NAME
salloc – Obtain a SLURM job allocation (a set of nodes), execute a command, and then release the allocation when the command is finished.

SYNOPSIS
salloc [options] [command [command args]]

DESCRIPTION
salloc is used to allocate a SLURM job allocation, which is a set of resources (nodes), possibly with some set of constraints (e.g. number of processors per node). When salloc successfully obtains the requested allocation, it then runs the command specified by the user. Finally, when the user specified command is complete, salloc relinquishes the job allocation.

The command may be any program the user wishes. Some typical commands are xterm, a shell script containing srun commands, and srun (see the EXAMPLES section). If no command is specified, then the value of SallocDefaultCommand in slurm.conf is used. If SallocDefaultCommand is not set, then salloc runs the user’s default shell.

The following document describes the the influence of various options on the allocation of cpus to jobs and tasks.
http://slurm.schedmd.com/cpu_management.html

NOTE: The salloc logic includes support to save and restore the terminal line settings and is designed to be executed in the foreground. If you need to execute salloc in the background, set its standard input to some file, for example: "salloc -n16 a.out </dev/null &"

OPTIONS
--A, --account=<account>
Charge resources used by this job to specified account. The account is an arbitrary string. The account name may be changed after job submission using the scontrol command.

--acctg-freq
Define the job accounting and profiling sampling intervals. This can be used to override the JobAcctGatherFrequency parameter in SLURM’s configuration file, slurm.conf. The supported format is as follows:

--acctg-freq=<datatype>=<interval>
where <datatype>=<interval> specifies the task sampling interval for the jobacct_gather plugin or a sampling interval for a profiling type by the acct_gather_profile plugin. Multiple, comma-separated <datatype>=<interval> intervals may be specified. Supported datatypes are as follows:

task=<interval>
where <interval> is the task sampling interval in seconds for the jobacct_gather plugins and for task profiling by the acct_gather_profile plugin. NOTE: This frequency is used to monitor memory usage. If memory limits are enforced the highest frequency a user can request is what is configured in the slurm.conf file. They can not turn it off (=0) either.

energy=<interval>
where <interval> is the sampling interval in seconds for energy profiling using the acct_gather_energy plugin
network=<interval>
  where <interval> is the sampling interval in seconds for infiniband profiling using the acct_gather_infiniband plugin.

filesystem=<interval>
  where <interval> is the sampling interval in seconds for filesystem profiling using the acct_gather_filesystem plugin.

The default value for the task sampling interval is 30. The default value for all other intervals is 0. An interval of 0 disables sampling of the specified type. If the task sampling interval is 0, accounting information is collected only at job termination (reducing SLURM interference with the job).

Smaller (non-zero) values have a greater impact upon job performance, but a value of 30 seconds is not likely to be noticeable for applications having less than 10,000 tasks.

-B --extra-node-info=<sockets:[cores]:[threads]>
Request a specific allocation of resources with details as to the number and type of computational resources within a cluster: number of sockets (or physical processors) per node, cores per socket, and threads per core. The total amount of resources being requested is the product of all of the terms. Each value specified is considered a minimum. An asterisk (*) can be used as a placeholder indicating that all available resources of that type are to be utilized. As with nodes, the individual levels can also be specified in separate options if desired:
  --sockets-per-node=<sockets>
  --cores-per-socket=<cores>
  --threads-per-core=<threads>

If SelectType is configured to select/cons_res, it must have a parameter of CR_Core, CR_Core_Memory, CR_Socket, or CR_Socket_Memory for this option to be honored. This option is not supported on BlueGene systems (select/bluegene plugin is configured). If not specified, the scontrol show job will display 'ReqS:C:T=*:*:*'.

--begin=<time>
Submit the batch script to the SLURM controller immediately, like normal, but tell the controller to defer the allocation of the job until the specified time.

Time may be of the form HH:MM:SS to run a job at a specific time of day (seconds are optional). (If that time is already past, the next day is assumed.) You may also specify midnight, noon, fika (3 PM) or teatime (4 PM) and you can have a time-of-day suffixed with AM or PM for running in the morning or the evening. You can also say what day the job will be run, by specifying a date of the form MMDDYY or MM/DD/YY YYYY-MM-DD. Combine date and time using the following format YYYY-MM-DD[THH:MM[:SS]]. You can also give times like now + count time-units, where the time-units can be seconds (default), minutes, hours, days, or weeks and you can tell SLURM to run the job today with the keyword today and to run the job tomorrow with the keyword tomorrow. The value may be changed after job submission using the scontrol command. For example:
  --begin=16:00
  --begin=now+1hour
  --begin=now+60 (seconds by default)
  --begin=2010-01-20T12:34:00

Notes on date/time specifications:
- Although the 'seconds' field of the HH:MM:SS time specification is allowed by the code, note that the poll time of the SLURM scheduler is not precise enough to guarantee dispatch of the job on the exact second. The job will be eligible to start on the next poll following the specified time. The exact poll interval depends on the SLURM scheduler (e.g., 60 seconds with the default
− If no time (HH:MM:SS) is specified, the default is (00:00:00).
− If a date is specified without a year (e.g., MM/DD) then the current year is assumed, unless the combination of MM/DD and HH:MM:SS has already passed for that year, in which case the next year is used.

---bell  Force salloc to ring the terminal bell when the job allocation is granted (and only if stdout is a tty).
By default, salloc only rings the bell if the allocation is pending for more than ten seconds (and only if stdout is a tty). Also see the option ---no-bell.

---comment=<string>
An arbitrary comment.

-C, ---constraint=<list>
Nodes can have features assigned to them by the SLURM administrator. Users can specify which of these features are required by their job using the constraint option. Only nodes having features matching the job constraints will be used to satisfy the request. Multiple constraints may be specified with AND, OR, matching OR, resource counts, etc. Supported constraint options include:

**Single Name**
Only nodes which have the specified feature will be used. For example, ---constraint="intel"

**Node Count**
A request can specify the number of nodes needed with some feature by appending an asterisk and count after the feature name. For example "---nodes=16 ---constraint=graphics*4 ..." indicates that the job requires 16 nodes at that at least four of those nodes must have the feature "graphics."

**AND**
If only nodes with all of specified features will be used. The ampersand is used for an AND operator. For example, ---constraint="intel&gpu"

**OR**
If only nodes with at least one of specified features will be used. The vertical bar is used for an OR operator. For example, ---constraint="intel|amd"

**Matching OR**
If only one of a set of possible options should be used for all allocated nodes, then use the OR operator and enclose the options within square brackets. For example: "---constraint=[rack1|rack2|rack3|rack4]" might be used to specify that all nodes must be allocated on a single rack of the cluster, but any of those four racks can be used.

**Multiple Counts**
Specific counts of multiple resources may be specified by using the AND operator and enclosing the options within square brackets. For example: "---constraint=[rack1*2&rack2*4]" might be used to specify that two nodes must be allocated from nodes with the feature of "rack1" and four nodes must be allocated from nodes with the feature "rack2."

---contiguous
If set, then the allocated nodes must form a contiguous set. Not honored with the topology/tree or topology/3d_torus plugins, both of which can modify the node ordering.
--cores-per-socket=<cores>
Restrict node selection to nodes with at least the specified number of cores per socket. See additional information under -B option above when task/affinity plugin is enabled.

-c, --cpus-per-task=<ncpus>
Advise the SLURM controller that ensuing job steps will require ncpus number of processors per task. Without this option, the controller will just try to allocate one processor per task.

For instance, consider an application that has 4 tasks, each requiring 3 processors. If our cluster is comprised of quad-processors nodes and we simply ask for 12 processors, the controller might give us only 3 nodes. However, by using the --cpus-per-task=3 options, the controller knows that each task requires 3 processors on the same node, and the controller will grant an allocation of 4 nodes, one for each of the 4 tasks.

-d, --dependency=<dependency_list>
Defer the start of this job until the specified dependencies have been satisfied/completed. <dependency_list> is of the form <type:job_id[:job_id][,type:job_id[:job_id]]>. Many jobs can share the same dependency and these jobs may even belong to different users. The value may be changed after job submission using the scontrol command.

  after:job_id[:jobid...]
  This job can begin execution after the specified jobs have begun execution.

  afterany:job_id[:jobid...]
  This job can begin execution after the specified jobs have terminated.

  afternotok:job_id[:jobid...]
  This job can begin execution after the specified jobs have terminated in some failed state (non-zero exit code, node failure, timed out, etc).

  afterok:job_id[:jobid...]
  This job can begin execution after the specified jobs have successfully executed (ran to completion with an exit code of zero).

  expand:job_id
  Resources allocated to this job should be used to expand the specified job. The job to expand must share the same QOS (Quality of Service) and partition. Gang scheduling of resources in the partition is also not supported.

  singleton
  This job can begin execution after any previously launched jobs sharing the same job name and user have terminated.

-D, --chdir=<path>
Change directory to path before beginning execution. The path can be specified as full path or relative path to the directory where the command is executed.

--exclusive
The job allocation can not share nodes with other running jobs. The default shared/exclusive behavior depends on system configuration and the partition’s Shared option takes precedence over the job’s option.

-F, --nodefile=<node file>
Much like --nodelist, but the list is contained in a file of name node file. The node names of the list may also span multiple lines in the file. Duplicate node names in the file will be ignored.
The order of the node names in the list is not important; the node names will be sorted by SLURM.

\texttt{--get-user-env}[^\text{timeout}][^\text{mode}]

This option will load login environment variables for the user specified in the \texttt{--uid} option. The environment variables are retrieved by running something of this sort "su – <username> –c /usr/bin/env" and parsing the output. Be aware that any environment variables already set in \texttt{slalloc}'s environment will take precedence over any environment variables in the user's login environment. The optional \texttt{timeout} value is in seconds. Default value is 3 seconds. The optional \texttt{mode} value control the "su" options. With a \texttt{mode} value of "S", "su" is executed without the "–" option. With a \texttt{mode} value of "L", "su" is executed with the "–" option, replicating the login environment. If \texttt{mode} not specified, the mode established at SLURM build time is used. Example of use include "--get-user-env", "--get-user-env=10" "--get-user-env=10L", and "--get-user-env=S".

\textbf{NOTE:} This option only works if the caller has an effective uid of "root". This option was originally created for use by Moab.

\texttt{--gid=<group>}

Submit the job with the specified \texttt{group}'s group access permissions. \texttt{group} may be the group name or the numerical group ID. In the default Slurm configuration, this option is only valid when used by the user root.

\texttt{--gres=<list>}

Specifies a comma delimited list of generic consumable resources. The format of each entry on the list is "name[>:count]". The name is that of the consumable resource. The count is the number of those resources with a default value of 1. The specified resources will be allocated to the job on each node. The available generic consumable resources is configurable by the system administrator. A list of available generic consumable resources will be printed and the command will exit if the option argument is "help". Examples of use include "--gres=gpu:2,mic=1" and "--gres=help".

\texttt{--H, --hold}

Specify the job is to be submitted in a held state (priority of zero). A held job can no wb er eleased using \texttt{scontrol} to reset its priority (e.g. "scontrol release <job_id>”).

\texttt{--h, --help}

Display help information and exit.

\texttt{--hint=<type>}

Bind tasks according to application hints

\textbf{compute_bound}

Select settings for compute bound applications: use all cores in each socket, one thread per core

\textbf{memory_bound}

Select settings for memory bound applications: use only one core in each socket, one thread per core

\textbf{[no]multithread}

[don't] use extra threads with in-core multi-threading which can benefit communication intensive applications

\textbf{help}

show this help message
salloc(1) SLURM Commands salloc(1)

- **--immediate=[<seconds>]**
  exit if resources are not available within the time period specified. If no argument is given, resources must be available immediately for the request to succeed. By default, **--immediate** is off, and the command will block until resources become available. Since this option’s argument is optional, for proper parsing the single letter option must be followed immediately with the value and not include a space between them. For example ”−I60” and not ”−I 60”.

- **--job-name=<jobname>**
  Specify a name for the job allocation. The specified name will appear along with the job id number when querying running jobs on the system. The default job name is the name of the "command" specified on the command line.

- **--jobid=<jobid>**
  Allocate resources as the specified job id. NOTE: Only valid for user root.

- **--kill-command=[signal]**
  salloc always runs a user–specified command once the allocation is granted. salloc will wait indefinitely for that command to exit. If you specify the **--kill-command** option salloc will send a signal to your command any time that the SLURM controller tells salloc that its job allocation has been revoked. The job allocation can be revoked for a couple of reasons: someone used scan-cancel to revoke the allocation, or the allocation reached its time limit. If you do not specify a signal name or number and SLURM is configured to signal the spawned command at job termination, the default signal is SIGHUP for interactive and SIGTERM for non–interactive sessions. Since this option’s argument is optional, for proper parsing the single letter option must be followed immediately with the value and not include a space between them. For example “−K1” and not ”−K 1”.

- **--no–kill**
  Do not automatically terminate a job if one of the nodes it has been allocated fails. The user will assume the responsibilities for fault–tolerance should a node fail. When there is a node failure, any active job steps (usually MPI jobs) on that node will almost certainly suffer a fatal error, but with **--no–kill**, the job allocation will not be revoked so the user may launch new job steps on the remaining nodes in their allocation.

  By default SLURM terminates the entire job allocation if any node fails in its range of allocated nodes.

- **--licenses=<license>**
  Specification of licenses (or other resources available on all nodes of the cluster) which must be allocated to this job. License names can be followed by a colon and count (the default count is one). Multiple license names should be comma separated (e.g. ”--licenses=foo:4,bar”).

- **--distribution=**
  
  *<block|cyclic|arbitrary|plane>=[options]>[:<block|cyclic>]*

  Specify alternate distribution methods for remote processes. In salloc, this only sets environment variables that will be used by subsequent srun requests. This option controls the assignment of tasks to the nodes on which resources have been allocated, and the distribution of those resources to tasks for binding (task affinity). The first distribution method (before the “:”) controls the distribution of resources across nodes. The optional second distribution method (after the “:”) controls the distribution of resources across sockets within a node. Note that with select/cons_res, the number of cpus allocated on each socket and node may be different. Refer to http://slurm.schedmd.com/mc_support.html for more information on resource allocation,
assignment of tasks to nodes, and binding of tasks to CPUs.

First distribution method:

block  The block distribution method will distribute tasks to a node such that consecutive tasks share a node. For example, consider an allocation of three nodes each with two cpus. A four–task block distribution request will distribute those tasks to the nodes with tasks one and two on the first node, task three on the second node, and task four on the third node. Block distribution is the default behavior if the number of tasks exceeds the number of allocated nodes.

cyclic  The cyclic distribution method will distribute tasks to a node such that consecutive tasks are distributed over consecutive nodes (in a round–robin fashion). For example, consider an allocation of three nodes each with two cpus. A four–task cyclic distribution request will distribute those tasks to the nodes with tasks one and four on the first node, task two on the second node, and task three on the third node. Note that when SelectType is select/cons_res, the same number of CPUs may not be allocated on each node. Task distribution will be round–robin among all the nodes with CPUs yet to be assigned to tasks. Cyclic distribution is the default behavior if the number of tasks is no larger than the number of allocated nodes.

plane  The tasks are distributed in blocks of a specified size. The options include a number representing the size of the task block. This is followed by an optional specification of the task distribution scheme within a block of tasks and between the blocks of tasks. The number of tasks distributed to each node is the same as for cyclic distribution, but the taskids assigned to each node depend on the plane size. For more details (including examples and diagrams), please see http://slurm.schedmd.com/mc_support.html and http://slurm.schedmd.com/dist_plane.html

arbitrary  The arbitrary method of distribution will allocate processes in–order as listed in file designated by the environment variable SLURM_HOSTFILE. If this variable is listed it will override any other method specified. If not set the method will default to block. Inside the hostfile must contain at minimum the number of hosts requested and be one per line or comma separated. If specifying a task count (\(\text{-n, --ntasks=<number>}\)), your tasks will be laid out on the nodes in the order of the file.  

NOTE: The arbitrary distribution option on a job allocation only controls the nodes to be allocated to the job and not the allocation of CPUs on those nodes. This option is meant primarily to control a job step’s task layout in an existing job allocation for the srun command.

Second distribution method:

block  The block distribution method will distribute tasks to sockets such that consecutive tasks share a socket.

cyclic  The cyclic distribution method will distribute tasks to sockets such that consecutive tasks are distributed over consecutive sockets (in a round–robin fashion).

\(\text{--mail–type=<type>}\)

Notify user by email when certain event types occur. Valid type values are BEGIN, END, FAIL, REQUEUE, and ALL (any state change). The user to be notified is indicated with \(\text{--mail–user}\).
--mail-user=<user>
User to receive email notification of state changes as defined by --mail-type. The default value is the submitting user.

--mem=<MB>
Specify the real memory required per node in MegaBytes. Default value is DefMemPerNode and the maximum value is MaxMemPerNode. If configured, both of parameters can be seen using the scontrol show config command. This parameter would generally be used if whole nodes are allocated to jobs (SelectType=select/linear). Also see --mem-per-cpu. --mem and --mem-per-cpu are mutually exclusive. NOTE: Enforcement of memory limits currently relies upon the task/cgroup plugin or enabling of accounting, which samples memory use on a periodic basis (data need not be stored, just collected). In both cases memory use is based upon the job’s Resident Set Size (RSS). A task may exceed the memory limit until the next periodic accounting sample.

--mem-per-cpu=<MB>
Minimum memory required per allocated CPU in MegaBytes. Default value is DefMemPerCPU and the maximum value is MaxMemPerCPU (see exception below). If configured, both of parameters can be seen using the scontrol show config command. Note that if the job’s --mem-per-cpu value exceeds the configured MaxMemPerCPU, then the user’s limit will be treated as a memory limit per task; --mem-per-cpu will be reduced to a value no larger than MaxMemPerCPU; --cpus-per-task will be set and value of --cpus-per-task multiplied by the new --mem-per-cpu value will equal the original --mem-per-cpu value specified by the user. This parameter would generally be used if individual processors are allocated to jobs (SelectType=select/cons_res). If resources are allocated by the core, socket or whole nodes; the number of CPUs allocated to a job may be higher than the task count and the value of --mem-per-cpu should be adjusted accordingly. Also see --mem. --mem and --mem-per-cpu are mutually exclusive.

--mem_bind=[{quiet,verbose}].type
Bind tasks to memory. Used only when the task/affinity plugin is enabled and the NUMA memory functions are available. Note that the resolution of CPU and memory binding may differ on some architectures. For example, CPU binding may be performed at the level of the cores within a processor while memory binding will be performed at the level of nodes, where the definition of "nodes" may differ from system to system. The use of any type other than "none" or "local" is not recommended. If you want greater control, try running a simple test code with the options "--mem_bind=verbose,none" to determine the specific configuration.

NOTE: To have SLURM always report on the selected memory binding for all commands executed in a shell, you can enable verbose mode by setting the SLURM_MEM_BIND environment variable value to "verbose".

The following informational environment variables are set when --mem_bind is in use:

SLURM_MEM_BIND_VERBOSE
SLURM_MEM_BIND_TYPE
SLURM_MEM_BIND_LIST

See the ENVIRONMENT VARIABLES section for a more detailed description of the individual SLURM_MEM_BIND* variables.

Supported options include:
q[uiet] quietly bind before task runs (default)

v[erbose] 
verbosely report binding before task runs

no[ne] don’t bind tasks to memory (default)

rank bind by task rank (not recommended)

local Use memory local to the processor in use

map_mem:<list>
bind by mapping a node’s memory to tasks as specified where <list> is
<cpuid1>,<cpuid2>,...<cpuidN>. CPU IDs are interpreted as decimal values unless they
are preceded with ’0x’ in which case they interpreted as hexadecimal values (not recom-
mended)

mask_mem:<list>
bind by setting memory masks on tasks as specified where <list> is
<mask1>,<mask2>,...<maskN>. memory masks are always interpreted as hexadecimal
values. Note that masks must be preceded with a ’0x’ if they don’t begin with [0-9] so
they are seen as numerical values by srun.

help show this help message

--mincpus=<n>
Specify a minimum number of logical cpus/processors per node.

-N, --nodes=<minnodes[−maxnodes]>
Request that a minimum of <minnodes> nodes be allocated to this job. A maximum node count may
also be specified with maxnodes. If only one number is specified, this is used as both the mini-
um and maximum node count. The partition’s node limits supersede those of the job. If a job’s
node limits are outside of the range permitted for its associated partition, the job will be left in a
PENDING state. This permits possible execution at a later time, when the partition limit is
changed. If a job node limit exceeds the number of nodes configured in the partition, the job will
be rejected. Note that the environment variable SLURM_NNODES will be set to the count of
nodes actually allocated to the job. See the ENVIRONMENT VARIABLES section for more
information. If –N is not specified, the default behavior is to allocate enough nodes to satisfy the
requirements of the –n and –c options. The job will be allocated as many nodes as possible within
the range specified and without delaying the initiation of the job. The node count specification
may include a numeric value followed by a suffix of "k" (multiplies numeric value by 1.024) or
"m" (multiplies numeric value by 1,048,576).

-n, --ntasks=<number>
salloc does not launch tasks, it requests an allocation of resources and executed some command.
This option advises the SLURM controller that job steps run within this allocation will launch a
maximum of <number> tasks and sufficient resources are allocated to accomplish this. The default is
one task per node, but note that the --cpus-per-task option will change this default.

--network=<type>
Specify information pertaining to the switch or network. The interpretation of type is system
dependent. This option is supported when running Slurm on a Cray natively. It is used to request
using Network Performance Counters. Only one value per request is valid. All options are case
in-sensitive. In this configuration supported values include:

system
Use the system-wide network performance counters. Only nodes requested will be marked
in use for the job allocation. If the job does not fill up the entire system the rest of the
nodes are not able to be used by other jobs using NPC, if idle their state will appear as PerfCnTs. These nodes are still available for other jobs not using NPC.

**blade** Use the blade network performance counters. Only nodes requested will be marked in use for the job allocation. If the job does not fill up the entire blade(s) allocated to the job those blade(s) are not able to be used by other jobs using NPC, if idle their state will appear as PerfCnTs. These nodes are still available for other jobs not using NPC.

In all cases the job allocation request **must specify the** --exclusive option. Otherwise the request will be denied.

Also with any of these options steps are not allowed to share blades, so resources would remain idle inside an allocation if the step running on a blade does not take up all the nodes on the blade.

The **network** option is also supported on systems with IBM’s Parallel Environment (PE). See IBM’s LoadLeveler job command keyword documentation about the keyword "network" for more information. Multiple values may be specified in a comma separated list. All options are case insensitive. Supported values include:

**BULK_XFER[=resources]**

Enable bulk transfer of data using Remote Direct-Memory Access (RDMA). The optional resources specification is a numeric value which can have a suffix of "k", "K", "m", "M", "g" or "G" for kilobytes, megabytes or gigabytes. **NOTE:** The resources specification is not supported by the underlying IBM infrastructure as of Parallel Environment version 2.2 and no value should be specified at this time.

**CAU=count**

Number of Collective Acceleration Units (CAU) required. Applies only to IBM Power7-IH processors. Default value is zero. Independent CAU will be allocated for each programming interface (MPI, LAPI, etc.)

**DEVNAME=name**

Specify the device name to use for communications (e.g. "eth0" or "mlx4_0").

**DEVTYPE=type**

Specify the device type to use for communications. The supported values of type are: "IB" (InfiniBand), "HFI" (P7 Host Fabric Interface), "IPONLY" (IP-Only interfaces), "HPCE" (HP Ethernet), and "KMUX" (Kernel Emulation of HPCE). The devices allocated to a job must all be of the same type. The default value depends upon what hardware is available and in order of preferences is IPONLY (which is not considered in User Space mode), HFI, IB, HPCE, and KMUX.

**IMMED=count**

Number of immediate send slots per window required. Applies only to IBM Power7-IH processors. Default value is zero.

**INSTANCES=count**

Specify number of network connections for each task on each network connection. The default instance count is 1.

**IPV4**

Use Internet Protocol (IP) version 4 communications (default).

**IPV6**

Use Internet Protocol (IP) version 6 communications.

**LAPI**

Use the LAPI programming interface.

**MPI**

Use the MPI programming interface. MPI is the default interface.

**PAMI**

Use the PAMI programming interface.
Use the OpenSHMEM programming interface.

Use all available switch networks (default).

Use one available switch network.

Use the UPC programming interface.

Use User Space communications.

Some examples of network specifications:

**Instances=2,US,MPI,SN_ALL**
Create two user space connections for MPI communications on every switch network for each task.

**US,MPI,Instances=3,Devtype=IB**
Create three user space connections for MPI communications on every InfiniBand network for each task.

**IPV4,LAPI,SN_Single**
Create a IP version 4 connection for LAPI communications on one switch network for each task.

**Instances=2,US,LAPI,MPI**
Create two user space connections each for LAPI and MPI communications on every switch network for each task. Note that SN_ALL is the default option so every switch network is used. Also note that Instances=2 specifies that two connections are established for each protocol (LAPI and MPI) and each task. If there are two networks and four tasks on the node then a total of 32 connections are established (2 instances x 2 protocols x 2 networks x 4 tasks).

**--nice[=adjustment]**
Run the job with an adjusted scheduling priority within SLURM. With no adjustment value the scheduling priority is decreased by 100. The adjustment range is from $-10000$ (highest priority) to $10000$ (lowest priority). Only privileged users can specify a negative adjustment. NOTE: This option is presently ignored if `SchedulerType=sched/wiki` or `SchedulerType=sched/wiki2`.

**--ntasks-per-core=<ntasks>**
Request the maximum `ntasks` be invoked on each core. Meant to be used with the `--ntasks` option. Related to `--ntasks-per-node` except at the core level instead of the node level. NOTE: This option is not supported unless `SelectTypeParameters=CR_Core` or `SelectTypeParameters=CR_Core_Memory` is configured.

**--ntasks-per-socket=<ntasks>**
Request the maximum `ntasks` be invoked on each socket. Meant to be used with the `--ntasks` option. Related to `--ntasks-per-node` except at the socket level instead of the node level. NOTE: This option is not supported unless `SelectTypeParameters=CR_Socket` or `SelectTypeParameters=CR_Socket_Memory` is configured.

**--ntasks-per-node=<ntasks>**
Request that `ntasks` be invoked on each node. If used with the `--ntasks` option, the `--ntasks` option will take precedence and the `--ntasks-per-node` will be treated as a maximum count of tasks per node. Meant to be used with the `--nodes` option. This is related to `--cpus-per-task=ncpus`, but does not require knowledge of the actual number of cpus on each node. In some cases, it is more convenient to be able to request that no more than a specific
number of tasks be invoked on each node. Examples of this include submitting a hybrid
MPI/OpenMP app where only one MPI "task/rank" should be assigned to each node while allow-
ing the OpenMP portion to utilize all of the parallelism present in the node, or submitting a single
setup/cleanup/monitoring job to each node of a pre–existing allocation as one step in a larger job
script.

---no–bell
Silence salloc’s use of the terminal bell. Also see the option --bell.

---no–shell
immediately exit after allocating resources, without running a command. However, the SLURM
job will still be created and will remain active and will own the allocated resources as long as it is
active. You will have a SLURM job id with no associated processes or tasks. You can submit srun
commands against this resource allocation, if you specify the --jobid= option with the job id of
this SLURM job. Or, this can be used to temporarily reserve a set of resources so that other jobs
cannot use them for some period of time. (Note that the SLURM job is subject to the normal con-
straints on jobs, including time limits, so that eventually the job will terminate and the resources
will be freed, or you can terminate the job manually using the scancel command.)

-O, ---overcommit
Overcommit resources. When applied to job allocation, only one CPU is allocated to the job per
node and options used to specify the number of tasks per node, socket, core, etc. are ignored.
When applied to job step allocations (the srun command when executed within an existing job
allocation), this option can be used to launch more than one task per CPU. Normally, srun will
not allocate more than one process per CPU. By specifying ---overcommit you are explicitly
allowing more than one process per CPU. However no more than MAX_TASKS_PER_NODE
tasks are permitted to execute per node. NOTE: MAX_TASKS_PER_NODE is defined in the file
slurm.h and is not a variable, it is set at SLURM build time.

---priority=<value>
Request a specific job priority. May be subject to configuration specific constraints.

---profile=<all|none|energy|[task],[lustre],[network]]>
enables detailed data collection by the acct_gather_profile plugin. Detailed data are typically
time-series that are stored in an HDF5 file for the job.

All All data types are collected. (Cannot be combined with other values.)

None No data types are collected. This is the default.
(Cannot be combined with other values.)

Energy Energy data is collected.

Task Task (I/O, Memory, ...) data is collected.

Lustre Lustre data is collected.

Network Network (InfiniBand) data is collected.
### salloc(1) SLURM Commands salloc(1)

#### -p, --partition=<partition_names>
Request a specific partition for the resource allocation. If not specified, the default behavior is to allow the slurm controller to select the default partition as designated by the system administrator. If the job can use more than one partition, specify their names in a comma separate list and the one offering earliest initiation will be used with no regard given to the partition name ordering (although higher priority partitions will be considered first). When the job is initiated, the name of the partition used will be placed first in the job record partition string.

#### -Q, --quiet
Suppress informational messages from salloc. Errors will still be displayed.

#### --qos=<qos>
Request a quality of service for the job. QOS values can be defined for each user/cluster/account association in the SLURM database. Users will be limited to their association’s defined set of qos’s when the SLURM configuration parameter, AccountingStorageEnforce, includes “qos” in it’s definition.

#### --reservation=<name>
Allocate resources for the job from the named reservation.

#### -s, --share
The job allocation can share resources with other running jobs. The resources to be shared can be nodes, sockets, cores, or hyperthreads depending upon configuration. The default shared behavior depends on system configuration and the partition’s Shared option takes precedence over the job’s option. This option may result the allocation being granted sooner than if the --share option was not set and allow higher system utilization, but application performance will likely suffer due to competition for resources. Also see the --exclusive option.

#### -S, --core-spec=<num>
Count of specialized cores per node reserved by the job for system operations and not used by the application. The application will not use these cores, but will be charged for their allocation. Default value is zero.

#### --signal=<sig_num>[[@<sig_time>]]
When a job is within sig_time seconds of its end time, send it the signal sig_num. Due to the resolution of event handling by SLURM, the signal may be sent up to 60 seconds earlier than specified. sig_num may either be a signal number or name (e.g. "10" or "USR1"). sig_time must have integer value between zero and 65535. By default, no signal is sent before the job’s end time. If a sig_num is specified without any sig_time, the default time will be 60 seconds.

#### --sockets-per-node=<sockets>
Restrict node selection to nodes with at least the specified number of sockets. See additional information under --B option above when task/affinity plugin is enabled.

#### --switches=<count>[[@<max_time>]]
When a tree topology is used, this defines the maximum count of switches desired for the job allocation and optionally the maximum time to wait for that number of switches. If SLURM finds an allocation containing more switches than the count specified, the job remains pending until it either finds an allocation with desired switch count or the time limit expires. It there is no switch count limit, there is no delay in starting the job. Acceptable time formats include “minutes”, “minutes:seconds”, “hours:minutes:seconds”, “days–hours”, “days–hours:minutes” and
"days-hours:minutes:seconds". The job’s maximum time delay may be limited by the system administrator using the `SchedulerParameters` configuration parameter with the `max_switch_wait` parameter option. The default max-time is the `max_switch_wait` `SchedulerParameter`.

`-t, --time=<time>`
Set a limit on the total run time of the job allocation. If the requested time limit exceeds the partition’s time limit, the job will be left in a PENDING state (possibly indefinitely). The default time limit is the partition’s default time limit. When the time limit is reached, each task in each job step is sent SIGTERM followed by SIGKILL. The interval between signals is specified by the SLURM configuration parameter `KillWait`. A time limit of zero requests that no time limit be imposed. Acceptable time formats include "minutes", "minutes:seconds", "hours:minutes:seconds", "days-hours", "days-hours:minutes" and "days-hours:minutes:seconds".

`--threads-per-core=<threads>`
Restrict node selection to nodes with at least the specified number of threads per core. NOTE: "Threads" refers to the number of processing units on each core rather than the number of application tasks to be launched per core. See additional information under `-B` option above when task/affinity plugin is enabled.

`--time-min=<time>`
Set a minimum time limit on the job allocation. If specified, the job may have its `--time` limit lowered to a value no lower than `--time-min` if doing so permits the job to begin execution earlier than otherwise possible. The job’s time limit will not be changed after the job is allocated resources. This is performed by a backfill scheduling algorithm to allocate resources otherwise reserved for higher priority jobs. Acceptable time formats include "minutes", "minutes:seconds", "hours:minutes:seconds", "days-hours", "days-hours:minutes" and "days-hours:minutes:seconds".

`--tmp=<MB>`
Specify a minimum amount of temporary disk space.

`-u, --usage`
Display brief help message and exit.

`--uid=<user>`
Attempt to submit and/or run a job as `user` instead of the invoking user id. The invoking user’s credentials will be used to check access permissions for the target partition. This option is only valid for user root. This option may be used by user root may use this option to run jobs as a normal user in a RootOnly partition for example. If run as root, `salloc` will drop its permissions to the uid specified after node allocation is successful. `user` may be the user name or numerical user ID.

`-V, --version`
Display version information and exit.

`-v, --verbose`
Increase the verbosity of `salloc`’s informational messages. Multiple `-v`’s will further increase `salloc`’s verbosity. By default only errors will be displayed.
--W, --wait=<seconds>
This option has been replaced by --immediate=<seconds>.

--w, --nodelist=<node name list>
Request a specific list of hosts. Unless constrained by the maximum node count, the job will contain all of these hosts. The list may be specified as a comma-separated list of hosts, a range of hosts (host[1−5,7,...] for example), or a filename. The host list will be assumed to be a filename if it contains a "/" character. If you specify a maximum node count and the host list contains more nodes, the extra node names will be silently ignored. If you specify a minimum node or processor count larger than can be satisfied by the supplied host list, additional resources will be allocated on other nodes as needed. Duplicate node names in the list will be ignored. The order of the node names in the list is not important; the node names will be sorted by SLURM.

--wait-all-nodes=<value>
Controls when the execution of the command begins. By default the job will begin execution as soon as the allocation is made.

0 Begin execution as soon as allocation can be made. Do not wait for all nodes to be ready for use (i.e. booted).
1 Do not begin execution until all nodes are ready for use.

--wckey=<wckey>
Specify wckey to be used with job. If TrackWCKey=no (default) in the slurm.conf this value is ignored.

--x, --exclude=<node name list>
Explicitly exclude certain nodes from the resources granted to the job.

The following options support Blue Gene systems, but may be applicable to other systems as well.

--blrts-image=<path>
Path to blrts image for bluegene block. BGL only. Default from blugene.conf if not set.

--cnload-image=<path>
Path to compute node image for bluegene block. BGP only. Default from blugene.conf if not set.

--conn-type=<type>
Require the block connection type to be of a certain type. On Blue Gene the acceptable of type are MESH, TORUS and NAV. If NAV, or if not set, then SLURM will try to fit a what the DefaultConnType is set to in the bluegene.conf if that isn’t set the default is TORUS. You should not normally set this option. If running on a BGP system and wanting to run in HTC mode (only for 1 midplane and below). You can use HTC_S for SMP, HTC_D for Dual, HTC_V for virtual node mode, and HTC_L for Linux mode. For systems that allow a different connection type per dimension you can supply comma separated list of connection types may be specified, one for each dimension (i.e. M,T,T,T will give you a torus connection in all dimensions expect the first).

--g, --geometry=<XxYxZ> | <AxAxYxZ>
Specify the geometry requirements for the job. On BlueGene/L and BlueGene/P systems there are three numbers giving dimensions in the X, Y and Z directions, while on BlueGene/Q systems there are four numbers giving dimensions in the A, X, Y and Z directions and can not be used to allocate sub-blocks. For example "--geometry=1x2x3x4", specifies a block of nodes having 1 x 2
x \(3 \times 4 = 24\) nodes (actually midplanes on BlueGene).

---ioload-image=<path>
Path to io image for bluegene block. BGP only. Default from blugene.conf if not set.

---linux-image=<path>
Path to linux image for bluegene block. BGL only. Default from blugene.conf if not set.

---mloader-image=<path>
Path to mloader image for bluegene block. Default from blugene.conf if not set.

-R, --no-rotate
Disables rotation of the job’s requested geometry in order to fit an appropriate block. By default the specified geometry can rotate in three dimensions.

---ramdisk-image=<path>
Path to ramdisk image for bluegene block. BGL only. Default from blugene.conf if not set.

---reboot
Force the allocated nodes to reboot before starting the job.

INPUT ENVIRONMENT VARIABLES
Upon startup, salloc will read and handle the options set in the following environment variables. Note: Command line options always override environment variables settings.

SALLOC_ACCOUNT Same as --A, --account
SALLOC_ACCTG_FREQ Same as --acctg-freq
SALLOC_BELL Same as --bell
SALLOC_CONN_TYPE Same as --conn-type
SALLOC_CORE_SPEC Same as --core-spec
SALLOC_DEBUG Same as --v, --verbose
SALLOC_EXCLUSIVE Same as --exclusive
SLURM_EXIT_ERROR
Specifies the exit code generated when a SLURM error occurs (e.g. invalid options). This can be used by a script to distinguish application exit codes from various SLURM error conditions. Also see SLURM_EXIT_IMMEDIATE.

SLURM_EXIT_IMMEDIATE
Specifies the exit code generated when the --immediate option is used and resources are not currently available. This can be used by a script to distinguish application exit codes from various SLURM error conditions. Also see SLURM_EXIT_ERROR.

SALLOC_GEOMETRY
Same as --g, --geometry
SALLOC_HINT or SLURM_HINT
Same as --hint

SALLOC_IMMEDIATE
Same as -I, --immediate

SALLOC_JOBID
Same as --jobid

SALLOC_KILL_CMD
Same as -K, --kill-command

SALLOC_MEM_BIND
Same as --mem_bind

SALLOC_NETWORK
Same as --network

SALLOC_NO_BELL
Same as --no-bell

SALLOC_NO_ROTATE
Same as -R, --no-rotate

SALLOC_OVERCOMMIT
Same as -O, --overcommit

SALLOC_PARTITION
Same as -p, --partition

SALLOC_PROFILE
Same as --profile

SALLOC_QOS
Same as --qos

SALLOC_REQ_SWITCH
When a tree topology is used, this defines the maximum count of switches desired for the job allocation and optionally the maximum time to wait for that number of switches. See --switches.

SALLOC_RESERVATION
Same as --reservation

SALLOC_SIGNAL
Same as --signal

SALLOC_TIMELIMIT
Same as -t, --time

SALLOC_WAIT
Same as -W, --wait

SALLOC_WAIT_ALL_NODES
Same as --wait-all-nodes

SALLOC_WCKEY
Same as --wckey

SALLOC_WAIT4SWITCH
Max time waiting for requested switches. See --switches

OUTPUT ENVIRONMENT VARIABLES
salloc will set the following environment variables in the environment of the executed program:

BASIL_RESERVATION_ID
The reservation ID on Cray systems running ALPS/BASIL only.

SLURM DISTRIBUTION
Same as --distribution

SLURM_JOB_ID (and SLURM_JOBID for backwards compatibility)
The ID of the job allocation.

SLURM_JOB_CPUS_PER_NODE
Count of processors available to the job on this node. Note the select/linear plugin allocates entire nodes to jobs, so the value indicates the total count of CPUs on each node. The select/cons_res plugin allocates individual processors to jobs, so this number indicates the number of processors on each node allocated to the job allocation.
**SLURM_JOB_NODELIST** (and **SLURM_NODELIST** for backwards compatibility)
List of nodes allocated to the job.

**SLURM_JOB_NUM_NODES** (and **SLURM_NNODES** for backwards compatibility)
Total number of nodes in the job allocation.

**SLURM_JOB_PARTITION**
Name of the partition in which the job is running.

**SLURM_MEM_BIND**
Set to value of the --mem_bind option.

**SLURM_SUBMIT_DIR**
The directory from which `salloc` was invoked.

**SLURM_SUBMIT_HOST**
The hostname of the computer from which `salloc` was invoked.

**SLURM_NODE_ALIASES**
Sets of node name, communication address and hostname for nodes allocated to the job from the cloud. Each element in the set if colon separated and each set is comma separated. For example:
```
SLURM_NODE_ALIASES=ec0:1.2.3.4:foo,ec1:1.2.3.5:bar
```

**SLURM_NTASKS**
Same as `-n`, `--ntasks`

**SLURM_NTASKS_PER_NODE**
Set to value of the `--ntasks-per-node` option, if specified.

**SLURM_PROFILE**
Same as `--profile`

**SLURM_TASKS_PER_NODE**
Number of tasks to be initiated on each node. Values are comma separated and in the same order as **SLURM_NODELIST**. If two or more consecutive nodes are to have the same task count, that count is followed by "(x#)" where "#" is the repetition count. For example, "SLURM_TASKS_PER_NODE=2(x3),1" indicates that the first three nodes will each execute three tasks and the fourth node will execute one task.

**MPIRUN_NOALLOCATE**
Do not allocate a block on Blue Gene L/P systems only.

**MPIRUN_NOFREE**
Do not free a block on Blue Gene L/P systems only.

**MPIRUN_PARTITION**
The block name on Blue Gene systems only.

**SIGNALS**
While `salloc` is waiting for a PENDING job allocation, most signals will cause `salloc` to revoke the allocation request and exit.

However if the allocation has been granted and `salloc` has already started the specified command, then `salloc` will ignore most signals. `salloc` will not exit or release the allocation until the command exits. One notable exception is SIGHUP. A SIGHUP signal will cause `salloc` to release the allocation and exit without waiting for the command to finish. Another exception is SIGTERM, which will be forwarded to the spawned process.

**EXAMPLES**
To get an allocation, and open a new xterm in which `srun` commands may be typed interactively:
$ salloc -N16 xterm
salloc: Granted job allocation 65537
(at this point the xterm appears, and salloc waits for xterm to exit)
salloc: Relinquishing job allocation 65537

To grab an allocation of nodes and launch a parallel application on one command line (See the salloc man page for more examples):

    salloc -N5 srun -n10 myprogram

COPYING

Copyright (C) 2006–2007 The Regents of the University of California. Produced at Lawrence Livermore National Laboratory (cf, DISCLAIMER).
Copyright (C) 2008–2010 Lawrence Livermore National Security.
Copyright (C) 2010–2013 SchedMD LLC.

This file is part of SLURM, a resource management program. For details, see <http://slurm.schedmd.com/>.

SLURM is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

SLURM is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

SEE ALSO

    sinfo(1), sattach(1), sbatch(1), squeue(1), scancel(1), scontrol(1), slurm.conf(5), sched_setaffinity (2), numa (3)