

containers / podman Public

<> Code Issues 935 Pull requests 149 Discussions Actions Projects

# Commit 43fbde4



Luap99 committed on Sep 4, 2025 Verified

kube play: don't follow volume symlinks onto the host

For ConfigMap and Secret kube play volumes podman populates the data from the yaml. However the volume content is not controlled by us and we can be tricked following a symlink to a file on the host instead.

Fixes: [CVE-2025-9566](#)

Signed-off-by: Paul Holzinger <pholzing@redhat.com>

main · v5.8.1 ... v5.7.0-rc1

1 parent [2499de2](#) commit 43fbde4

4 files changed +71 -4 lines changed

[↑ Top](#)

▼ pkg/domain/infra/abi

- play.go
- play\_linux.go
- play\_unsupported.go
- play\_utils.go

4 files changed +71 -4 lines changed



▼ pkg/domain/infra/abi/play.go



```
@@ -810,8 +810,7 @@ func (ic *ContainerEngine) playKubePod(ctx
context.Context, podName string, podY
```

```
810 810 defaultMode := v.DefaultMode
```

```

811 811 // Create files and add data to the volume mountpoint based on the
      Items in the volume
812 812     for k, v := range v.Items {
813 813         -         dataPath := filepath.Join(mountPoint, k)
814 814         -         f, err := os.Create(dataPath)
813 813         +         f, err := openPathSafely(mountPoint, k)
815 814         if err != nil {
816 815             return nil, nil, fmt.Errorf("cannot create file %q at
      volume mountpoint %q: %w", k, mountPoint, err)
817 816         }
@@ -821,7 +820,7 @@ func (ic *ContainerEngine) playKubePod(ctx
context.Context, podName string, podY
821 820         return nil, nil, err
822 821     }
823 822     // Set file permissions
824 822     -     if err := os.Chmod(f.Name(), os.FileMode(defaultMode)); err !=
      nil {
823 823     +     if err := f.Chmod(os.FileMode(defaultMode)); err != nil {
825 824         return nil, nil, err
826 825     }
827 826     }

```

▼ pkg/domain/infra/abi/play\_linux.go

```

... @@ -0,0 +1,18 @@
1 + //go:build !remote
2 +
3 + package abi
4 +
5 + import (
6 +     "os"
7 +
8 +     securejoin "github.com/cyphar/filepath-securejoin"
9 + )
10 +
11 + // openSymlinkPath opens the path under root using securejoin.OpenatInRoot().
12 + func openSymlinkPath(root *os.File, unsafePath string, flags int) (*os.File,
      error) {
13 +     file, err := securejoin.OpenatInRoot(root, unsafePath)
14 +     if err != nil {

```

```

15 +         return nil, err
16 +     }
17 +     return securejoin.Reopen(file, flags)
18 + }

```

▼ pkg/domain/infra/abi/play\_unsupported.go

```

... @@ -0,0 +1,13 @@
1 + //go:build !linux && !remote
2 +
3 + package abi
4 +
5 + import (
6 +     "errors"
7 +     "os"
8 + )
9 +
10 + // openSymlinkPath is not supported on this platform.
11 + func openSymlinkPath(root *os.File, unsafePath string, flags int) (*os.File,
    error) {
12 +     return nil, errors.New("cannot safely open symlink on this platform")
13 + }

```

▼ pkg/domain/infra/abi/play\_utils.go

```

... @@ -2,7 +2,14 @@
2 2
3 3     package abi
4 4
5 5 - import "github.com/containers/podman/v5/libpod/define"
6 6 + import (
7 7 +     "fmt"
8 8 +     "os"
9 9 +     "strings"
10 10 +     "github.com/containers/podman/v5/libpod/define"
11 11 +     "golang.org/x/sys/unix"
12 12 + )
13 13
14 14 // getSdNotifyMode returns the `sdNotifyAnnotation/$name` for the specified
15 15 // name. If name is empty, it'll only look for `sdNotifyAnnotation`.

```



```
@@ -16,3 +23,33 @@ func getSdNotifyMode(annotations map[string]string, name
string) (string, error)
```

```
16 23     }
17 24     return mode, define.ValidateSdNotifyMode(mode)
18 25     }

26 +
27 + // openPathSafely opens the given name under the trusted root path, the
    unsafeName
28 + // must be a single path component and not contain "/".
29 + // The resulting path will be opened or created if it does not exists.
30 + // Following of symlink is done within staying under root, escapes outside
31 + // of root are not allowed and prevent.
32 + //
33 + // This custom function is needed because securejoin.SecureJoin() is not race
    safe
34 + // and the volume might be mounted in another container that could swap in a
    symlink
35 + // after the function has run. securejoin.OpenInRoot() doesn't work either
    because
36 + // it cannot create files and doesn't work on freebsd.
37 + func openPathSafely(root, unsafeName string) (*os.File, error) {
38 +     if strings.Contains(unsafeName, "/") {
39 +         return nil, fmt.Errorf("name %q must not contain path separator",
    unsafeName)
40 +     }
41 +     fdDir, err := os.OpenFile(root, unix.O_RDONLY, 0)
42 +     if err != nil {
43 +         return nil, err
44 +     }
45 +     defer fdDir.Close()
46 +     flags := unix.O_CREAT | unix.O_WRONLY | unix.O_TRUNC | unix.O_CLOEXEC
47 +     fd, err := unix.Openat(int(fdDir.Fd()), unsafeName, flags|unix.O_NOFOLLOW,
    0o644)
48 +     if err == nil {
49 +         return os.NewFile(uintptr(fd), unsafeName), nil
50 +     }
51 +     if err == unix.ELOOP {
52 +         return openSymlinkPath(fdDir, unsafeName, flags)
53 +     }
54 +     return nil, &os.PathError{Op: "openat", Path: unsafeName, Err: err}
```

55 + }

**Comments** 0



Please [sign in](#) to comment.