NAME

spctl — SecAssessment system policy security

SYNOPSIS

spctl *--assess* [**-t** *type*] [**-**] *file ...*
spctl *--global-enable* | *--global-disable*
spctl *--enable* | *--disable* | *--remove* [**-t** *type*] [**--path** *path*]
[**--requirement** *requirement*] [**--anchor** *hash*] [**--hash** *hash*]
spctl *--status*

DESCRIPTION

spctl manages the security assessment policy subsystem.

This subsystem maintains and evaluates rules that determine whether the system allows the installation, execution, and other operations on files on the system.

spctl requires one command option that determines its principal operation:

--add Add rule(s) to the system-wide assessment rule database.

-a, --assess
Requests that **spctl** perform an assessment on the *files* given.

--disable
Disable one or more rules in the assessment rule database. Disabled rules are not considered when performing assessment, but remain in the database and can be re-enabled later.

--enable
Enable rule(s) in the assessment rule database, counteracting earlier disabling.

--disable

--global-disable
Disable the assessment subsystem altogether. Operations that would be denied by system policy will be allowed to proceed; assessment APIs always report success. Requires root access.

--global-enable
Enable the assessment subsystem. Operations that are denied by system policy will fail; assessment APIs report the truth. Requires root access.

--remove
Remove rule(s) from the assessment rule database.

--status
Query whether the assessment subsystem is enabled or disabled.

In addition, the following options are recognized:

--anchor
In rule update operations, indicates that the arguments are hashes of anchor certificates.

--continue
If the assessment of a file fails, continue assessing additional file arguments. Without this option, the first failed assessment terminates operation.

--hash In rule update operations, indicates that the arguments are code directory hashes.

--ignore-cache
Do not query or use the assessment object cache. This may significantly slow down operation. Newly generated assessments may still be stored in the cache.

--label *label*
Specifies a string label to attach to new rules, or find in existing rules. Labels are arbitrary strings that are assigned by convention. Rule labels are optional.

−−no-cache
Do not place the outcome of any assessments into the assessment object cache. No other assessment may reuse this outcome. This option not prohibit the use of existing cache entries.

−−path In rule update operations, indicates that the argument(s) denote paths to files on disk.

−−priority *priority*
In rule update operations, specifies the priority of the rule(s) created or changed. Priorities are floating-point numbers. Higher numeric values indicate higher priority.

−−raw When displaying the outcome of an assessment, write it as a "raw" XML plist instead of parsing it in somewhat more friendly form. This is useful when used in scripts, or to access newly invented assessment aspects that **spctl** does not yet know about.

−−requirement
In rule update operations, indicates that the argument(s) are code requirement source.

−−reset-default
Unconditionally reset the system policy database to its default value. This discards all changes made by administrators. It also heals any corruption to the database. It does not implicitly either enable or disable the facility. This must be done as the super user. Reboot after use.

−−rule In rule update operations, indicates that the argument(s) are the index numbers of existing rules.

−t, −−type
Specify which type of assessment is desired: *execute* to assess code execution, *install* to assess installation of an installer package, and *open* to assess the opening of documents. The default is to assess execution.

−v, −−verbose
Requests more verbose output. Repeat the option or give it a higher numeric value to increase verbosity.

RULE SUBJECTS

The system assessement rule database contains entries that match candidates based on Code Requirements. **spctl** allows you to specify these requirements directly using the **−−requirement** option. In addition, individual programs on disk can be addressed with the --path option (which uses their Designated Requirement). The **−−anchor** option takes the hash of a (full) certificate and turns it into a requirement matching any signature based on that anchor certificate. Alternatively, it can take the absolute path of a certificate file on disk, containing the DER form of an anchor certificate. Finally, the **−−hash** option generates a code requirement that denotes only and exactly one program whose CodeDirectory hash is given. The means of specifying subjects does not affect the remaining processing.

FILES

`/var/db/SystemPolicy` The system policy database.

`/var/db/.SystemPolicy-default`
A copy of the initial distribution version of the system policy database. Useful for starting over if the database gets messed up beyond recognition.

EXAMPLES

To check whether Mail.app is allowed to run on the local system:

```
spctl -a /Applications/Mail.app
```

To allow Frobozz.app to run on the local system:

```
spctl --add --label "My Stuff" /Applications/Frobozz.app
```

To forbid all code obtained from the Mac App Store from running:

```
spctl --disable --label "Mac App Store"
```

DIAGNOSTICS

spctl exits zero on success, or one if an operation has failed. Exit code two indicates unrecognized or unsuitable arguments. If an assessment operation results in denial but no other problem has occurred, the exit code is three.

SEE ALSO

codesign(1), syspolicyd(1)

HISTORY

The system policy facility and **spctl** command first appeared in Mac OS X Lion 10.7.3 as a limited developer preview.