Seccomp security profiles for Docker

Secure computing mode (`seccomp`) is a Linux kernel feature. You can use it to restrict the actions available within the container. The `seccomp()` system call operates on the seccomp state of the calling process. You can use this feature to restrict your application's access.

This feature is available only if Docker has been built with `seccomp` and the kernel is configured with `CONFIG_SECCOMP` enabled. To check if your kernel supports `seccomp`:

```
$ grep CONFIG_SECCOMP= /boot/config-$(uname -r)
CONFIG_SECCOMP=y
```


Pass a profile for a container

The default `seccomp` profile provides a sane default for running containers with seccomp and disables around 44 system calls out of 300+. It is moderately protective while providing wide application compatibility. The default Docker profile can be found here.

In effect, the profile is an allowlist which denies access to system calls by default, then allowlists specific system calls. The profile works by defining a `defaultAction` of `SCMP_ACT_ERRNO` and overriding that action only for specific system calls. The effect of `SCMP_ACT_ERRNO` is to cause a `Permission Denied` error. Next, the profile defines a specific list of system calls which are fully allowed, because their `action` is overridden to be `SCMP_ACT_ALLOW`. Finally, some specific rules are for individual system calls such as `personality`, and others, to allow variants of those system calls with specific arguments.

`seccomp` is instrumental for running Docker containers with least privilege. It is not recommended to change the default `seccomp` profile.

When you run a container, it uses the default profile unless you override it with the `--security-opt` option. For example, the following explicitly specifies a policy:

```
$ docker run --rm \
             -it \
             --security-opt seccomp=/path/to/seccomp/profile.json \
             hello-world
```

Significant syscalls blocked by the default profile

Docker's default seccomp profile is an allowlist which specifies the calls that are allowed. The table below lists the significant (but not all) syscalls that are effectively blocked because they are not on the Allowlist. The table includes the reason each syscall is blocked rather than white-listed.

Syscall	Description
`acct`	Accounting syscall which could let containers disable their own resource limits or process accounting. Also gated by `CAP_SYS_PACCT`.
`add_key`	Prevent containers from using the kernel keyring, which is not namespaced.
`bpf`	Deny loading potentially persistent bpf programs into kernel, already gated by `CAP_SYS_ADMIN`.
`clock_adjtime`	Time/date is not namespaced. Also gated by `CAP_SYS_TIME`.
`ock_settime`	Time/date is not namespaced. Also gated by `CAP_SYS_TIME`.

Syscall	Description
`clone`	Deny cloning new namespaces. Also gated by `CAP_SYS_ADMIN` for CLONE_* flags, except `CLONE_NEWUSER`.
`create_module`	Deny manipulation and functions on kernel modules. Obsolete. Also gated by `CAP_SYS_MODULE`.
`delete_module`	Deny manipulation and functions on kernel modules. Also gated by `CAP_SYS_MODULE`.
`finit_module`	Deny manipulation and functions on kernel modules. Also gated by `CAP_SYS_MODULE`.
`get_kernel_syms`	Deny retrieval of exported kernel and module symbols. Obsolete.
`get_mempolicy`	Syscall that modifies kernel memory and NUMA settings. Already gated by `CAP_SYS_NICE`.
`init_module`	Deny manipulation and functions on kernel modules. Also gated by `CAP_SYS_MODULE`.
`ioperm`	Prevent containers from modifying kernel I/O privilege levels. Already gated by `CAP_SYS_RAWIO`.
`iopl`	Prevent containers from modifying kernel I/O privilege levels. Already gated by `CAP_SYS_RAWIO`.
`kcmp`	Restrict process inspection capabilities, already blocked by dropping `CAP_SYS_PTRACE`.
`kexec_file_load`	Sister syscall of `kexec_load` that does the same thing, slightly different arguments. Also gated by `CAP_SYS_BOOT`.
`kexec_load`	Deny loading a new kernel for later execution. Also gated by `CAP_SYS_BOOT`.
`keyctl`	Prevent containers from using the kernel keyring, which is not namespaced.
`lookup_dcookie`	Tracing/profiling syscall, which could leak a lot of information on the host. Also gated by `CAP_SYS_ADMIN`.
`mbind`	Syscall that modifies kernel memory and NUMA settings. Already gated by `CAP_SYS_NICE`.
`mount`	Deny mounting, already gated by `CAP_SYS_ADMIN`.
`move_pages`	Syscall that modifies kernel memory and NUMA settings.
`nfsservctl`	Deny interaction with the kernel nfs daemon. Obsolete since Linux 3.1.
`open_by_handle_at`	Cause of an old container breakout. Also gated by `CAP_DAC_READ_SEARCH`.
`perf_event_open`	Tracing/profiling syscall, which could leak a lot of information on the host.
`personality`	Prevent container from enabling BSD emulation. Not inherently dangerous, but poorly tested, potential for a lot of kernel vulns.
`pivot_root`	Deny `pivot_root`, should be privileged operation.
`process_vm_readv`	Restrict process inspection capabilities, already blocked by dropping `CAP_SYS_PTRACE`.
`process_vm_writev`	Restrict process inspection capabilities, already blocked by dropping `CAP_SYS_PTRACE`.
`ptrace`	Tracing/profiling syscall. Blocked in Linux kernel versions before 4.8 to avoid seccomp bypass. Tracing/profiling arbitrary processes is already blocked by dropping `CAP_SYS_PTRACE`, because it could leak a lot of information on the host.
`query_module`	Deny manipulation and functions on kernel modules. Obsolete.
`quotactl`	Quota syscall which could let containers disable their own resource limits or process accounting. Also gated by `CAP_SYS_ADMIN`.
`reboot`	Don't let containers reboot the host. Also gated by `CAP_SYS_BOOT`.
`request_key`	Prevent containers from using the kernel keyring, which is not namespaced.
`set_mempolicy`	Syscall that modifies kernel memory and NUMA settings. Already gated by `CAP_SYS_NICE`.
`setns`	Deny associating a thread with a namespace. Also gated by `CAP_SYS_ADMIN`.
`settimeofday`	Time/date is not namespaced. Also gated by `CAP_SYS_TIME`.
`stime`	Time/date is not namespaced. Also gated by `CAP_SYS_TIME`.
`swapon`	Deny start/stop swapping to file/device. Also gated by `CAP_SYS_ADMIN`.

Syscall	Description
`swapoff`	Deny start/stop swapping to file/device. Also gated by `CAP_SYS_ADMIN`.
`sysfs`	Obsolete syscall.
`_sysctl`	Obsolete, replaced by /proc/sys.
`umount`	Should be a privileged operation. Also gated by `CAP_SYS_ADMIN`.
`umount2`	Should be a privileged operation. Also gated by `CAP_SYS_ADMIN`.
`unshare`	Deny cloning new namespaces for processes. Also gated by `CAP_SYS_ADMIN`, with the exception of `unshare --user`.
`uselib`	Older syscall related to shared libraries, unused for a long time.
`userfaultfd`	Userspace page fault handling, largely needed for process migration.
`ustat`	Obsolete syscall.
`vm86`	In kernel x86 real mode virtual machine. Also gated by `CAP_SYS_ADMIN`.
`vm86old`	In kernel x86 real mode virtual machine. Also gated by `CAP_SYS_ADMIN`.

Run without the default seccomp profile

You can pass `unconfined` to run a container without the default seccomp profile.

```
$ docker run --rm -it --security-opt seccomp=unconfined debian:jessie \
    unshare --map-root-user --user sh -c whoami
```