命令行作业，也支持用户投递容器（Docker 和 Singularity）作业。用户提交的作业（包括程序和容器）将以该用户的权限运行和读写数据，保证安全。root 用户不允许提交作业。

用户只需根据 qsub 要求，定义好资源，即可执行任意脚本。用户可同时投递任意多个任务，也无需关心任务在哪个节点执行。Paladin 根据优先级和队列策略调度作业，如果资源不足，作业按照调度策略排队等待可用资源。

命令行参数

用户可通过下面的方式运行 qsub 命令：

```
qsub [global options] command [command options] [arguments...]
```

基本的参数选项说明如下：

选项	例子	说明
--name value, -N value	-N yawei_job	指定作业名称
--workdir, --cwd	--cwd	使用当前目录为作业的工作目录。默认为：false。
--working_directory value, --wd value	--wd /mnt/vol1/job1	使用指定的目录为作业的工作目录。用户应该将目录指向共享存储。如果不指定“wd”和“cwd”，则使用当前用户的 home 目录为作业的工作目录。系统管理员应确保用户的 home 目录在所有计算节点已经挂载到共享存储上。

--err_path_list value, -e value	-e /mnt/vol1/test.err	指定作业的标准错误输出。 如果不指定，默认的标准错误输出指向提交用户的 home 目录下的一个自动生成的文件，qstat 命令可以展示文件路径
--out_path_list value, -o value	-o /mnt/vol1/test.out	指定作业的标准输出。 如果不指定，默认的标准输出指向提交用户的 home 目录下的一个自动生成的文件，qstat 命令可以展示文件路径
--variable_list value, -v value	-v USER=root,VAR=xx	为作业添加环境变量
--revocable_resource		指定作业是否优先使用浮动资源。默认为 false，即不可使用浮动资源。使用浮动资源的任务当系统负载过重将会被杀死。（默认：false） 注意：与 SGE 不同，在 Achelous 平台上内存使用超过限制的作业都会被杀死。
--resource_list value, -l value	-l cpu=1:mem=128M:gpus=1(gpui=1/gpumem=512M):h=host1	指定作业的物理资源需求： 1）cpu: cpu 数，可以为浮点数例如 0.2, 1.3 等等。其中 1.0 表示一个 CPU 的 100%，其它类推。 2）mem：物理内存需求 3）gpus：GPU 卡或者虚拟 GPU 卡（A100 类型的 MIG 的 compute instance）的数量，必须为整数 4) gpui：GPU instance 的数量，必须为整数 5）gpumem: GPU显存的数量。gpus/gpui和gpumem不能同时使用，且gpumem只能在容器模式下使用。 6）h：指明作业运行的计算节点。如果该计算节点不能满足资源需求，作业将排队等待。
--wc_queue_list value, -q value	-q queue_1	指定作业绑定的队列。 注意：提交作业的用户需要拥有对该队列的操作权限。
--task_id_range value, -t value	-t n[-m[:s]]	以列表方式创建多任务作业，该组任务共用同一个作业 id,每个任务有各自的任务 id。参数必须整数范围并指定步长(步长为默认值 1 时可以不指定)。 注意：如 -t 1-3 是指创建一个（默认步长）多任务数组作业，该作业的子任务 id 分别为 1，2，3。 如-t 2-7:2 是指创建一个步长为 2，包含三个任务数组的作业，该作业的子任务 id 分别为 2，4，6。

--image value, -i value	-i my_docker_image	该选项指明容器作业所用的 Docker 镜像名字。如果用户指定该选项，提交的作业将以 Docker 容器的方式运行，作业脚本将被自动映射到容器内，执行的将是容器内的程序或者命令。其它与命令行方式一致。
--singularity_image value, --si value	1）-si /mnt/vol1/user1/my_singularity_image.sif 2）-si my_docker_image	该选项指明容器作业所用的 Singularity 镜像名字。如果用户指定该选项，提交的作业将以 Singularity 容器的方式运行，作业脚本将被自动映射到容器内，执行的将是容器内的程序或者命令。其它与命令行方式一致。 注意：用户可以直接指定一个 sif 格式的文件路径（该路径需要在共享存储上或者共享的 home 目录下），也可以指定一个 Docker 镜像名称。
--volume value, --vol value	--vol /host/my_app:/container/my_app --vol /host/my_dir	在 Docker 或者 Singularity 容器运行模式下，用户通过该选项指定映射到容器的存储卷。其格式为“--vol [host_path]:[container_path]”，前面为主机上的目录，后面为映射到容器内的目录。如果没有“：”则默认容器内目录与主机目录相同。 注意：Paladin 默认会将作业用户的 home 目录以及执行脚本所在目录映射到容器内。
--network value, --net value	--net HOST	适用于容器作业场景。指定作业的容器网络的模式，只有对投递的容器任务生效。支持下面几种选项： 1）BRIDGE：使用容器的网桥。如果需要通过网络访问容器，需要进行端口映射。 2）HOST：使用主机网络，使用主机的 IP 地址和端口。 3）NONE：默认值，由 Paladin 选择默认值。Paladin 通常为 Docker 容器选择主机模式，为 Singularity 容器选择网桥模式。
--port_num value, -P value	--port_num 3	适用于容器作业场景。当用户作业指定的容器网络模式为 HOST 时，该选项生效，为容器在运行的机器上分配指定数量的端口。 用户作业可以在容器内通过环境变量获取分配给该作业的端口: 1）CONTAINER_PORTS_AVAILABLE_HOST：在 HOST 网络模式下存储分配给容器的端口 2）CONTAINER_PORTS_AVAILABLE_BRIDGE：在BRIDGE网络模式下存储分配给容器的端口

--port value, -p value	1）--port [host_port]:[container_port] 2）--port [host_port] 3）--port [container_port]	适用于容器作业场景。 1）当用户作业指定的网络模式为BRIDGE，指定[host_port:container_port]时，如果主机端口host_port已被分配，则返回错误；否则，启动容器，并在容器运行节点，将主机上的host_port端口映射到容器内的container_port端口。 2）当用户选择网络模式为 BRIDGE，指定[container_port]时，Paladin 将为作业分配一个未被占用的主机端口 host_port，启动容器，并且将主机上的 host_port 端口映射到容器内的 container_port 端口。 3）当用户选择网络模式为 HOST，指定[host_port]时，Paladin 将为作业寻找 host_port 尚未分配的主机，如果分配不到则失败；如果分配成功，则在该计算节点以 HOST 网络模式启动容器。
--priority value, -p value	-p value	定义作业的优先级，范围[0，10]，数值越大，级别越高。 注意：如果用户同时指定了作业优先级及队列，当指定的作业优先级和队列已绑定的优先级数值不一致时，Paladin 将使用较小的优先级作为该作业的优先级。
--sgehelp,--hl	--hl	输出 sge 帮助（默认为 false）
--date_time value, -a value		设置作业的开始时间 Paladin 暂时不支持该选项
--context_list value, --ac value		设置全局变量 Paladin 暂时不支持该选项
--fname value, --Ap value		为文件增加一个新的并行执行变量 Paladin 暂时不支持该选项
--binary value, -b value		判断作业指定是二进制文件或 scripts。y：二进制文件 n：scripts Paladin 暂时不支持该选项
--ckpt_selector value, -c value		定义作业的检查点类型 Paladin 暂时不支持该选项
--directive_prefix value, -C value		定义作业脚本的指令前缀 Paladin 暂时不支持该选项
--simple_context_list value, --dc value		删除全局变量 Paladin 暂时不支持该选项
--listname_list value, --dul value		删除用户设置列表 Paladin 暂时不支持该选项

--job_identifier_list value, --hold_jid value		定义或重定义当前提交的作业对哪些作业有依赖关系。 Paladin 暂时不支持该选项
--hard, --hr		定义作业被调度的硬性要求（默认:false） Paladin 暂时不支持该选项
--start_now value, --now value		作业立即执行或者根本不执行 Paladin 暂时不支持该选项
-context_lists value, --sc value		设置作业的环境（代替旧环境） Paladin 暂时不支持该选项
--soft, --sof		定义作业被调度的软性要求 Paladin 暂时不支持该选项
--rqs_list value,--srqs value		显示设置的配额资源信息 Paladin 暂时不支持该选项
--sync,--sy		等待任务结束才返回（默认：false） Paladin 暂时不支持该选项
--verify,--vf	--vf	仅验证，不提交（默认：false）
--env,-V	-V	为作业导入当前客户端所有的环境变量（默认：false）
--merge_stdout_and_stderr value, -j value	-j y	合并作业的标准输出和标准错误输出，标记‘yes’或‘y’会被合并输出，标记为‘no’或‘n’不会被合。（默认：no）
--wc_pe_name value,--pe value	-pe smp value	为并行的作业设置 cpu 个数。 注意：如-pe smp 3，即为并行作业设置的 cpu 个数为 3。如果 smp 设置的 cpu 个数和-l 指定资源的 cpu 个数不一致，以-l 指定资源的 cpu 个数为准。
--auto_adjust_resource value,--aar value	--aar auto	自动调节作业资源，支持‘up’，‘down’，‘auto’
--starve_timeout value,--st value	--st 6h	如果作业在饥饿超时时间内没有被成功调度，该作业将会被终止。饥饿超时时间支持整数类型，单位：h,m,s 。 注意：如 35s 代表 35 秒，19m 代表 19 分钟，10h 代表 10 小时，目前不支持同时设置多个时间单位。
--shell_interpreter value, --shell value		指定 shell 脚本 解释器。例如'--shell /usr/bin/bash'
--help,-h	-h	显示帮助（默认：false）
--disable_oom_kill		在任务运行发生 OOM 时候，不是直接杀死任务进

		程，而是保持 hang 状态，只对 docker 任务模式有效。（默认：false）
--running_timeout value		如果作业在运行超时时间内没有结束退出，该作业将会被终止。运行超时时间支持整数类型，单位：h,m,s 。 注意：如 35s 代表 35 秒，19m 代表 19 分钟，10h 代表 10 小时，目前不支持同时设置多个时间单位。
--callback_addr value		指定状态变更和资源变更通知的服务地址

投递命令行作业

用户可以通过运行 qsub 命令运行前文所述方式安装的软件程序，如下面的例子所示。命令行将返回一个作业 id，如例子中的“2315”。用户通过该 id 可以查询和删除作业。

```
[root@Cc2Apc ~]# qsub -N test -p 3 -q zwb -l cpu=0.5:mem=128M --shell /bin/bash --aar auto /home/wenjie/scripts/test-bash.sh
Your job 2315 ("test") has been submitted
[root@Cc2Apc ~]#
[root@Cc2Apc ~]#
[root@Cc2Apc ~]#
```

注意：用户需要确保工作目录、程序脚本、程序文件以及标准输入输出均指向共享存储的目录或者文件路径。程序将以提交用户的身份在计算节点运行，只允许访问该用户有权限访问的数据。

投递 Docker 容器作业

命令行 qsub 支持用户投递一个 Docker 容器作业。在上述运行命令行作业的基础上，用户只需要添加一个参数“-i”，指明 Docker 镜像名称。不熟悉 Docker 容器技术的用户可以访问极道的 Achelous 的社区链接（www.achelous.org）了解相关内容。

下面的例子展示了如何投递一个容器作业：

```
[jason@Cc1Apc ~]$
[jason@Cc1Apc ~]$
[jason@Cc1Apc ~]$ qsub -l cpu=1:mem=1024M:gpus=1 -i biotk -e /autofs/vol6/jason/test.err -o /autofs/vol6/jason/test.out /autofs/vol6/jason/sge/gpu_test.sh
Job (id: 160, name: nil) has been submitted
[jason@Cc1Apc ~]$
[jason@Cc1Apc ~]$
```

投递容器作业的资源设置、标准输入输出和名字等参数与投递命令行作业的场景完全一致。qsub通过“-i”选项设别这是一个Docker容器任务。用户需要注意：

- 如果作业需要访问数据，用户需要通过“--vol”参数将存储目录挂载到容器内部。Paladin 默认将用户的 home 目录和脚本文件所在的目录映射到容器内部，其它目录由用户自己负责挂载。
- 用户的脚本文件或者命令行都将指向容器内部版本，确保软件及其依赖安装到容器内的正确路径。
- Docker 容器也将以提交作业的用户的权限运行，只能访问用户权限允许的数据。
- Docker 容器默认使用主机的网络，用户可以通过“—net”参数指定网络类型。

投递 Singularity 容器作业

Paladin 不但支持 Docker 容器，也支持 Singularity 容器技术。不熟悉 Singularity 容器技术的用户可以访问极道的 Achelous 的社区链接（www.achelous.org）了解相关内容。qsub 命令通过参数"--si"识别并启动一个 Singularity 容器作业。

下面例子展示了如何投递一个 Singularity 容器作业：

```
[jason@Cc1Apc jason]$
[jason@Cc1Apc jason]$
[jason@Cc1Apc jason]$ qsub -l cpu=1:mem=1024M:gpus=1 -si /autofs/vol6/jason/jason-tf.sif -e /autofs/vol6/jason/test.err -o /autofs/vol6/jason/test.out /autofs/vol6/jason/sge/gpu_test.sh
Job (id: 161, name: nil) has been submitted
[jason@Cc1Apc jason]$
[jason@Cc1Apc jason]$
```

与 Docker 容器作业不同，Singulairty 容器作业具备下述特点：

- 参数"--si"既可指向一个 Docker 镜像，也可指向一个公共可访问的 Singularity 的镜像仓库上的 Singularity 镜像，也可指向存储上的一个 sif 文件。这是 Singularity 容器技术区别与 Docker 的显著特点。
- Singularity 容器默认使用“bridge”网络，用户可以通过“—net”参数指定网络类型。

监控作业（qstat）

qsub 投递作业到 Paladin 后即退出，并返回一个作业 id。用户可以通过 qstat 命令查看作业运行状态及相关信息。

监控所有运行的作业

用户通过命令“qstat”可显示当前 Paladin 上本用户的所有未完成的作业。下面是一个例子：

```
[root@Cc2Apc ~]#
[root@Cc2Apc ~]# qstat
Total: 5
List (max display limit is 200):
--------------------------------------------------------------------------------------------------------------------------------------------------
job-ID   prior  name                            user        state  submit at                  start at                   terminate at   queue        ja-task-ID
--------------------------------------------------------------------------------------------------------------------------------------------------
2319     4      root.paladin-task.kp1DPFRf      root        r      2022-10-11T16:38:11+08:00  2022-10-11T16:38:12+08:00  -              all@Cc4Apc   -
2318     4      root.paladin-task.sHmr7D6S      root        r      2022-10-11T16:38:10+08:00  2022-10-11T16:38:11+08:00  -              all@Cc6Apc   -
2317     0      say-hi                          root        qw     2022-10-11T16:37:04+08:00  -                          -              -            -
2316     0      play                            root        qw     2022-10-11T16:36:44+08:00  -                          -              -            -
2315     0      test                            root        qw     2022-10-11T16:35:34+08:00  -                          -              -            -
[root@Cc2Apc ~]#
```

命令输出信息说明如下表。

数据项	说明
job-ID	作业 id
prior	作业运行的优先级
name	作业名称
user	提交作业的用户
state	作业状态： 1) r：作业正在运行 2) qw：作业等待调度，可能是资源不满足或者配额限制 3) t：作业终止，作业可能被 kill 4) u：作业失去联系
submit at	作业提交时间
start at	作业开始运行时间
terminate at	作业终止时间
queue	作业队列，如果作业正在运行，会显示在队列的哪个计算节点运行。如果作业未运行，则显示-。
slots	始终为 1，目前 Paladin 不支持。
ja-task-ID	多任务数组作业的子任务 id

查询特定作业

用户运行“qstat -j jobid”命令查询指定作业的状态。下面是一个例子。

```
[root@Cc2Apc ~]# qstat -j 2319

==============================================================
job_number:                 2319
job_name:                   root.paladin-task.kp1DPFRf
job_type:                   PROCESS
state:                      running
state_reason:               -
exit_code:                  -
exit_code_msg:              -
exec_file:                  job_scripts/paladin-task.73954daf-a49a-459e-9c0d-becd9c86306a
submission_time:            2022-10-11T16:38:11+08:00
execution_time:             2022-10-11T16:38:12+08:00
deadline:                   -
unreachable_time:           -
owner:                      root
account:                    root
uid:                        0
gid:                        0
groups:                     [0(root) 512(Domain Admins)]
sge_o_home:                 /home/root
sge_o_log_name:             root
sge_o_path:                 /root/.cargo/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/usr/local/go/bin:/root/bin
sge_o_shell:                /bin/bash
sge_o_workdir:              /root
sge_o_host:                 Cc4Apc
stdout_path_list:           /root/test-bash.sh.o2319Cc2Apc
stderr_path_list:           /root/test-bash.sh.e2319Cc2Apc
mail_list:                  root@Cc2Apc
priority:                   4
env_list:                   HOME=/root,NQUEUES=1,ENVIRONMENT=BATCH,JOB_ID=2319,SGE_CWD_PATH=/root,SGE_O_HOME=/root,SGE_STDERR
_PATH=/root/test-bash.sh.o2319Cc2Apc
script_file:                /home/wenjie/scripts/test-bash.sh
binding:                    NONE
container_image:            NONE
container_network:          NONE
container_port_env:         NONE
port_binding:               NONE
volume_binding:             NONE
starve_timeout:             0h 30m 0s
running_timeout:            NONE
resource_request:           cpu=0.01, mem=64.00 M, gpus=0.00, gpui=0.00, gpumem=0.00  M
usage        2319:          cpu=0.01, mem=64.00 M, gpus=0.00, gpui=0.00, gpumem=0.00  M
binding      2319:          NONE
command_launch_type:        POSIX SHELL
command_content:            "/bin/bash -c /home/wenjie/scripts/test-bash.sh"
constraints:                []
schedule_domain:            [all]
labels:                     []
scheduling_info:            (Collecting of scheduler job information is turned off)
==============================================================
```

其中显示信息说明如下表:

数据项	说明
job_number	作业 id
job_name	作业名字
job_type	作业类型： 1）PROCESS：命令行作业 2）DOCKER：Docker 容器作业 3）SINGULARITY：Singularity 容器作业

state	作业运行的状态： 1) running：作业正在运行 2) staging：作业正在启动 3) starting：作业正在启动 4) finished：作业成功结束 5) failed：作业结束，返回值为失败 6) killed：作业被杀，可能是因为资源或者负载原因 7) lost：作业运行所在的节点失去联系 8) error：启动作业过程中调度器错误 9) unreachable：运行的作业失去联系 1) queued：作业处于调度队列，等待资源 2) terminating：作业正在终止 3) terminated：作业被意外终止 4) unknown：作业状态未知
state_reason	如果作业处于失败或者终止状态，它表示作业失败的原因
exec_code	作业进程退出码
exec_code_msg	作业进程退出码的描述信息
exec_file	Paladin 用这个数据项记录作业内部的长 ID，对用户没有特别的意义
submission_time	作业提交的时间
execution_time	作业开始被调度运行的时间
deadline	如果作业完成，记录完成的时间
unreachable_time	作业失去联系的时间
owner	提交作业的用户
uid	提交作业的用户 ID
group	提交作业的用户所在的组
gid	提交作业的用户所在的组 ID
sge_o_home	提交作业用户所在的 home 目录
sge_o_log_name	用户名，目前没有特别意义。
sge_o_path	提交作业所在节点用户的 PATH 环境变量，目前未使用
sge_o_shell	提交作业用户的 shell
sge_o_workdir	作业的工作目录
sge_o_host	作业运行的主机名
account	作业用户名，没有特别意义

stderr_path_list	作业标准错误输出文件的路径
mail_list	用户名相关，没有特别意义
stdout_path_list	作业的标准输出文件路径
priority	作业优先级
env_list	作业的环境变量
script_file	作业执行脚本的路径
binding	暂时不支持
container_image	容器作业的镜像名或者镜像文件路径
container_network	作业指定的网络类型，适用于容器作业
container_port_env	存储容器作业分配的端口的环境变量的值
port_binding	容器作业的端口映射
volume_binding	容器作业的卷映射
starve_timeout	作业提交后进入队列分配不到资源等待的最长时间，超过时间则作业自动失败
running_timeout	任务运行最大时间，超过该时间任务将被终止
resource_request	作业的资源申请数量
usage	作业的资源分配数量
command_launch_type	命令的启动类型，目前支持‘POSIX SHELL’,‘POSIX EXEC’
command_content	执行的具体命令
constraints	提交作业设置的约束条件，以 key:value 的形式限制作业允许调度的计算节点的集合
schedule_domain	作业运行的调度域，与提交作业绑定的队列对应
labels	作业的 label 列表
schedule_info	暂时不支持

删除作业（qdel）

用户可以通过命令行 qdel 删除投递到 Paladin 的作业。对于个命令，用户需要了解作业的标识号，即成功执行 qsub 命令后显示的编号。 如果忘记该编号，请使用 qstat 命令进行检索。

命令行参数：

--all，-a：删除当前用户所有的任务

示例：

```
[root@Cc2Apc ~]# qstat
Total: 3
List (max display limit is 200):
-------------------------------------------------------------------------------------------------------------------------------------------------
job-ID    prior  name                                                    user            state  submit at                start at
-------------------------------------------------------------------------------------------------------------------------------------------------
2317      0      say-hi                                                  root            qw     2022-10-11T16:37:04+08:00  -
2316      0      play                                                    root            qw     2022-10-11T16:36:44+08:00  -
2315      0      test                                                    root            qw     2022-10-11T16:35:34+08:00  -
[root@Cc2Apc ~]#
[root@Cc2Apc ~]# qdel 2317
accept to deleted job 2317
[root@Cc2Apc ~]#
[root@Cc2Apc ~]# qstat
Total: 2
List (max display limit is 200):
-------------------------------------------------------------------------------------------------------------------------------------------------
job-ID    prior  name                                                    user            state  submit at                start at
-------------------------------------------------------------------------------------------------------------------------------------------------
2316      0      play                                                    root            qw     2022-10-11T16:36:44+08:00  -
2315      0      test                                                    root            qw     2022-10-11T16:35:34+08:00  -
[root@Cc2Apc ~]#
```

管理主机（qhost）

用户可通过 qhost 命令显示 Achelous 计算平台的主机以及每个主机的资源列表。如下面示例：

```
[root@Cc2Apc ~]#
[root@Cc2Apc ~]# qhost
Cluster: apc
-------------------------------------------------------------------------------------------------------------------------------------------------
HOSTNAME                ARCH      ACTIVE  NCPU     CPUSUSE  NSOC NCOR NTHR LOAD  MEMTOT   MEMUSE   GPUSTOT  GPUSUSE  GPUISTOT  GPUISUSE  GPUMEMTOT  GPUMEMUSE  SWAPTO  SWAPUS
-------------------------------------------------------------------------------------------------------------------------------------------------
global                  -         -       -        -        -    -    -    -     -        -        -        -        -         -         -          -          -       -
Cc1Apc                  lx-amd64  true    53.00    45.30    4    4    4    0.01  178.0G   118.6G   -        -        -         -         -          -          0.0     0.0
Cc2Apc                  lx-amd64  true    53.00    42.20    4    4    4    0.01  178.0G   99.5G    -        -        -         -         -          -          0.0     0.0
Cc3Apc                  lx-amd64  true    53.00    31.70    4    4    4    0.01  178.0G   72.5G    -        -        -         -         -          -          0.0     0.0
Cc4Apc                  lx-amd64  true    55.00    0.00     4    4    4    0.01  186.4G   0.0M     -        -        -         -         -          -          0.0     0.0
Cc5Apc                  lx-amd64  true    55.00    14.00    4    4    4    0.01  150.0G   45.9G    1        0        -         -         -          -          0.0     0.0
Cc6Apc                  lx-amd64  true    52.00    47.10    4    4    4    0.01  185.0G   60.9G    1        1        -         -         -          -          0.0     0.0
Cc7Apc                  lx-amd64  true    50.00    4.90     4    4    4    0.01  150.0G   10.1G    -        -        -         -         -          -          0.0     0.0
Cc8Apc                  lx-amd64  true    54.00    0.00     4    4    4    0.01  166.2G   0.0M     -        -        -         -         -          -          0.0     0.0
[root@Cc2Apc ~]#
```

用户使用“qhost -q”命令可以显示主机上的队列信息：

```
[root@Cc2Apc ~]# qhost -q
Cluster: apc
---------------------------------------------------------------------------------------------------------------------------------------------------
HOSTNAME                ARCH      ACTIVE  NCPU     CPUSUSE  NSOC NCOR NTHR LOAD  MEMTOT   MEMUSE   GPUSTOT GPUSUSE GPUISTOT GPUISUSE GPUMEMTOT GPUMEMUSE SWAPTO  SWAPUS
---------------------------------------------------------------------------------------------------------------------------------------------------
global                  -         -       -        -        -    -    -    -     -        -        -       -       -        -         -          -       -       -
Cc1Apc                  lx-amd64  true    53.00    45.30    4    4    4    0.01  178.0G   118.6G   -       -       -        -         -          -       0.0     0.0
    all                           BIP     0/0/4
    anna-test                     BIP     0/0/4
    dom-test                      BIP     0/0/4
    domain-bug-test               BIP     0/0/4
    Leng                          BIP     0/0/4
    biotest                       BIP     0/0/4
    janus                         BIP     0/0/4
    pri-test                      BIP     0/0/4
    test003                       BIP     0/0/4
    xtao-test                     BIP     0/0/4
    a100                          BIP     0/0/4
    alpha                         BIP     0/0/4
    servicemgr                    BIP     0/0/4
    test001                       BIP     0/0/4
    xtao2                         BIP     0/0/4
    xtao3                         BIP     0/0/4
    leng-test1                    BIP     0/0/4
    sandbox                       BIP     0/0/4
    scanner                       BIP     0/0/4
    xtao1                         BIP     0/0/4
    zwb                           BIP     0/0/4
Cc2Apc                  lx-amd64  true    53.00    42.20    4    4    4    0.01  178.0G   99.5G    -       -       -        -         -          -       0.0     0.0
    dom-test                      BIP     0/0/4
    pri-test                      BIP     0/0/4
    xtao-test                     BIP     0/0/4
    biotest                       BIP     0/0/4
    sandbox                       BIP     0/0/4
    xtao2                         BIP     0/0/4
    Leng                          BIP     0/0/4
    test001                       BIP     0/0/4
    scanner                       BIP     0/0/4
    xtao1                         BIP     0/0/4
```

其中数据项的说明如下表：

数据项	说明
HOSTNAME	主机名
ARCH	目前不支持，无意义
ACTIVE	节点是否 active
NCPU	主机上的 CPU 总数
CPUSUSE	已经被调度分配给作业的 CPU 数
MEMTOT	主机上的内存资源总数
MEMUSE	已经分配出去的内存资源量
GPUSTOT	主机上的 GPU 总数
GPUSUSE	已经分配出去的 GPU 数

GPUISTOT	主机上的 GPUI 总数
GPUISUSE	已经分配出去的 GPUI 数
GPUMEMTOT	主机上的 GPU 显存资源总数
GPUMEMUSE	已经分配出去的 GPU 显存资源量
SWAPTO	目前不支持，无意义
SWAPUS	目前不支持，无意义
LOAD	目前不支持，无意义
NSOC	目前不支持，无意义
NCOR	目前不支持，无意义
NTHR	目前不支持，无意义

管理队列（qconf）

配置队列

在 Achelous 平台，用户可以将若干个计算节点组成调度域。调度域具有任务优先级、配额、资源预留和用户权限等调度策略和安全策略。为了兼容传统 HPC 用户的使用习惯，Paladin 使用调度域模拟提供类似 SGE 的队列功能。因此 Paladin 的队列与调度域一一对应。

系统管理员请参考《Achelous 集群管理和运维手册》配置调度域，相关的配置将自动同步为 Paladin 的队列配置。极道的 qconf 命令不支持修改队列配置。

显示队列信息

用户可通过“qconf ”命令显示 Achelous 计算平台的队列信息。如下图示例，展示了每个计算节点属于哪些队列。

```
[root@Cc2Apc ~]# qconf -sql

Cluster: apc
  * host: Cc6Apc
    - active: true
    - queues: nerdctl-test, yawei-cc3, all, alpha, container-test, Leng, biotest, janus, scale-test, xtao1, dbio, emergency-domain, jason-quota-test, xta

  * host: Cc3Apc
    - active: true
    - queues: test003, Leng, alpha, biotest, sandbox, xtao3, all, xtao2, scanner, servicemgr, xtao-test, xtao1, yawei-janus-3, janus

  * host: Cc4Apc
    - active: true
    - queues: Leng, hg-test1, scale-test, test001, all, biotest, docker-test, hg-test, servicemgr, test003, xtao1, xtao2, xtao3

  * host: Cc2Apc
    - active: true
    - queues: dom-test, pri-test, test001, Leng, alpha, biotest, sandbox, xtao1, xtao2, xtao3, all, janus, scanner, xtao-test

  * host: Cc1Apc
    - active: true
    - queues: pri-test, scanner, xtao-test, xtao3, biotest, janus, anna-test, dom-test, test003, xtao1, zwb, a100, all, sandbox, test001, domain-bug-test

  * host: Cc8Apc
    - active: true
    - queues: xtao1, xtao2, Leng, emergency-domain, gpufat, xtao3, yawei-cc78, alpha, scale-test, test008, all, biotest, container-test, docker-test

  * host: Cc5Apc
    - active: true
    - queues: a100, alpha, biotest, dbio, jason-quota-test, xtao1, Leng, anna-test, test001, xtao2, janus, scale-test, yawei-cc3, all, container-test, d

  * host: Cc7Apc
    - active: true
    - queues: emergency-domain, jason-quota-test, xtao2, all, xtao3, yawei-cc78, Leng, biotest, dbio, yawei-cc3, alpha, container-test, docker-test, sca

[root@Cc2Apc ~]#
```

修改任务信息（qalter）

用户可通过“qalter ”命令修改尚未调度运行的任务的一些信息。目前支持修改任务优先级，回调通知地址，饥饿超时时间。

基本的参数选项说明如下：

选项	例子	说明
--priority value, -p value	-p 5	指定要修改的目标优先级
--callback value		指定要修改的接收回调通知的服务地址
--starve_timeout		指定要修改的饥饿超时时间
--job_ids value, -j value	-j 1,5,7	指定要修改的任务短 id
--long_job_ids value,-l value	-l paladin-task.uuid	指定要修改的任务长 id

例如修改任务的优先级从 4 到 2：

```
[root@Cc2Apc v2]# qstat
Total: 1
List (max display limit is 200):
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------
job-ID    prior  name                                              user          state  submit at                 start at              terminate at          queue                 ja-task-ID
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------
13        4      root.paladin-task.1FL238T9                        root          qw     2022-10-11T17:43:53+08:00 -                     -                     -                     -
[root@Cc2Apc v2]# qalter -p 2 -j 13
The priority of all specified jobs has been successfully updated
[root@Cc2Apc v2]#
[root@Cc2Apc v2]# qstat
Total: 1
List (max display limit is 200):
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------
job-ID    prior  name                                              user          state  submit at                 start at              terminate at          queue                 ja-task-ID
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------
13        2      root.paladin-task.1FL238T9                        root          qw     2022-10-11T17:43:53+08:00 -                     -                     -                     -
[root@Cc2Apc v2]#
```

查看任务启动日志

可以通过 sge logs 命令获取任务启动时候的 launcher 日志信息，而任务运行时候的日志记录在任务的 stderr 和 stdout 文件中。

```
[zhao@Cc2Apc stress-100k-small-running-test]$ sge logs 50490
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
* Job Number: 50490
* Launcher Stderr *
Marked '/' as rslave
I1018 11:08:35.264753 36435 exec.cpp:167] Version: 1.9.0
I1018 11:08:35.267946 36455 exec.cpp:245] Executor registered on agent 6efb5da9-19e6-49d6-977a-9e96b2abb69d-S0
I1018 11:08:35.269882 36461 executor.cpp:215] Received SUBSCRIBED event
I1018 11:08:35.270529 36461 executor.cpp:219] Subscribed executor on Cc6Apc
I1018 11:08:35.270658 36461 executor.cpp:215] Received LAUNCH event
I1018 11:08:35.271525 36461 executor.cpp:816] Starting task paladin-task.849e52a6-9b9c-40a3-9988-f56b859d29f5
I1018 11:08:35.279917 36461 executor.cpp:880] Forked command at 36493
I1018 11:08:35.364686 36473 executor.cpp:1250] Command exited with status 126 (pid: 36493)
I1018 11:08:35.369560 36477 exec.cpp:433] Set stop promise.
I1018 11:08:36.366928 36492 process.cpp:935] Stopped the socket accept loop


* Launcher Stdout *
Chdir to current working directory: /home/zhao/computer-test/xtsge-test/stress/stress-100k-small-running-test
```