用 SLURM 优化超级计算机内的资源管理

深入研究这篇对用于资源管理的 Simple Linux Utility 的简介

超级计算机之间的军备竞赛实在是非同寻常，因为它们各自不断进化的架构体现了越来越多的性能。有关超级计算机的一个有趣事实是它们都运行了某个版本的 Linux。为了在一个架构中提供最多的功能，SLURM 开源作业调度程序（由中国天河-IA 超级计算机以及即将推出的 IBM® Sequoia 超级计算机采用）优化了资源分配和监视。本文将了解 SLURM 及其平行集群内的工作负载的方式。

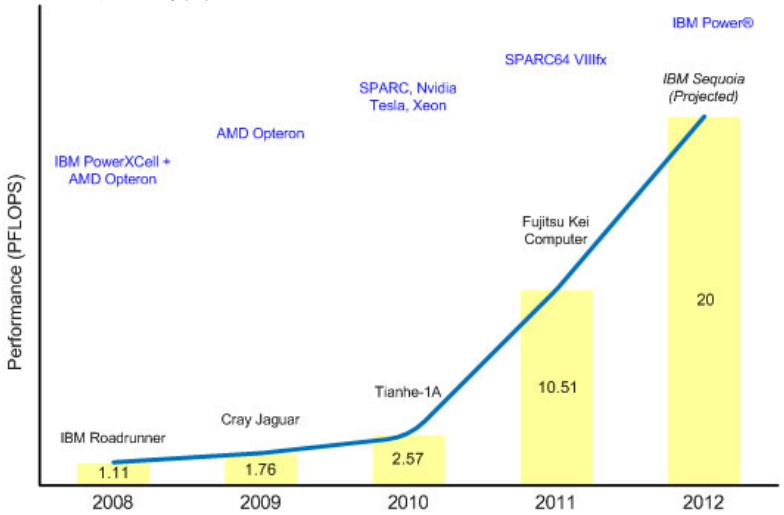
M. Tim Jones 是一名嵌入式固件架构师，也是 Artificial Intelligence:A Systems Approach、GNU/Linux Application Programming（现已发行第二版）、AI Application Programming（第二版）和 BSD Sockets Programming from a Multilanguage Perspective 等书籍的作者。他的工程背景非常广泛，从对地同步宇宙飞船的内核开发到嵌入式系统架构设计与网络协议开发。Tim 居住在科罗拉多州 Longmont，他是 Intel 的一名平台架构师，也是一名作家。

2012 年 6 月 21 日

超级计算机是军备竞赛的一个典型示例。随着现代超级计算机不断增强的性能扩展到新的问题领域，这些巨大的系统也在提供解决新问题的平台。超级计算机是国家和企业荣誉感的来源，因为公司和国家都致力于提高 LINPACK 的成绩。图 1 展示了过去五年间超级计算机军备竞赛的情况，同时 IBM Sequoia 超级计算机目前被预测为 2012 年内的领军者。如图所示，IBM Roadrunner 是第一台打破一直不变的千万亿次障碍的超级计算机（而 IBM Blue Gene®/L 则自 2004 年至 2008 年一直位居榜首）。

联系 Tim
Tim 是我们最受欢迎的撰稿人之一，并且是一位多产撰稿人。请浏览 developerWorks 上的 Tim 的所有文章。查看 Tim 的个人简介，并在 developerWorks 社区 与 Tim、其他撰稿人以及开发伙伴们交流。

图 **1.** 超级计算机性能：**2008-2012**



早期的超级计算机目的在于为核武器建模。如今，它们的应用更为广泛，可处理气候研究、分子建模、大型物理模拟甚至强力的密码破译等领域内的大量计算问题。

1964 年至今

通常认为，第一台超级计算机是 1964 年发布的（由 Seymour Cray 设计）Control Data Corporation (CDC) 6600。6600 使用硬件、Freon 冷却系统和能完成每秒浮点操作数为 3 百万的单个 CPU 填充

何为 LINPACK 基准？
为了对比互相竞争的超级计算机的性能，就创建了 LINPACK 性能基准。LINPACK 度量的是浮点运算的执行速度。具体而

了四个机柜。虽然并不缺少美感，但它的机柜却明显可见很多用于将外围单元处理器连接到单个 CPU 上以使其尽量繁忙的彩色电线。

言，LINPACK 是一组用来解决密集的线性方程系统的程序。

快速发展至今，目前的超级计算机的领先者是日本的 Kei 计算机（由 Fujitsu 构建）。此系统注重于蛮力计算功能，使用了超过 88,000 个 SPARC64 处理器，占用了 864 个机柜。Kei 超级计算机的一个显著特点是突破了 10 千万亿次的障碍。与 CDC 6600 类似，Kei 使用的是水冷加气冷。

什么是超级计算机？

超级计算机不是关于任何特定的架构，它只是处在计算性能尖端的一种设计。如今，这意味着如果以 LINPACK 基准度量，该系统能够在千万亿次（或百万之四次方的 FLOPS）的性能范围内运行。

无论超级计算机如何实现这些 FLOPS，任何超级计算机架构的一个低层目标都是在有工作可做时最佳地保持计算资源忙碌。与 CDC 6600 用来保持其单个 CPC 忙碌的外围处理器类似，现代的超级计算机需要同样的基本性能。让我们来看这样一个计算节点资源管理的实现，其名为 *Simple Linux® Utility for Resource Management* (SLURM)。

SLURM 简介

SLURM 是一种可用于大型计算节点集群的高度可伸缩和容错的集群管理器和作业调度系统。SLURM 维护着一个待处理工作的队列并管理此工作的整体资源利用。它还以一种排他或非排他的方式管理可用的计算节点（取决于资源的需求）。最后，SLURM 将作业分发给一组已分配的节点来执行工作并监视平行作业至其完成。

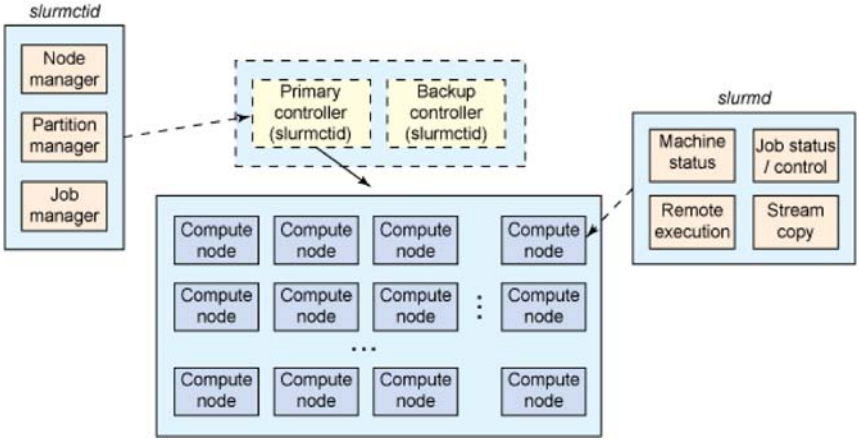
本质上，SLURM 是一个强健的集群管理器（更关注于对功能丰富性的需求方面），它高度可移植、可伸缩至大型节点集群、容错好，而且更重要的是它是开源的。SLURM 最早是一个开源的资源管理器，由几家公司（包括 Lawrence Livermore National Laboratory）协作开发。如今，SLURM 已经成为了很多最强大的超级计算机上使用的领先资源管理器。

SLURM 架构

SLURM 实现的是一种非常传统的集群管理架构（参见 图 2）。在顶部是一对冗余集群控制器（虽然冗余是可选项）。这些集群控制器可充当计算集群的管理器并实现一种管理守护程序，名为 `slurmctld`。`slurmctld` 守护程序提供了对计算资源的监视，但更重要的是，它将进入的作业（工作）映射到基本的计算资源。

每个计算节点实现一个守护程序，名为 `slurmd`。`slurmd` 守护程序管理在其上执行的节点，包括监视此节点上运行的任务、接受来自控制器的工作，以及将该工作映射到节点内部核心之上的任务。如果控制器发出请求，`slurmd` 守护程序也可以停止任务的执行。

图 **2. SLURM** 架构的高级别视图

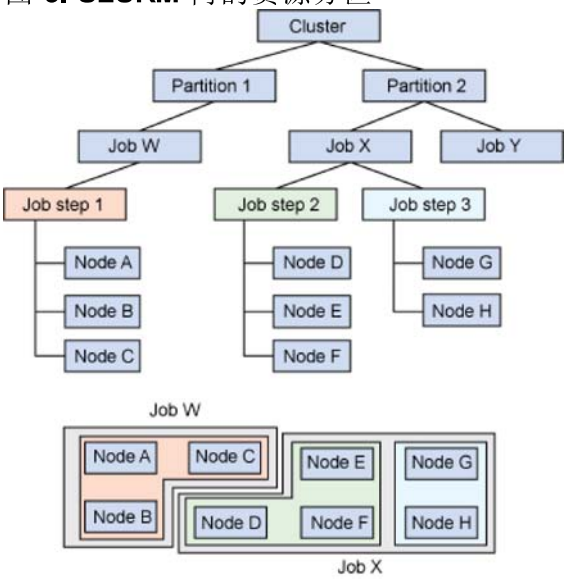


此架构内还存在其他的守护程序，比如，实现安全的身份验证。但是集群并不仅仅是节点的随机组合，因为这些节点可以是逻辑相关的，以适时实现平行计算。

一组节点也可以组成一个逻辑组，称为分区，分区通常会包含进入工作的队列。分区也可以配置各种约束条件，比如哪个用户可以使用它，分区支持的时限的作业大小。分区的更进一步优化，就是将分区内的一组节点在工作的一段时间内映射到一个用户，这就是一个作业。一个作业内，是一个或多个作业步骤，即在节点子集上执行的任务集。

图 3 展示了这个层次结构，进一步说明了资源的 SLURM 分区。请注意，这种分区包含了对资源的感知，相当于确保协作节点间的低延迟通信。

图 **3. SLURM** 内的资源分区



安装 SLURM

如何安装 SLURM 最终取决于您特定的 Linux 环境，但过程与使用一个包管理器一样简单。SLURM 是完全打包的，这就使其很容易进行安装和配置。对于我所钟爱的 distro、Ubuntu，我使用了 Advanced Packaging Tool (APT) 来安装 SLURM 包及其所有的依赖项：

```
$ sudo apt-get install slurm-llnl
```

此操作会使用少于 40MB 的空间且不只包含了 SLURM，还包含了依赖项、基础插件以及其他所需的包。

配置 SLURM

启动 SLURM 之前，必须根据特定的环境配置它。为了创建我的配置文件，我使用了在线的 SLURM 配置器，由它为我生成基于表单数据的配置文件。请注意此文件需要在末尾处进行修改以删除不再受支持的选项。清单 1 显示了我的结果配置文件（存储于 /etc/slurm-llnl/slurm.conf）。

清单 **1.** 面向单节点集群的 **SLURM** 配置文件

```
# slurm.conf file generated by configurator.html.
# Put this file on all nodes of your cluster.
# See the slurm.conf man page for more information.
#
ControlMachine=mtj-VirtualBox
#
AuthType=auth/none
CacheGroups=0
CryptoType=crypto/openssl
MpiDefault=none
ProctrackType=proctrack/pgid
ReturnToService=1
SlurmctldPidFile=/var/run/slurmctld.pid
SlurmctldPort=6817
SlurmdPidFile=/var/run/slurmd.pid
SlurmdPort=6818
SlurmdSpoolDir=/tmp/slurmd
SlurmUser=slurm
StateSaveLocation=/tmp
SwitchType=switch/none
TaskPlugin=task/none
#
# TIMERS
InactiveLimit=0
KillWait=30
MinJobAge=300
SlurmctldTimeout=120
```

```
SlurmdTimeout=300
Waittime=0
#
# SCHEDULING
FastSchedule=1
SchedulerType=sched/backfill
SchedulerPort=7321
SelectType=select/linear
#
# LOGGING AND ACCOUNTING
AccountingStorageType=accounting_storage/none
ClusterName=cluster
JobCompType=jobcomp/none
JobCredentialPrivateKey = /usr/local/etc/slurm.key
JobCredentialPublicCertificate = /usr/local/etc/slurm.cert
JobAcctGatherFrequency=30
JobAcctGatherType=jobacct_gather/none
SlurmctldDebug=3
SlurmdDebug=3
#
# COMPUTE NODES
NodeName=mtj-VirtualBox State=UNKNOWN
PartitionName=debug Nodes=mtj-VirtualBox default=YES MaxTime=INFINITE State=UP
```

请注意在一个真实的集群内，NodeName 应指的是一组节点，比如 snode[0-8191]，以表示此集群内的 8192 个独特的节点（名为 snode0 至 snode8191）。

最后一个步骤是为我的站点创建一组作业凭证密钥。我选择使用 openssl 作为我的凭证密钥（在 清单 1 内的配置文件中作为 JobCredential* 引用）。我只使用 openssl 来生成这些凭证，如 清单 2 所示。

清单 **2.** 为 **SLURM** 创建凭证

```
$ sudo openssl genrsa -out /usr/local/etc/slurm.key 1024
Generating RSA private key, 1024 bit long modulus
..............++++++
.............................................++++++
e is 65537 (0x10001)
$ sudo openssl rsa -in /usr/local/etc/slurm.key -pubout -out /usr/local/etc/slurm.cert
writing RSA key
```

这些步骤完成后，就万事齐备了，我就能告诉 SLURM 我的配置了。我现在就可以启动 SLURM 并与其交互。

启动 SLURM

要启动 SLURM，只需使用 /etc/init.d/slurm 内定义的管理脚本。此脚本接受 start、stop、restart 和 startclean（以忽略之前保存的所有状态）。用这种方法启动 SLURM 会导致 slurmctld 守护程序的启动（在这个简单配置中，还包括您节点上的 slurmd 守护程序）：

```
$ sudo /etc/init.d/slurm-llnl start
```

为了验证 SLURM 是否在运行，可以使用 sinfo 命令。sinfo 命令会返回有关这些 SLURM 节点和分区的信息（在本例中，集群由单个节点组成），如 清单 3 所示。

清单 **3.** 使用 **sinfo** 命令来查看集群

```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
debug*       up   infinite      1   idle mtj-VirtualBox
$
```

更多的 SLURM 命令

SLURM 内还有更多的命令可用来获得有关 SLURM 集群的更多信息。在 启动 SLURM 这个章节内，您会看到 sinfo 命令，可用来了解您的集群。您还可以用 scontrol 命令获得更多信息，这就使您可以查看集群各方面的详细信息（例如 清单 4 内的分区和节点）。

清单 **4.** 用 **scontrol** 获得有关集群的详细信息

```
$ scontrol show partition
PartitionName=debug
   AllocNodes=ALL AllowGroups=ALL Default=YES
   DefaultTime=NONE DisableRootJobs=NO Hidden=NO
   MaxNodes=UNLIMITED MaxTime=UNLIMITED MinNodes=1
   Nodes=mtj-VirtualBox
   Priority=1 RootOnly=NO Shared=NO PreemptMode=OFF
   State=UP TotalCPUs=1 TotalNodes=1

$ scontrol show node mtj-VirtualBox
NodeName=mtj-VirtualBox Arch=i686 CoresPerSocket=1
   CPUAlloc=0 CPUErr=0 CPUTot=1 Features=(null)
```

```
   Gres=(null)
   OS=Linux RealMemory=1 Sockets=1
   State=IDLE ThreadsPerCore=1 TmpDisk=0 Weight=1
   BootTime=2012-03-07T14:59:01 SlurmdStartTime=2012-04-17T11:10:43
   Reason=(null)
```

要测试这个简单的 SLURM 集群，可以使用 `srun` 命令。`srun` 命令可以为您的作业分配一个计算资源并启动一个任务。请注意您也可以分别实现这两个目的（通过 `salloc` 和 `sbatch`）。如 清单 5 内所示，您可以提交一个简单的 shell 命令作为您的作业来演示 `srun`，然后再提交一个 `sleep` 命令（带参数）来演示 `squeue` 命令的使用，从而展示集群内存在的作业。

清单 **5.** 向集群提交作业并检查队列状态

```
$ srun -l hostname
0: mtj-VirtualBox
$ srun -l sleep 5 &
[1] 24127
$ squeue
  JOBID PARTITION     NAME     USER  ST       TIME  NODES NODELIST(REASON)
     15     debug    sleep      mtj   R       0:03      1 mtj-VirtualBox
$
[1]+  Done                    srun -l sleep 5
$
```

注意在 清单 5 内，向集群提交的作业可以是一个简单的 Linux 命令、一个 shell 脚本文件或一个适当的可执行文件。

作为最后一个例子，让我们来看看如何停止一个作业。在本例中，您启动一个运行较长的作业并使用 `squeue` 来识别其 ID。然后，使用 `scancel` 命令与这个作业 ID 来终止该作业步骤（参见 清单 6）。

清单 **6.** 终止一个作业步骤

```
$ srun -l sleep 60 &
[1] 24262
$ squeue
  JOBID PARTITION     NAME     USER  ST       TIME  NODES NODELIST(REASON)
     16     debug    sleep      mtj   R       0:03      1 mtj-VirtualBox
$ scancel 16
srun: Force Terminated job 16
$ srun: Job step aborted: Waiting up to 2 seconds for job step to finish.
0: slurmd[mtj-VirtualBox]: error: *** STEP 16.0 CANCELLED AT 2012-04-17T12:08:08 ***
srun: error: mtj-VirtualBox: task 0: Terminated

[1]+  Exit 15                 srun -l sleep 60
$
```

最后，可以使用相同的 `slurm-llnl` 脚本来停止集群，如 清单 7 所示。

清单 **7.** 停止 **SLURM** 集群

```
$ sudo /etc/init.d/slurm-llnl stop
 * Stopping slurm central management daemon slurmctld                    [ OK ]
 * Stopping slurm compute node daemon slurmd                             [ OK ]
slurmd is stopped
$
```

与 Apache Hadoop 不同，SLURM 没有分布式文件系统的概念。因此，为了一个给定的计算，它需要更多的处理才能将数据分布到节点。SLURM 包含了这样一个命令，名为 `sbcast`，可用来将一个文件传递到一个 SLURM 作业分配的所有节点。跨 SLURM 集群的节点使用平行或分布式的文件系统是很有可能的（而且更为高效），这样一来，就不需要 `sbcast` 来分布要处理的数据了。

在这个简单 SLURM 的演示中，我们使用的只是可用命令的一个子集，以及这些命令可用选项的一个更小的子集（比如，参见 `srun` 命令的可用选项）。即便是用最少数量的可用命令，SLURM 都能实现一个有效和高效的集群管理器。

定制 SLURM

SLURM 并不是一个静态的资源管理器，而是一个可以结合新行为的高度动态的资源管理器。SLURM 实现了一个插件应用程序编程接口 (API)，允许运行时库在运行时动态加载。这个 API 已经用于开发各种新行为，包括互连结构、身份验证和调度。插件接口支持各种其他功能，比如作业统计、加密功能、消息传递接口 (MPI)、过程跟踪以及资源选择。所有这些都允许 SLURM 可以轻松支持不同的集群架构和实现。请查看 参考资料 部分的 SLURM 程序员指南，以了解详细信息。

SLURM 的前景

2011 年，SLURM 因各种新特性的加入而得到了更新，包括对 IBM Blue Gene/Q 超级计算机和 Cray XT 以及 XE 计算机的支持。此外，还添加了对 Linux 控制组 (cgroups) 的支持，这对 Linux 过程容器提供了更大的控制。

2012 年，Blue Gene/Q 支持将会全面实现，同时实现的还有改进的资源选择，该资源选择取决于作业需求和资源功能（比如，节点特性 AMD）。一种新的工具计划用来报告调度统计，而且在不久的将来，还将会有一种基于 Web 的管理工具。SLURM 的另一个未来计划是在云爆发的上下文中，这会涉及到在云提供者中分配资源，以及将溢出的工作从一个本地集群迁移到云中（也要运行 SLURM 守护程序）。这个模型非常有用，而且支持某些超级计算机工作负载弹性的理念。

最后，SLURM 开发人员也在考虑使用功率和热量数据，以便更有效地分配集群内的工作，比如，将消耗大功率（也会产生更多热量）的作业放在集群内散热较好的区域。

结束语

对 SLURM 的简单介绍阐明了这个开源资源管理器的简便性。虽然现代的超级计算机超出了大多数人的价格范围，SLURM 仍提供了可伸缩的集群管理器的基础，可将商用服务器转变成高性能集群。而且，SLURM 的架构还使得更易于对超级计算机（或商品集群）架构定制资源管理器。这可能也是其成为超级计算机领域内领先的集群管理器的原因。

参考资料

学习

SLURM: A Highly Scalable Resource Manager 具有一个由 Lawrence Livermore National Laboratory 管理的有用网站。该站点提供了对 SLURM 的概述、一系列出版物和演示文稿、文档以及一个有用的 FAQ。您还可以在 SchedMD 获得 SLURM 开发人员的专业支持。

A Brief History of Supercomputing 由 Microsoft 的 Gordon Bell 撰写，介绍了从 CDC 6600 到 Beowulf 集群的超级计算的有趣（虽然有些过时）历史。

The LINPACK Benchmark: Past, Present, and Future（Jack Dongarra，Piotr Luszczek 和 Anttoine Petitet）提供了对 LINPACK 历史和内部构件的有用介绍。还展示了一组常用的硬件（AMD Athlon，Intel® Pentium® 等）之上的 LINPACK。

SLURM Elastic Computing 实现了称为云爆发 的功能，允许集群根据置于其内的需要而进行增长和收缩。特别是，集群的增长使用了公共云资源来经济有效地扩展集群的功能。

The Top500 project 列出了全球最快的超级计算机的排名，以及当前和历史系统的详细信息。Top500 每年发布两次其更新的列表。不足为奇的是，Top500 所列的前十名均运行某种形式的 Linux。

IBM 超级计算机具有悠久卓越的历史。查看 this reference from Wikipedia，获取由 IBM 开发的某些超级计算机的列表和详细信息。

SLURM Programmer's Guide 简单易懂地介绍了如何构建插件，同时也能帮您了解 SLURM 源和文档布局。

SLURM Configuration Tool 提供了从简单形式的输入生成 slurm.conf 文件的简单方式。在输入了所需的信息后，这个网站就会发出一个 slurm.conf，可放入您的集群中。

developerWorks 中国网站开源技术专区 提供了有关开源工具以及使用开源技术的丰富信息。

developerWorks 中国网站 Web 开发专区：专门研究各种基于 Web 的解决方

案的文章。

观看 developerWorks 演示中心，那里提供了包括面向初学者的产品安装和设置演示，以及为经验丰富的开发人员提供的高级功能。

在 developerWorks Linux 专区 寻找为 Linux 开发人员（包括 Linux 新手入门）准备的更多参考资料，查阅我们 最受欢迎的文章和教程。

在 developerWorks 上查阅所有 Linux 技巧 和 Linux 教程。

随时关注 developerWorks 技术活动和网络广播。

获得产品和技术

以最适合您的方式 IBM 产品评估试用版软件：下载产品试用版，在线试用产品，在云环境下使用产品，或者在IBM SOA 人员沙箱 中花费几个小时来学习如何高效实现面向服务的架构。

讨论

加入 developerWorks 中文社区，developerWorks 社区是一个面向全球 IT 专业人员，可以提供博客、书签、wiki、群组、联系、共享和协作等社区功能的专业社交网络社区。

加入 IBM 软件下载与技术交流群组，参与在线交流。