NAME

launchctl -- Interfaces with launchd

SYNOPSIS

launchctl subcommand [arguments ...]

DESCRIPTION

launchctl interfaces with launchd to manage and inspect daemons, agents and XPC services.

SUBCOMMANDS

launchctl allows for detailed examination of launchd endpoints. A domain manages the execution policy for a collection of services. A service may be thought of as a virtual process that is always available to be spawned in response to demand. Each service has a collection of endpoints, and sending a message to one of those endpoints will cause the service to launch on demand. Domains advertise these endpoints in a shared namespace and may be thought of as synonymous with Mach bootstrap subsets.

Many subcommands in launchctl take a specifier which indicates the target domain or service for the subcommand. This specifier may take one of the following forms:

system/[service-name]
Targets the system domain or a service within the system domain. The system domain manages the root Mach bootstrap and is considered a privileged execution context. Anyone may read or query the system domain, but root privileges are required to make modifications.

user/<uid>/[service-name]
Targets the user domain for the given UID or a service within that domain. A user domain may exist independently of a logged-in user. User domains do not exist on iOS.

login/<asid>/[service-name]
Targets a user-login domain or service within that domain. A user-login domain is created when the user logs in at the GUI and is identified by the audit session identifier associated with that login. If a user domain has an associated login domain, the print subcommand will display the ASID of that login domain. User-login domains do not exist on iOS.

gui/<uid>/[service-name]
Another form of the login specifier. Rather than specifying a user-login domain by its ASID, this specifier targets the domain based on which user it is associated with and is generally more convenient.

Note: GUI domains and user domains share many resources. For the purposes of the Mach bootstrap name lookups, they are "flat", so they share the same set of registered names. But they still have discrete sets of services. So
when printing the user domain's contents, you may see many Mach bootstrap name registrations from services that exist in the GUI domain for that user, but you will not see the services themselves in that list.

pid/<pid>/[service-name]
Targets the domain for the given PID or a service within that domain. Each process on the system will have a PID domain associated with it that consists of the XPC services visible to that process which can be reached with
xpc_connection_create(3).

For instance, when referring to a service with the identifier com.apple.example loaded into the GUI domain of a user with UID 501, domain-target is gui/501/, service-name is com.apple.example, and service-target is
gui/501/com.apple.example.

SUBCOMMANDS
bootstrap | bootout domain-target [service-path service-path2 ...] | service-target
Bootstraps or removes domains and services. When service arguments are present, bootstraps and correspondingly removes their definitions into the domain. Services may be specified as a series of paths or a service identi-
fier. Paths may point to XPC service bundles, launchd.plist(5) s, or a directories containing a collection of either. If there were one or more errors while bootstrapping or removing a collection of services, the problematic
paths will be printed with the errors that occurred.

If no paths or service target are specified, these commands can either bootstrap or remove a domain specified as a domain target. Some domains will implicitly bootstrap pre-defined paths as part of their creation.

enable | disable service-target
Enables or disables the service in the requested domain. Once a service is disabled, it cannot be loaded in the specified domain until it is once again enabled. This state persists across

boots of the device. This subcommand
              may only target services within the system domain or user and user-login domains.

     kickstart [-kp] service-target
              Instructs launchd to run the specified service immediately, regardless of its configured launch conditions.

              -k   If the service is already running, kill the running instance before restarting the service.

              -p   Upon success, print the PID of the new process or the already-running process to stdout.

     attach [-ksx] service-target
              Attaches the system's debugger to the process currently backing the specified service. By default, if the service is not running, this subcommand will block until the service starts.

              -k   If the service is already running, kill the running instance.

: