Rootless containers from scratch

Liz Rice

VP Open Source Engineering, Aqua Security

@lizrice

What is a container?

What is a rootless container?

By default, containers run as root

The same root as on the host

Warning:

Adding a user to the "docker" group grants them the ability to run containers which can be used to obtain root privileges on the Docker host. Refer to Docker Daemon Attack Surface for more information.

aqua

Note:

To install Docker without root privileges, see Run the Docker daemon as a non-root user (Rootless mode).

Rootless mode is currently available as an experimental feature.

aqua

Namespaces

Limit what a process can **see**

- Unix Timesharing System
- Process IDs
- Mounts
- Network
- InterProcess Comms
- User IDs

Namespaces

Limit what a process can **see**

- Unix Timesharing System
- Process IDs
- Mounts
- Network
- InterProcess Comms
- User IDs

man user_namespaces

Starting in Linux 3.8, unprivileged processes can create user namespaces

man user_namespaces

Starting in Linux 3.8, unprivileged processes can create user namespaces

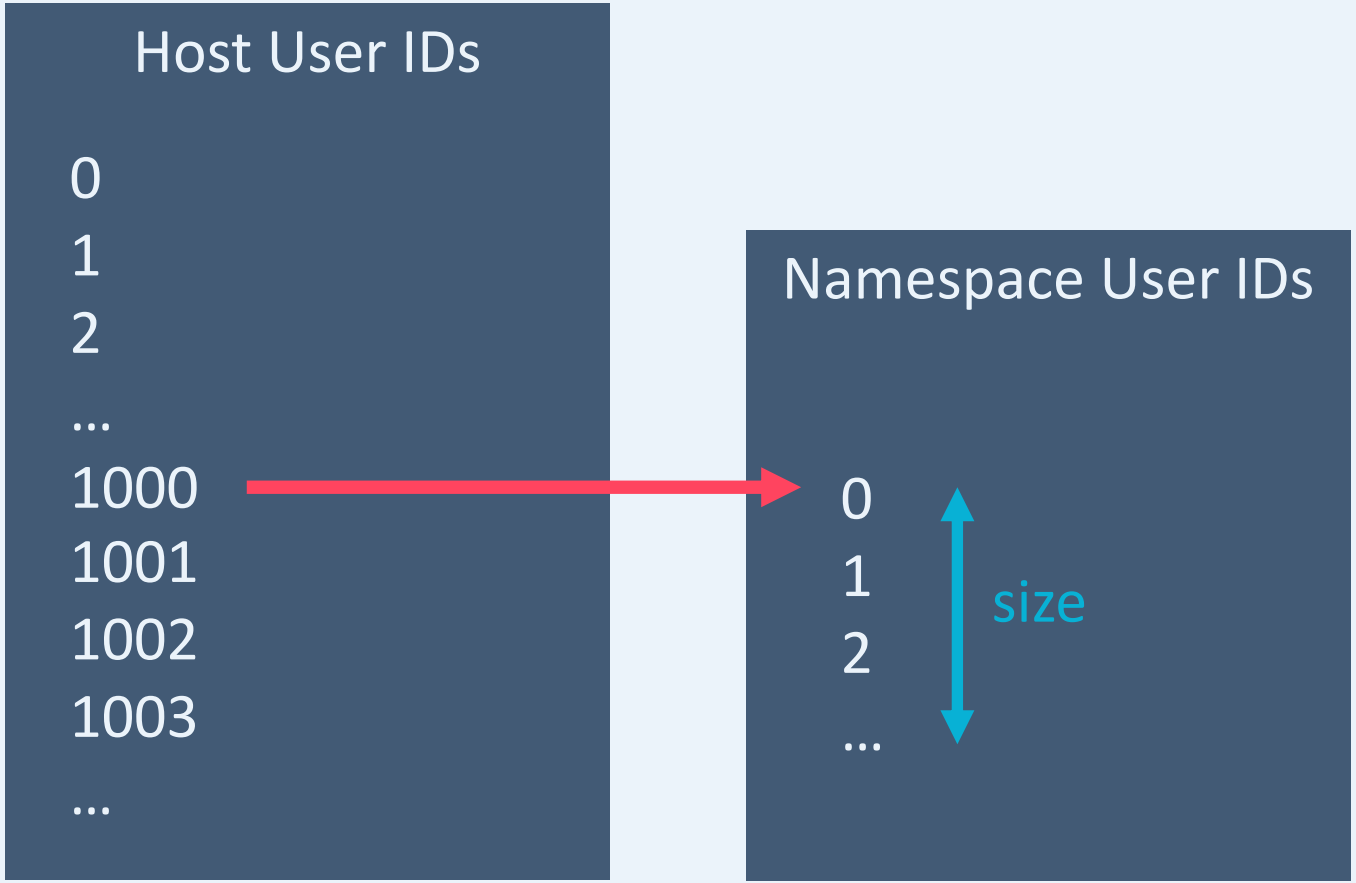
The child process created … with the CLONE_NEWUSER flag starts out with a complete set of capabilities in the new user namespace.

man user_namespaces

Starting in Linux 3.8, unprivileged processes can create user namespaces

The child process created … with the CLONE_NEWUSER flag starts out with a complete set of capabilities in the new user namespace.

If CLONE_NEWUSER is specified along with other CLONE_NEW* flags … the user namespace is guaranteed to be created first, giving the child … **privileges over the remaining namespaces** created by the call.


Host User IDs
0
1
2
...
1000
1001
1002
1003
...
Namespace User IDs
0
1
2
...
size

aqua

```go
func main() {
  switch os.Args[1] {
    case "run":
        run()
    case "child":
        child()
    default:
        panic("Missing argument 1")
    }
}

func run() {
  fmt.Printf("Running %v as user %d in process %d\n", os.Args[2:], os.Geteuid(), os.Getpid())

  cmd := exec.Command("/proc/self/exe", append([]string{"child"}, os.Args[2:]...)...)
  cmd.Stdout = os.Stdout
  cmd.Stderr = os.Stderr
  cmd.Stdin = os.Stdin
  cmd.SysProcAttr = &syscall.SysProcAttr{
    Cloneflags: syscall.CLONE_NEWUTS | syscall.CLONE_NEWUSER|syscall.CLONE_NEWNS|syscall.CLONE_NEWPID,
    UidMappings: []syscall.SysProcIDMap{{
      ContainerID: 0,
      HostID:      1000,
      Size:        1}},
  }
  must(cmd.Run())
}
```

```go
func child() {
    fmt.Printf("Running %v as user %d in process %d\n", os.Args[2:], os.Geteuid(), os.Getpid())

    must(syscall.Chroot("/home/vagrant/alpinefs"))
    must(os.Chdir("/"))
    must(syscall.Mount("proc", "proc", "proc", 0, ""))

    cmd := exec.Command(os.Args[2], os.Args[3:]...)
    cmd.Stdout = os.Stdout
    cmd.Stderr = os.Stderr
    cmd.Stdin = os.Stdin
    must(cmd.Run())

    must(syscall.Unmount("proc", 0))
}

func must(err error) {
   if err != nil {
      panic(err)
    }
}
```

Rootless container support

Changelog

For official release notes for Docker Engine CE, visit the release notes page.

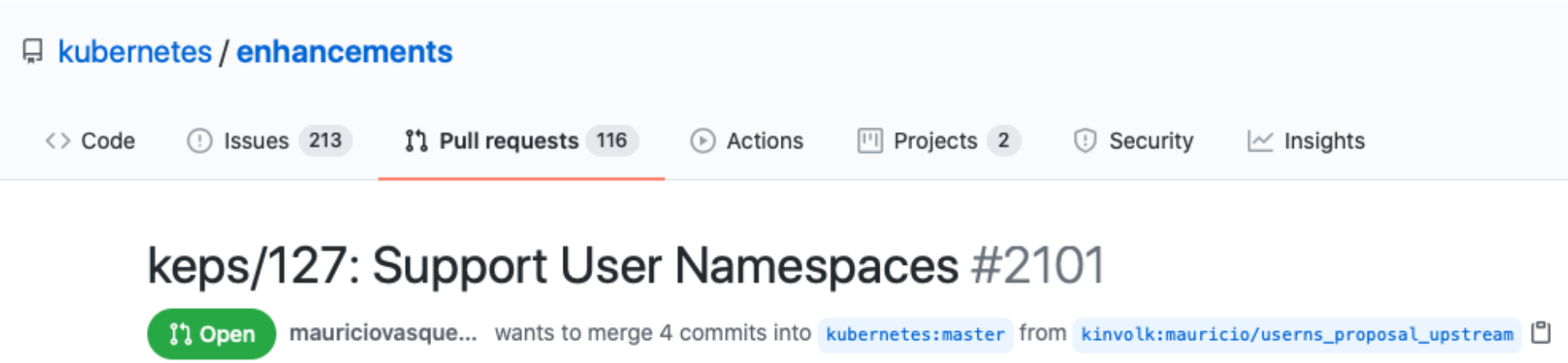
20.10.0

Rootless

- rootless: graduate from experimental moby/moby#40759

aqua

Rootless container support



aqua

Thank you

github.com/rootless-containers/rootlesskit

github.com/lizrice/containers-from-scratch