Configuration

Tractor configuration settings can be broadly divided into controls for engine dispatching behavior, blade characteristics, user permissions, and data locations. Discussions by topic are linked below. The configuration settings themselves are kept in several files in the "config" area, with site overrides stored in a separate directory.

Engine	Engine initialization and scheduling controls.
Blades	Blade customizations for your execution environment.
Crews	Crew membership and their access policies.
PostgreSQL	PostgreSQL server process management.
Dashboard	Authentication and extensions of dashboard functionality.
Limits	Management of limits, a cornerstone of sharing and managing resources.
Logging	Management of system and command log files.
Login Management	Valid user and authentication management.
RenderMan for Maya	RenderMan for Maya configuration for Tractor.

Configuration Files

Tractor's settings are located in .config files. The "config" directory in the Tractor install area holds all of the files with their default settings. Studio-specific overrides to these files should go in a site-specific directory outside of the install area, whose location is given to tractor-engine using the --configdir (path) s tart-up option. Remember to set read and write permissions on these files appropriately for your site. The most commonly customized files include:

tractor.configtractor.config contains the basic configuration and policy settings for Tractor, such as license server location, the job database location, and basic scheduling mode. See the comments in the file itself for details.crews.configThe crews.config file contains crew definitions and basic job queue editing permissions. Crews are simply named lists of tractor users. Permissions are granted to individual users or to all members of a designated crew. See the comments in the file itself for details.blade.configThe blade.config file controls the basic scheduling attributes of hosts on the farm. Host configurations are described usingblade profiles, each of which may be applied to many blade servers.limits.configThe limits.config file specifies restrictions on the number of concurrently executing applications that Tractor will launch on the farm. Limit "tags" are arbitrary keywords associated with each application, and Tractor maintains a tally for each tag. For example, if there are only limited number of licenses available for an application across the whole studio, then an entry in limits.config can ensure that Tractor does not exceed that number of simultaneous launches of that program.

Site-Specific Configuration Overrides

During installation, the engine startup service is adjusted to point the engine to the directory where you maintain your custom configuration files. It is a good idea to keep your modifications in a directory *outside* the installation directory for any particular version of Tractor so that you can resuse them as new versions are installed and so that your changes are not lost if the product is reinstalled.

Starting with Tractor 2.0, your configuration directory may be more *sparse,* containing only copies of the files that you actually want to modified. Files not in your configuration directory will be read from the stock originals in the Tractor-2.0/config install area.

Furthermore, special handling is applied to the two files tractor.config and db.config -- your copies of these files in your --configdir directory can be reduced down to *only* the few key-value pairs that you actually want to alter. For these two files, the engine first loads the stock file from the intalled distribution config directory, then it loads the versions from your --configdir and overlays "deltas" into the resulting dictionary. Other config files are *not* treated this way to avoid complexity related to deleting or partially replacing nested stock entries.

Select an "EngineOwner" login

IMPORTANT -- If you are going to start the tractor-engine process as root, which is a useful default so that it can acquire the "protected" port 80 for connection requests, then you MUST edit the tractor.config entry called "EngineOwner" and specify a NON-root account that will own the engine and database processes after start-up. The database app will not allow itself to run as root, so the engine takes care of starting it as the login you specify in tractor.config. Many studios create an account specifically to own these types of service processes.

The "@merge" Convenience Function

The engine's configuration file parser supports a "@merge" simple mechanism for loading JSON "subfiles" into a parameter block. This can be used, for example, in blade.config when many profiles share the same environment handler definitions; the common text can be placed in a separate file which is then "merged" into each profile definition.

The parser examines value strings for text like "@merge('file.json')" and the string is replaced with the parsed contents of the named file. Note that the **@m erge call must be INSIDE a regular JSON string,** it must be enclosed in double-quotes.

Tractor Network Port Usage

The table below lists the network service ports used be various Tractor components. These *listener ports* allow the different pieces of Tractor on different hosts to communicate. Networking *firewalls* may block these connections by default, so you may need to add "firewall exceptions" (sometimes called "pinholes") for these ports. Not every port needs an exception on every machine!

Host	Port	Purpose
engine	80 /tcp	The main job queue listens on this port for all job spooling and Dashboard connections from all client components. Alternate ports (and interfaces) can be specified with "tractor-engine --port=NNNN" or with ListenerPort in tractor.config.
engine	80 /udp	The engine's UDP listener receives progress updates from blades, they are collected and sent to Dashboard sessions to draw progress bars on individual tasks. This port number is always set to the same numeric value as the main TCP listener, above.
engine	9180 /tcp	The default logging mechanism for command output involves sending each command's output stream to a centralized log archiver process running on the engine host. Studios with many blades or with heavy logging requirements should instead use direct writes.
engine	9876 /tcp	The PostgreSQL database server port. NOTE: in normal usage this port is only used on the "loopback" interface for private connections from tractor-engine itself. There is usually no need for any connections from external hosts, and thus no need to create firewall pinholes for this service.
blade	9005 /tcp	Tractor-blade listens for several types of asynchronous connections on this port. It is not strictly necessary for simple dispatching since the blade makes outbound-only requests to the engine for task assignements. However, interrupting a running job requires asynchronous connections, as do certain types of blade status reports. Use the "tractor-blade -L NNNN" parameter to specify an alternate port, the blade will register the value with the engine. Use "-L 0" (zero) to let the operating system assign a random unused port for this purpose; however this approach is not appropriate when firewall pinholes are needed.
engine, others	239.255.255.250 1900 /udp	The "engine discovery" multicast scheme (SSDP) uses the given multicast address and port to both request and listen for availability advertisements from the current engine. These broadcasts are optional, and often undesireable at large large studios.
licenseserver	9010 /tcp	The PixarLicenseServer process listens for license check-out requests on this port. Alternate port numbers are possible, but require an updated license from Customer Support.