# 10 Kubernetes Security Context settings you should understand

snyk

## 1. `runAsNonRoot` / 

Always set this to `true` to:

- enforce the use of non-root users for your pod's containers.
- limit access to any host resources that might mistakenly get exposed to the container.

## 2. `runAsUser/runAsGroup` / 

These settings can be used to enforce a specific runtime user and group.

Use with caution—these IDs must exist in the image for the container to run. Do not use these as a replacement for `runAsNonRoot`.

## 3. `seLinuxOptions` / 

This sets the `SELinux` context which is applied to the container or pod. Be aware when re-labeling `SELinux` contexts that this may allow unintended access.

## 4. `seccompProfile` / 

Be cautious when using `seccomp` profiles. Generally, it's okay to provide a profile that is *more* restrictive than the default, as long as your process can run under those restrictions. However, a less restrictive profile can potentially expose calls to the host system that could be dangerous.

## 5. `privileged` / `allowPrivilegeEscalation`

It is usually a bad idea to grant `privileged` access to containers. Use specific capability flags or other Kubernetes APIs instead.

In most cases, you should also explicitly set `allowPrivilegeEscalation` to `false` to stop processes from attaining higher privileges i.e. via `sudo, setuid`.

## 6. `capabilities`

Only provide the minimum required for your application to function. Linux capabilities provide fine-grained control over access to kernel-level calls.

## 7. `readonlyRootFilesystem`

Set this to `true` whenever possible. In the event your container was to get compromised, a read-write filesystem makes it easier for the attacker to install software or change configurations. Also, consider making any volumes mounted to your container read-only for similar reasons.

## 8. `procMount`

Do not change the `procMount` from the Default setting, unless you have very specific configurations—such as nested containers.

## 9. `fsGroup` / `fsGroupChangePolicy`

If other processes depend on the volume's pre-existing GID, changing ownership of a volume using `fsGroup` can have impacts on pod startup performance, as well as possible negative ramifications on shared file systems.

## 10. `sysctls`

Modification of kernel parameters via `sysctl` should be avoided—unless you have very specific requirements—as this may destabilize the host operating system.

Pod / Container


**Eric Smalling**
@ericsmalling
Sr. Dev. Advocate at Snyk

**Matt Jarvis**
@mattj_io
Sr. Dev. Advocate at Snyk