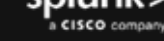
Linux (Pseudo) Filesystems

The Hidden Backbone of Cloud Native



Daniel Drack
@DrackThor

Senior DevOps Engineer

Host @ Cloud Native Days Austria
Founder @ Cloud Native Austria
Organizer @ Cloud Native Chapter Graz

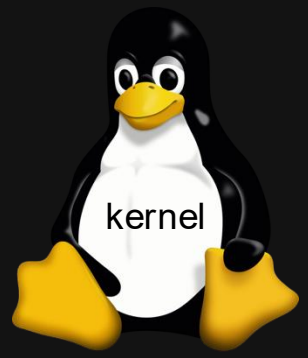
BSc | MA | MBA
Kubestronaut
SUSE | Exoscale | Snyk | GitLab | Scrum

What are containers?

Linux Filesystems

FS under the hood of containers

- Applications
- GUI
- SSH access
- Packages
- Toolchains
- Compilers




```
/
├── var
├── run
│   └── k3s
│       └── contianerd
│           └── ...
│               └── rootfs
├── bin
├── opt
└── etc
```

PSTREE

Users

root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
.....
drad:x:1006:1006:Daniel Drack:/home/drad:/bin/bash
grep:x:1007:1007:Paul Greuter:/home/grep:/bin/bash

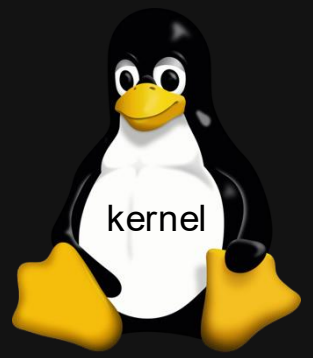
network
IPC
users
mount
cgroups
PID
UTS

CPU

Memory

Storage

Linux kernel namespace




```
UID          PID    PPID  C STIME TTY          TIME CMD
1001           1       0  0 Jan21 ?        01:22:21 redis-server
*:6379
1001     2297276       0  0 12:53 pts/0    00:00:00 sh
1001     2297338 2297276  0 12:53 pts/0    00:00:00 ps -eaf
```

```
/
├── var
├── bin
├── opt
├── usr
│   └── sbin
│       └── nginx
└── etc
```

```
$ hostname
container-1
```

Users

root:x:0:0:root:/root:/bin/bash
app:x:1006:1006:app:/app:/bin/bash

CPU | Memory

```
/
├── var
├── run
│   └── k3s
│       └── contianerd
│           └── ...
│               └── rootfs
├── bin
├── opt
└── etc
```

PSTREE

```
UID        PID     CMD
1006       2297279 CONTAINER
1006       2297276 sh
1006       2297338 ps -eaf
```

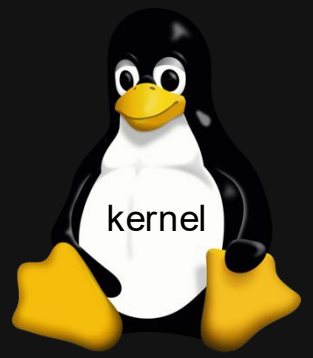
Users

root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
.....
drad:x:1006:1006:Daniel Drack:/home/drad:/bin/bash
grep:x:1007:1007:Paul Greuter:/home/grep:/bin/bash

network
IPC
users
mount
cgroups
PID
UTS

CPU | Memory

Storage




```
UID          PID    PPID  C STIME TTY          TIME CMD
1001           1       0  0 Jan21 ?        01:22:21 redis-server
*:6379
1001     2297276       0  0 12:53 pts/0    00:00:00 sh
1001     2297338 2297276  0 12:53 pts/0    00:00:00 ps -eaf
```

```
/
├── var
├── bin
├── opt
├── usr
│   └── sbin
│       └── nginx
└── etc
```

```
$ hostname
container-1
```

Users

root:x:0:0:root:/root:/bin/bash
app:x:1006:1006:app:/app:/bin/bash

CPU | Memory

```
/
├── var
│   ├── run
│   │   └── k3s
│   │       └── contianerd
│   │           └── ...
│   │               └── rootfs
├── bin
├── opt
└── etc
```

PSTREE

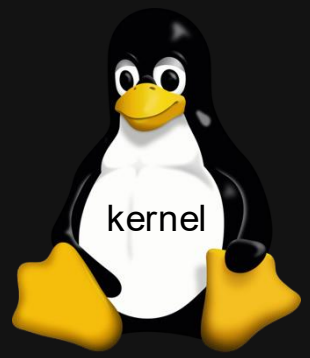
Users

root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
.....
drad:x:1006:1006:Daniel Drack:/home/drad:/bin/bash
grep:x:1007:1007:Paul Greuter:/home/grep:/bin/bash

network
IPC
users
mount
cgroups
PID
UTS

CPU | Memory

Storage




```
UID          PID    PPID  C STIME TTY          TIME CMD
1001           1       0  0 Jan21 ?        01:22:21 redis-server
*:6379
1001     2297276       0  0 12:53 pts/0    00:00:00 sh
1001     2297338 2297276  0 12:53 pts/0    00:00:00 ps -eaf
```

```
/
├── var
├── bin
├── opt
├── usr
│   └── sbin
│       └── nginx
└── etc
```

```
$ hostname
container-1
```

Users

root:x:0:0:root:/root:/bin/bash
app:x:1006:1006:app:/app:/bin/bash

CPU | Memory

```
/
├── var
├── run
│   └── k3s
│       └── contianerd
│           └── ...
│               └── rootfs
├── bin
├── opt
└── etc
```

PSTREE

Users

root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
.....
drad:x:1006:1006:Daniel Drack:/home/drad:/bin/bash
grep:x:1007:1007:Paul Greuter:/home/grep:/bin/bash

network
IPC
users
mount
cgroups
PID
UTS

CPU | Memory

Storage

```
UID          PID    PPID  C STIME TTY          TIME CMD
1001           1       0  0 Jan21 ?        01:22:21 redis-server
*:6379
1001     2297276       0  0 12:53 pts/0    00:00:00 sh
1001     2297338 2297276  0 12:53 pts/0    00:00:00 ps -eaf
```

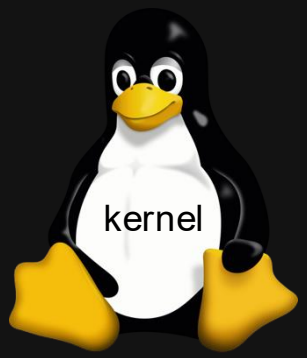
```
/
├── var
├── bin
├── opt
├── usr
│   └── sbin
│       └── nginx
└── etc
```

```
$ hostname
container-1
```

Users

root:x:0:0:root:/root:/bin/bash
app:x:1006:1006:app:/app:/bin/bash

CPU | Memory




```
/
├── var
│   ├── run
│   │   └── k3s
│   │       └── contianerd
│   │           └── ...
│   │               └── rootfs
├── bin
├── opt
└── etc
```

PSTREE

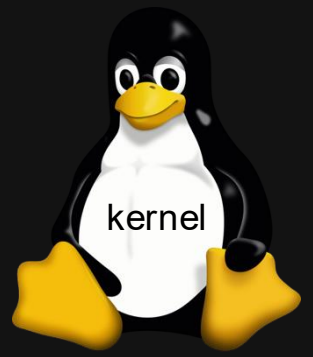
Users

root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
.....
drad:x:1006:1006:Daniel Drack:/home/drad:/bin/bash
grep:x:1007:1007:Paul Greuter:/home/grep:/bin/bash

network
IPC
users
mount
cgroups
PID
UTS

CPU | Memory

Storage




```
UID          PID    PPID  C STIME TTY          TIME CMD
1001           1       0  0 Jan21 ?        01:22:21 redis-server
*:6379
1001     2297276       0  0 12:53 pts/0    00:00:00 sh
1001     2297338 2297276  0 12:53 pts/0    00:00:00 ps -eaf
```

```
/
├── var
├── bin
├── opt
├── usr
│   └── sbin
│       └── nginx
└── etc
```

```
$ hostname
container-1
```

Users

root:x:0:0:root:/root:/bin/bash
app:x:1006:1006:app:/app:/bin/bash

CPU | Memory

```
/
├── var
│   ├── run
│   │   └── k3s
│   │       └── contianerd
│   │           └── ...
│   │               └── rootfs
├── bin
├── opt
└── etc
```

PSTREE

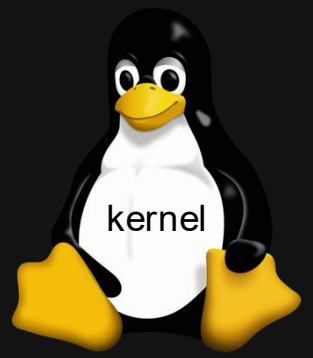
Users

root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
.....
drad:x:1006:1006:Daniel Drack:/home/drad:/bin/bash
grep:x:1007:1007:Paul Greuter:/home/grep:/bin/bash

network
IPC
users
mount
cgroups
PID
UTS

CPU | Memory

Storage




```
UID          PID    PPID  C STIME TTY          TIME CMD
1001           1       0  0 Jan21 ?        01:22:21 redis-server
*:6379
1001     2297276       0  0 12:53 pts/0    00:00:00 sh
1001     2297338 2297276  0 12:53 pts/0    00:00:00 ps -eaf
```

```
/
├── var
├── bin
├── opt
├── usr
│   └── sbin
│       └── nginx
└── etc
```

```
$ hostname
container-1
```

Users

root:x:0:0:root:/root:/bin/bash
app:x:1006:1006:app:/app:/bin/bash

CPU | Memory

```
/
├── var
│   ├── run
│   │   └── k3s
│   │       └── contianerd
│   │           └── ...
│   │               └── rootfs
├── bin
├── opt
└── etc
```

PSTREE

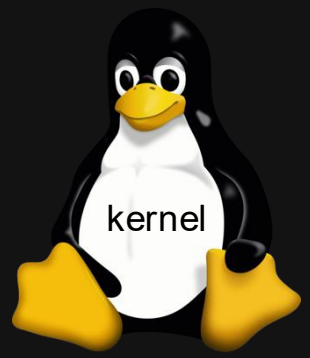
Users

root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
.....
drad:x:1006:1006:Daniel Drack:/home/drad:/bin/bash
grep:x:1007:1007:Paul Greuter:/home/grep:/bin/bash

network
IPC
users
mount
cgroups
PID
UTS

CPU | Memory

Storage




```
UID          PID    PPID  C STIME TTY          TIME CMD
1001           1       0  0 Jan21 ?        01:22:21 redis-server
*:6379
1001     2297276       0  0 12:53 pts/0    00:00:00 sh
1001     2297338 2297276  0 12:53 pts/0    00:00:00 ps -eaf
```

```
/
├── var
├── bin
├── opt
├── usr
│   └── sbin
│       └── nginx
└── etc
```

$ hostname
container-1

Users

root:x:0:0:root:/root:/bin/bash
app:x:1006:1006:app:/app:/bin/bash

CPU | Memory

```
/
├── var
│   ├── run
│   │   └── k3s
│   │       └── contianerd
│   │           └── ...
│   │               └── rootfs
├── bin
├── opt
└── etc
```

PSTREE

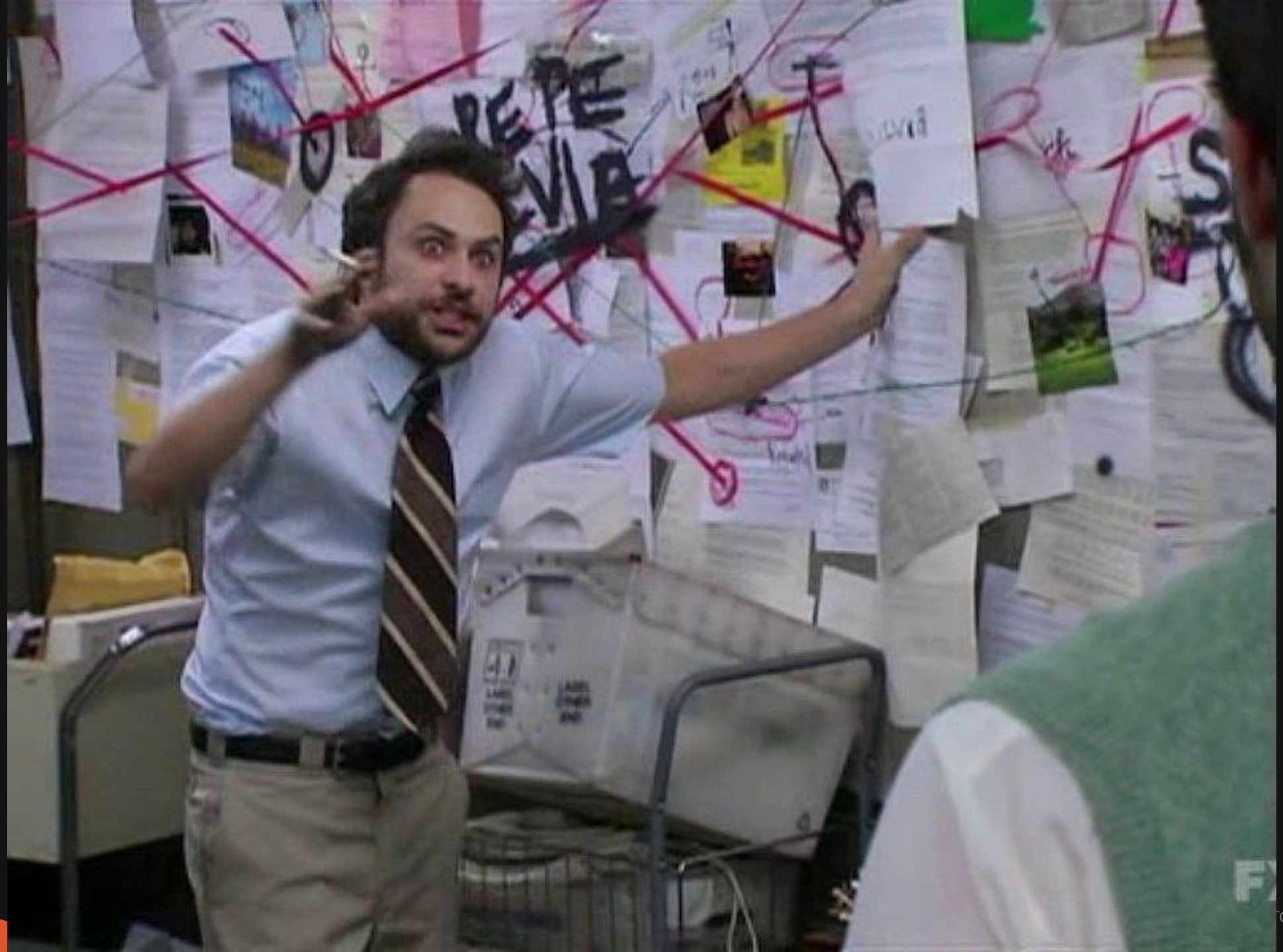
Users

root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
.....
drad:x:1006:1006:Daniel Drack:/home/drad:/bin/bash
grep:x:1007:1007:Paul Greuter:/home/grep:/bin/bash

network
IPC
users
mount
cgroups
PID
UTS

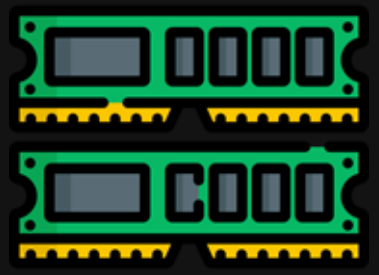
CPU | Memory

Storage



Linux Filesystems

```
/
├── var
├── run
│   └── k3s
│       └── contianerd
│           └── ...
│               └── rootfs
├── bin
├── tmp
└── etc
```

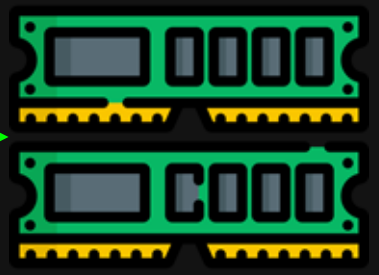


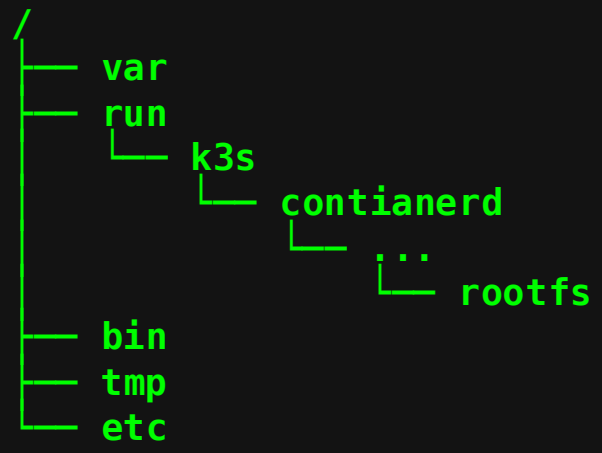
```
/
├── var
├── run
│   └── k3s
│       └── contianerd
│           └── ...
│               └── rootfs
├── bin
├── tmp
└── etc
```

ext4

FAT32

tmpFS



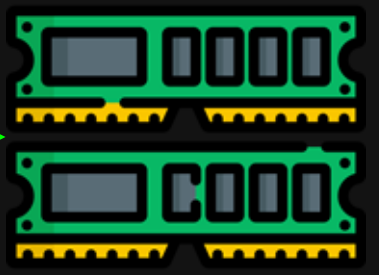


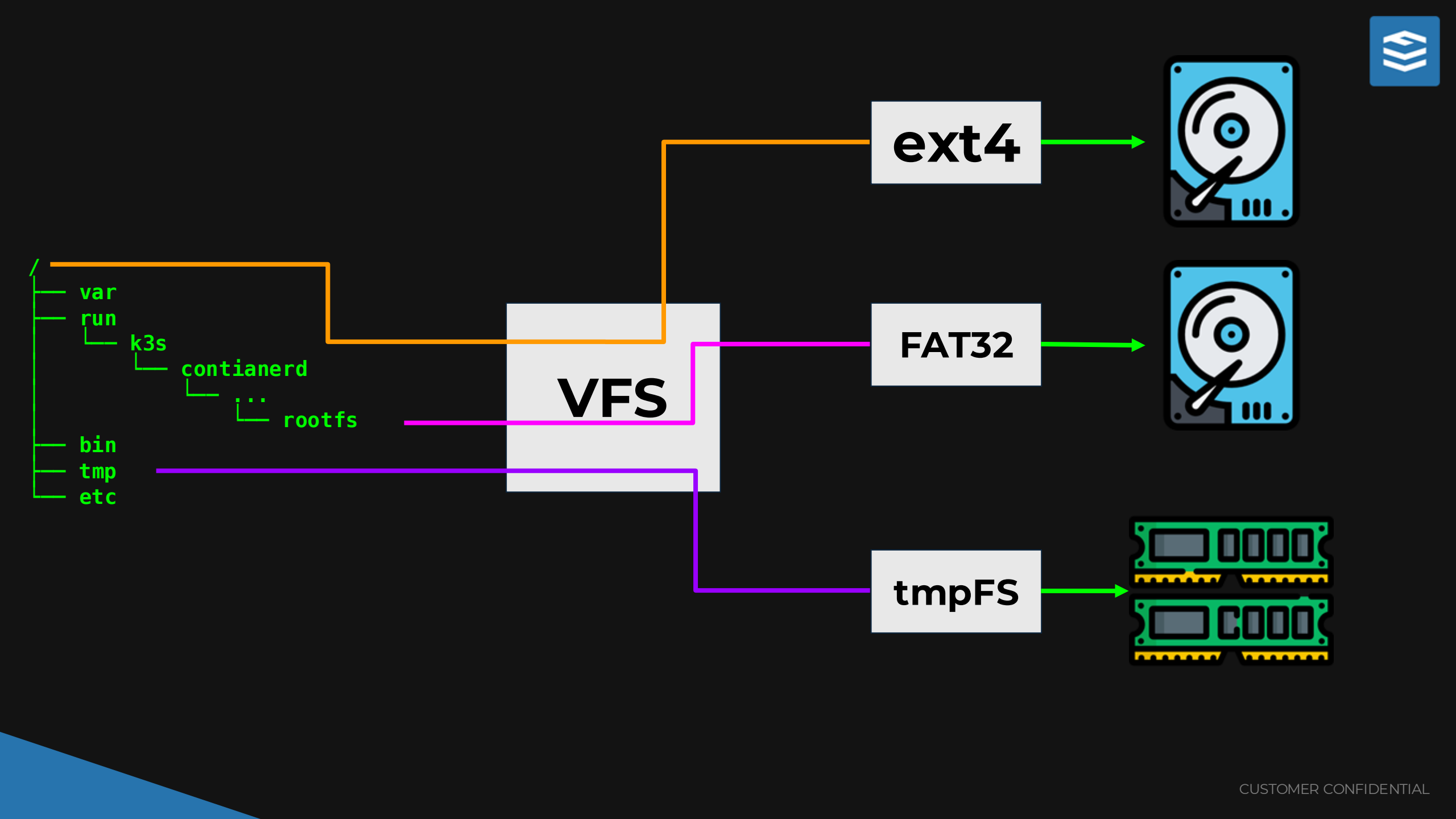
VFS

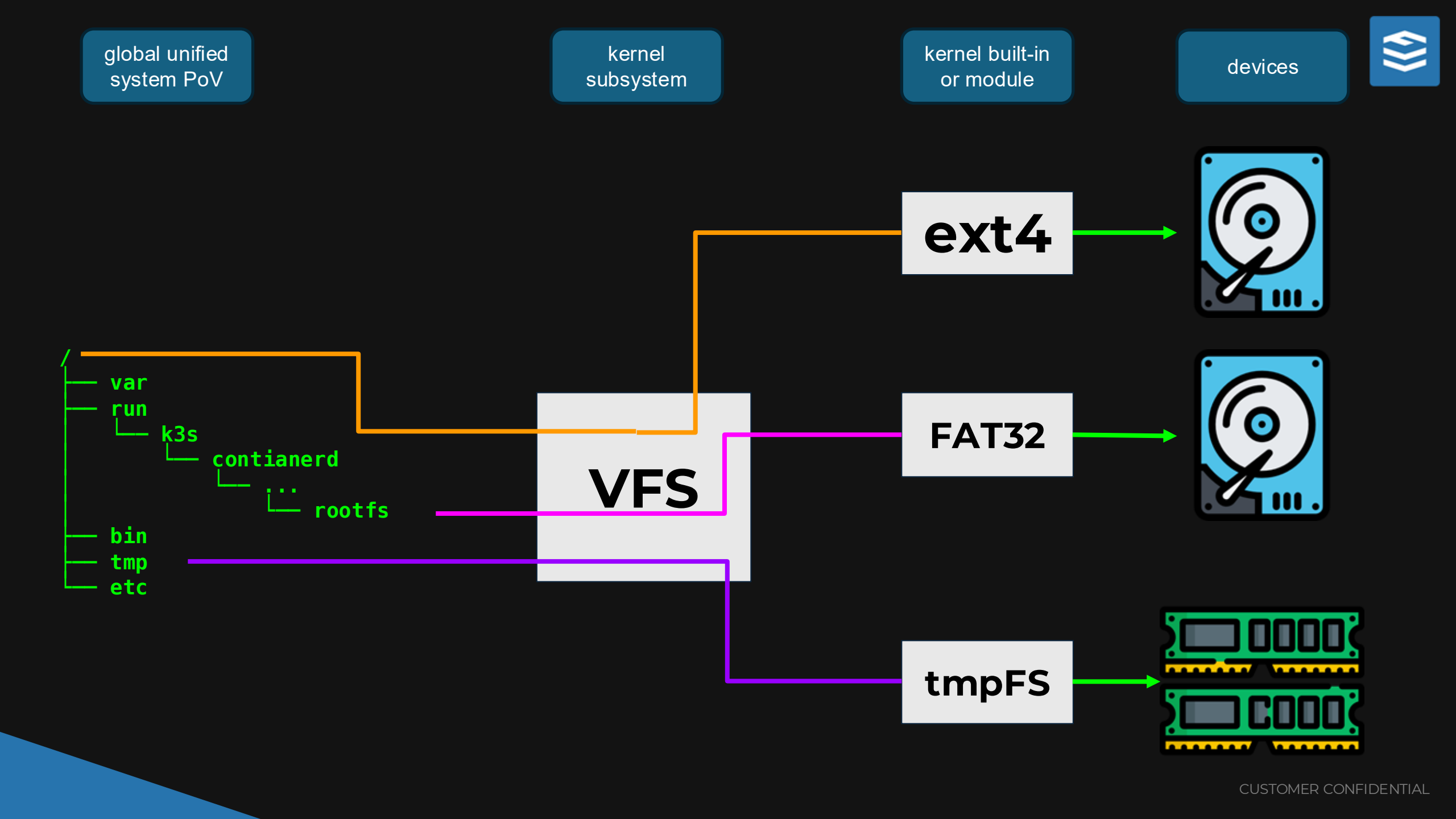
ext4

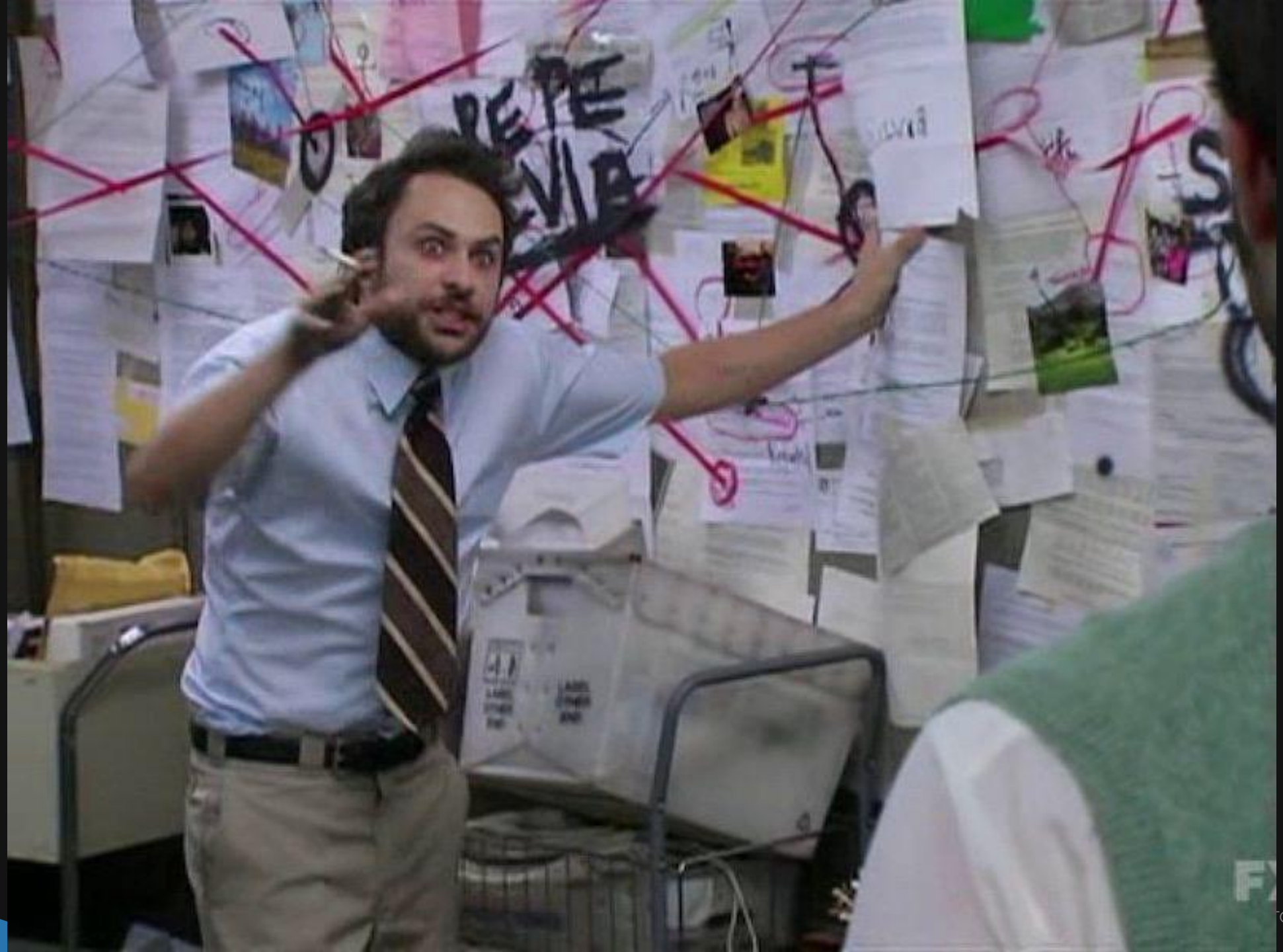
FAT32

tmpFS









root@dev01-node03:~# cat /proc/filesystems

```
nodev   sysfs
nodev   tmpfs
nodev   proc
nodev   cgroup
nodev   cgroup2
nodev   ramfs
        ext3
        ext4
        squashfs
        vfat
nodev   ecryptfs
        fuseblk
nodev   fuse
nodev   fusectl
nodev   efivarfs
        btrfs
nodev   overlay
nodev   nfs
nodev   nfs4
```

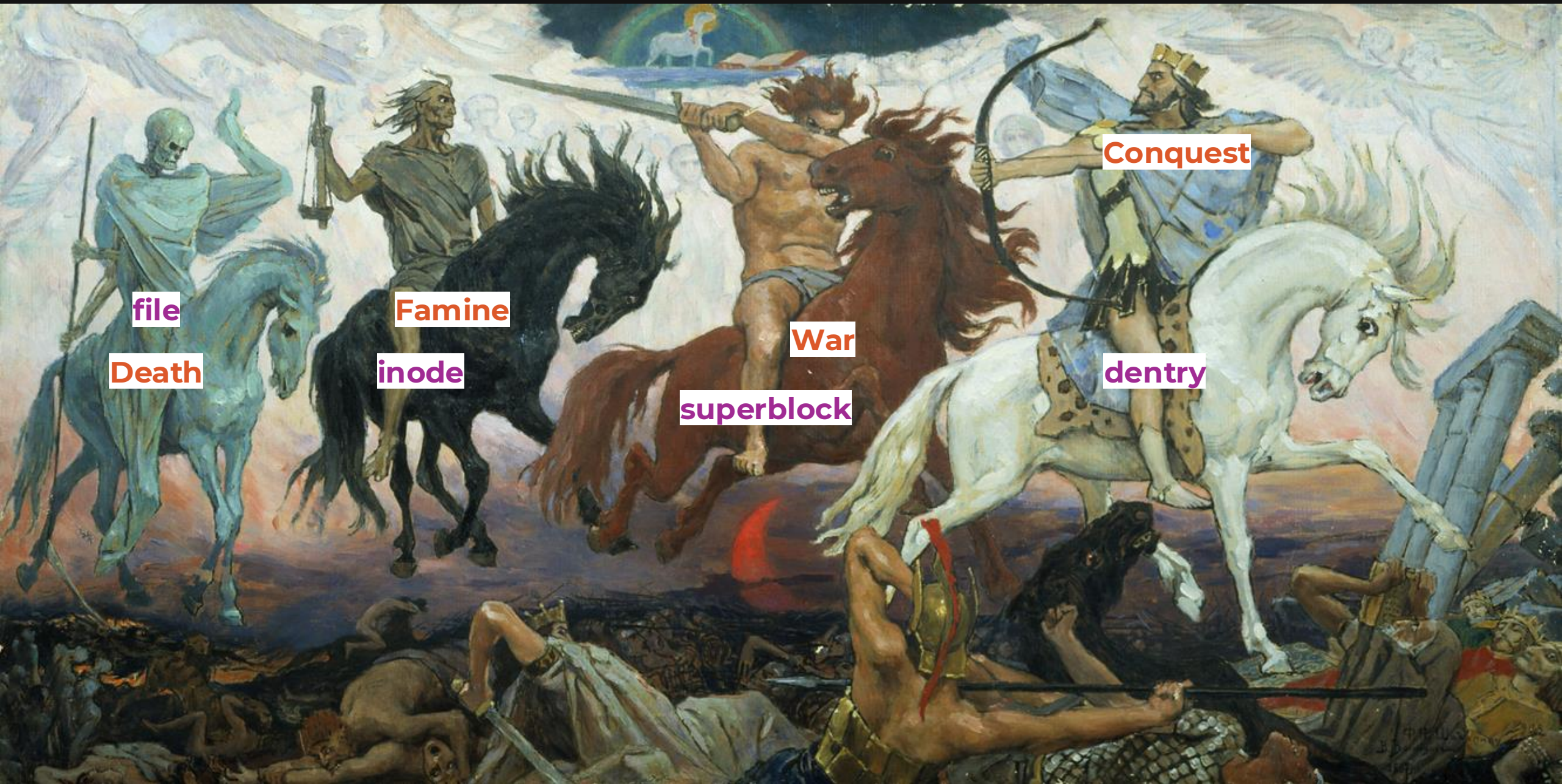
implementing a Filesystem from scratch
→ C code
Metadata – SUPER – INODE - FILE

```
lookup()
open()
read()
write()
stat()
unlink()
rename()
```

VFS

```c
#include <linux/fs.h>

extern int register_filesystem(struct file_system_type *);
extern int unregister_filesystem(struct file_system_type *);

struct file_system_type {
        const char *name;
        const struct fs_parameter_spec *parameters;
        struct dentry *(*mount) (struct file_system_type *, int, const char *, void *);
        // ...
};

struct super_operations {
        struct inode *(*alloc_inode)(struct super_block *sb);
        void (*destroy_inode)(struct inode *);
        int (*write_inode) (struct inode *, struct writeback_control *wbc);
        // ...
};

struct inode_operations {
        struct dentry * (*lookup) (struct inode *,struct dentry *, unsigned int);
        struct dentry *(*mkdir) (struct mnt_idmap *, struct inode *,struct dentry *,umode_t);
        int (*rmdir) (struct inode *,struct dentry *);
        // ...
};

struct file_operations {
        int (*open) (struct inode *, struct file *);
        ssize_t (*read_iter) (struct kiocb *, struct iov_iter *);
        ssize_t (*write_iter) (struct kiocb *, struct iov_iter *);
        // ...
};
```

Four Horsemen of Linux Filesystems



VFS (Virtual File System [Switch])

- Kernel abstraction layer for all filesystems
- Provides common API (open, read, write, stat)
- Hides filesystem-specific details (ext4, XFS, NFS, etc.)

Superblock

- Describes a mounted filesystem instance
- Stores filesystem metadata (type, size, block size)
- One superblock per mounted filesystem

```
struct super_block {
        dev_t                   s_dev;          /* identifier */
        struct file_system_type s_type;         /* filesystem type */
        struct super_operations s_op;           /* superblock methods */
        struct dentry           *s_root;        /* directory mount point */
        unsigned long           s_flags;        /* mount flags */
        struct block_device     *s_bdev;        /* associated block device */
        unsigned long           s_magic;        /* filesystem's magic number */
        void                    *s_fs_info;     /* filesystem-specific info */
        unsigned long           s_blocksize;    /* block size in bytes */
        unsigned long long      s_maxbytes;     /* max file size */
        unsigned char           s_dirt;         /* dirty flag */
        // ...
};
```

VFS (Virtual File System [Switch])

- Kernel abstraction layer for all filesystems
- Provides common API (open, read, write, stat)
- Hides filesystem-specific details (ext4, XFS, NFS, etc.)

inode

- Represents a file's metadata
- Stores permissions, owner, size, timestamps, data block pointers
- Does NOT store filename

```
struct inode {
        struct hlist_node       i_hash;         /* hash list */
        struct list_head        i_list;         /* list of inodes */
        struct list_head        i_dentry;       /* list of dentries */
        unsigned long           i_ino;          /* inode number */
        atomic_t                i_count;        /* reference counter */
        umode_t                 i_mode;         /* access permissions */
        unsigned int            i_nlink;        /* number of hard links */
        uid_t                   i_uid;          /* user id of owner */
        gid_t                   i_gid;          /* group id of owner */
        kdev_t                  i_rdev;         /* real device node */
        loff_t                  i_size;         /* file size in bytes */
        struct timespec         i_atime;        /* last access time */
        struct timespec         i_mtime;        /* last modify time */
        struct timespec         i_ctime;        /* last change time */
        struct inode_operations *i_op;          /* inode ops table */
        struct file_operations  *i_fop;         /* default inode ops */
        struct super_block      *i_sb;          /* associated superblock */
        struct file_lock        *i_flock;       /* file lock list */
        unsigned char           i_sock;         /* is this a socket? */
        atomic_t                i_writecount;   /* count of writers */
        void                    *i_security;    /* security module */
        __u32                   i_generation;   /* inode version number */
        // ...
};
```

VFS (Virtual File System [Switch])

- Kernel abstraction layer for all filesystems
- Provides common API (open, read, write, stat)
- Hides filesystem-specific details (ext4, XFS, NFS, etc.)

dentry

- Maps filename → inode
- Caches path lookups for performance
- Exists in memory (can be transient)

```
struct dentry {
        struct qstr             d_name;       /* dentry name */
        struct inode            *d_inode;     /* associated inode */
        struct dentry_operations *d_op;       /* dentry operations table */
        struct list_head        d_child;      /* list of dentries within */
        struct list_head        d_subdirs;    /* subdirectories */
        struct list_head        d_alias;      /* list of alias inodes */
        struct dentry           *d_parent;    /* dentry object of parent */
        struct super_block      *d_sb;        /* superblock of file */
        int                     d_mounted;    /* is this a mount point? */
        atomic_t                d_count;      /* usage count */
        unsigned long           d_vfs_flags;  /* dentry cache flags */
        unsigned long           d_time;       /* revalidate time */
        // ...
};
```

VFS (Virtual File System [Switch])

- Kernel abstraction layer for all filesystems
- Provides common API (open, read, write, stat)
- Hides filesystem-specific details (ext4, XFS, NFS, etc.)

file

- Represents an open file (per process)
- Stores file offset, access mode, file operations
- Created per open() call

```
struct file {
        struct dentry           *f_dentry;      /* associated dentry object */
        struct file_operations  *f_op;          /* file operations table */
        atomic_t                f_count;        /* file object's usage count */
        unsigned int            f_uid;          /* user's UID */
        unsigned int            f_gid;          /* user's GID */
        mode_t                  f_mode;         /* file access mode */
        loff_t                  f_pos;          /* file offset (file pointer) */
        unsigned int            f_flags;        /* flags specified on open */
        // ..
};
```

VFS (Virtual File System [Switch])

- Kernel abstraction layer for all filesystems
- Provides common API (open, read, write, stat)
- Hides filesystem-specific details (ext4, XFS, NFS, etc.)

file

- Represents an open file (per process)
- Stores file offset, access mode, file operations
- Created per open() call

dentry

- Maps filename → inode
- Caches path lookups for performance
- Exists in memory (can be transient)

inode

- Represents a file's metadata
- Stores permissions, owner, size, timestamps, data block pointers
- Does NOT store filename

Superblock

- Describes a mounted filesystem instance
- Stores filesystem metadata (type, size, block size)
- One superblock per mounted filesystem

FS under the hood of containers

proc

- virtual filesystem: exposes live kernel and process state as files

- not stored on disk

- contents are generated on demand by the kernel.

- /proc/<pid>/.. Infos
  - cmdline - command line
  - exe - the executable
  - cwd - the current working directory
  - fd - open files
  - ns - process namespaces
  - cgroup - 🤷
  - root - root path location
  - mountinfo - mounts for this ns

proc

```
[root@localhost ~]# ls -l /proc/9166/
total 0
-r--r--r--.  1 root root 0 Feb  5 13:26 cgroup
-r--r--r--.  1 root root 0 Feb  5 13:26 cmdline
lrwxrwxrwx.  1 root root 0 Feb  5 13:26 cwd -> /mnt
lrwxrwxrwx.  1 root root 0 Feb  5 13:26 exe -> /usr/bin/bash
dr-x------.  2 root root 4 Feb  5 13:26 fd
-rw-r--r--.  1 root root 0 Feb  5 13:26 gid_map
-r--r--r--.  1 root root 0 Feb  5 13:26 limits
-r--r--r--.  1 root root 0 Feb  5 13:26 mountinfo
-r--------.  1 root root 0 Feb  5 13:26 mountstats
dr-xr-xr-x. 59 root root 0 Feb  5 13:26 net
dr-x--x--x.  2 root root 0 Feb  5 13:26 ns
lrwxrwxrwx.  1 root root 0 Feb  5 13:26 root -> /
-r--r--r--.  1 root root 0 Feb  5 13:26 status

...
```

proc

```
[root@localhost 9166]# cat mountinfo
582 475 0:53 / / rw,relatime - overlay overlay \
    rw,context="system_u:object_r:container_file_t:s0:c180,c646",\
    lowerdir=   /var/lib/containers/storage/overlay/l/FZIZLYG7CUE443WZ37FFNYZ7OY:
                /var/lib/containers/storage/overlay/l/JROXE7AJ7LEYMN3YO7VQ7TVT4O:
                /var/lib/containers/storage/overlay/l/HSIO5SOQZQGVW3JC5CATXEQ7WV:
                /var/lib/containers/storage/overlay/l/I3CH5KJL4EYGFMURLV4Z2O7ITD:
                /var/lib/containers/storage/overlay/l/FXNULZ4M4XLWUXX7CH2Z4SLJOF:
                /var/lib/containers/storage/overlay/l/MK52TPA2GWR4WUG7H2K5S4GVKL:
                /var/lib/containers/storage/overlay/l/VAV4PZVY2K2YMVOLEJIVJ7IBZH,
    upperdir=   /var/lib/containers/storage/overlay/9a233a9134...b028/diff,
    workdir=    /var/lib/containers/storage/overlay/9a233a9134...b028/work,
    redirect_dir=on,uuid=on,metacopy=on
583 582 0:64 / /proc rw,nosuid,nodev,noexec,relatime - proc proc rw
585 582 0:66 / /sys ro,nosuid,nodev,noexec,relatime - sysfs sysfs rw,seclabel
589 582 0:28 /containers/storage/overlay-containers/f0a6a39...76d39e3/userdata/hosts
             /etc/hosts
             rw
             -
             tmpfs tmpfs
             rw,seclabel,size=1122816k,nr_inodes=819200,mode=755,inode64
```

overlay FS

image config JSON

```
{
  "schemaVersion": 2,
  "mediaType": "application/vnd.oci.image.manifest.v1+json",
  "config": {
    "mediaType": "application/vnd.oci.image.config.v1+json",
    "digest": "sha256:e05ea06e638a296341e6..",
    "size": 3034
  },
  "layers": [
    {
      "mediaType": "application/vnd.oci.image.layer.v1.tar+gzip",
      "digest": "sha256:e693fa4edc2d351b4f..",
      "size": 32108244
    },
    {
      "mediaType": "application/vnd.oci.image.layer.v1.tar+gzip",
      "digest": "sha256:bebfa6dac0c3f0abf0..",
      "size": 1131
    },
    {
      "mediaType": "application/vnd.oci.image.layer.v1.tar+gzip",
      "digest": "sha256:5202592916112a8945..",
      "size": 853
    }
}
```

extracted image

```
$ ls -lR

drwxr-xr-x@   - drackthor 2026 Feb 03 08:23 blobs/
.rw-r--r--@ 272 drackthor 2026 Feb 03 08:23 index.json
.rw-r--r--@  30 drackthor 2026 Feb 03 08:23 oci-layout

./blobs:
drwxr-xr-x@ - drackthor 2026 Feb 03 08:23 sha256/

./blobs/sha256:
.rw-r--r--@  144 drackthor 2026 Feb 03 08:23    51e5d9a1dc408635c83a05275ddaa29bcfc6...
.rw-r--r--@ 1.3k drackthor 2026 Feb 03 08:23    550faad68d1725569c1ebb50a8ef03664f73...
.rw-r--r--@  853 drackthor 2026 Feb 03 08:23    5202592916112a894573492ac1d98263db6b...
.rw-r--r--@   34 drackthor 2026 Feb 03 08:23    bd9ddc54bea929a22b334e73e026d4136e5b...
.rw-r--r--@ 1.1k drackthor 2026 Feb 03 08:23    bebfa6dac0c3f0abf072bbf9f4b202bf2379...
.rw-r--r--@  533 drackthor 2026 Feb 03 08:23    cf93a83fa68ea9bee0fcd713aa68960760fe...
.rw-r--r--@ 3.0k drackthor 2026 Feb 03 08:23    e05ea06e638a296341e60cf6f21c8e283961...
.rw-r--r--@  32M drackthor 2026 Feb 03 08:23    e693fa4edc2d351b4f9847101ffd623e0826...
.rw-r--r--@  14M drackthor 2026 Feb 03 08:23    fb81a21d44f04e54b33454ab1a397bdfe28a...
```

overlay FS

```
$ mount | grep overlay
overlay on /var/lib/containers/storage/overlay/070b286e6d77f5d37142527b882039afc…/merged
\
    type overlay (
        rw,nodev,relatime,
        context="system_u:object_r:container_file_t:s0:c661,c962",
        lowerdir=\
            /var/lib/containers/storage/overlay/l/FZIZLYG7CUE443WZ37FFNYZ7OY:
            /var/lib/containers/storage/overlay/l/JROXE7AJ7LEYMN3YO7VQ7TVT4O:
            /var/lib/containers/storage/overlay/l/HSIO5SOQZQGVW3JC5CATXEQ7WV:
            /var/lib/containers/storage/overlay/l/I3CH5KJL4EYGFMURLV4Z2O7ITD:
            /var/lib/containers/storage/overlay/l/FXNULZ4M4XLWUXX7CH2Z4SLJOF:
            /var/lib/containers/storage/overlay/l/MK52TPA2GWR4WUG7H2K5S4GVKL:
            /var/lib/containers/storageoverlay/l/VAV4PZVY2K2YMVOLEJIVJ7IBZH,
        upperdir=/var/lib/containers/storage/overlay/070b286e6d77f5d37142527b8820…/diff,
        workdir=/var/lib/containers/storage/overlay/070b286e6d77f5d37142527b88203…/work,
        redirect_dir=on,uuid=on,metacopy=on
    )
```

shm/tmpfs

- tmpfs provides fast, ephemeral, namespace-isolated storage for IPC and runtime state

- backed by RAM but eligible for swap

- Common uses:
  - /dev/shm → shared memory & IPC
  - /etc/hosts
  - /etc/resolv.conf
  - /run
  - /tmp

sysfs

- virtual filesystem that exposes the kernel's internal object model
- files represent kernel objects and attributes
- reading/writing files = interacting with kernel state

- Rarely given access to containers
- Usually RO access, except special CSI, CNI,.. containers
  - Sensitive subtrees are masked

sysfs

```
# HOST PoV -> masking sysfs in container with empty tmpfs
[root@localhost 2679]# cat mountinfo | grep /sys
570 567 0:63 / /sys ro,nosuid,nodev,noexec,relatime - sysfs sysfs rw,seclabel
596 570 0:67 / /sys/firmware ro,relatime - tmpfs tmpfs rw,context="..",size=0k,inode64
597 570 0:68 / /sys/fs/selinux ro,relatime - tmpfs tmpfs rw,context="..",size=0k,inode64

[root@localhost 2679]# ls -l /sys/firmware/
total 0
drwxr-xr-x. 4 root root   0 Feb  8 10:44 acpi
drwxr-xr-x. 2 root root   0 Feb  8 11:07 devicetree
drwxr-xr-x. 4 root root   0 Feb  8 10:44 dmi
drwxr-xr-x. 4 root root   0 Feb  8 10:44 efi
-r--------. 1 root root 830 Feb  8 11:07 fdt

# container PoV

root@895e9b1a021c:/data# ls -l /sys/
total 0
...
drwxr-xr-x.  15 root root  0 Feb  8 10:05 devices
drwxrwxrwt.   2 root root 40 Feb  8 10:05 firmware
drwxr-xr-x.  12 root root  0 Feb  8 10:05 fs
drwxr-xr-x.  16 root root  0 Feb  8 10:05 kernel
drwxr-xr-x. 208 root root  0 Feb  8 10:05 module
drwxr-xr-x.   3 root root  0 Feb  8 10:05 power
root@895e9b1a021c:/data# ls -l /sys/firmware/
total 0
root@895e9b1a021c:/data#
```

cgroup(fs)

- kernel mechanism that hierarchically groups processes
- account for and enforce CPU, memory, IO, and PID limits
- use controllers for resources under mgmt
- /proc/<pid>/cgroup

cgroup(fs)

```
[root@localhost]# mount | grep cgroup
cgroup2 on /sys/fs/cgroup type cgroup2 \
(rw,nosuid,nodev,noexec,relatime,seclabel,nsdelegate,memory_recursiveprot)
```

```
# /proc/<pid>
[root@localhost 9166]# cat cgroup
0::/machine.slice/libpod-f0a6a3..39e3.scope/container
```

```
[root@localhost]# cat /sys/fs/cgroup/machine.slice/libpod-f0a6a3..39e3-
.../cgroup.controllers
```

```
cpuset cpu io memory pids misc
```

cgroup(fs)

```
[root@localhost]# ls -l /sys/fs/cgroup/machine.slice/libpod-f0a6a3..39e3.scope/
total 0
-r--r--r--. 1 root root 0 Feb  5 13:26 cgroup.controllers
--w-------. 1 root root 0 Feb  5 13:26 cgroup.kill
-rw-r--r--. 1 root root 0 Feb  5 13:26 cgroup.max.depth
-rw-r--r--. 1 root root 0 Feb  5 13:26 cgroup.max.descendants
-rw-r--r--. 1 root root 0 Feb  5 13:26 cgroup.procs
-r--r--r--. 1 root root 0 Feb  5 13:26 cgroup.stat
-rw-r--r--. 1 root root 0 Feb  5 13:26 cgroup.type
drwxr-xr-x. 2 root root 0 Feb  5 13:26 container
-rw-r--r--. 1 root root 0 Feb  5 13:26 cpu.max
-rw-r--r--. 1 root root 0 Feb  5 13:26 cpuset.cpus
-rw-r--r--. 1 root root 0 Feb  5 13:26 io.max
-r--r--r--. 1 root root 0 Feb  5 13:26 memory.current
-rw-r--r--. 1 root root 0 Feb  5 13:26 memory.max
-rw-r--r--. 1 root root 0 Feb  5 13:26 memory.min
-rw-r--r--. 1 root root 0 Feb  5 13:26 memory.oom.group
-r--r--r--. 1 root root 0 Feb  5 13:26 memory.swap.current
-rw-r--r--. 1 root root 0 Feb  5 13:26 memory.swap.max
-r--r--r--. 1 root root 0 Feb  5 13:26 pids.current
-rw-r--r--. 1 root root 0 Feb  5 13:26 pids.max
```

cgroup(fs)

```
root@localhost libpod-f0a6a3..39e3.scope]# cat memory.max
268435456 # bytes

[root@localhost libpod-f0a6a3..39e3.scope]# cat cpu.max
100000 100000 # $MAX $PERIOD

[root@localhost libpod-f0a6a3..39e3.scope]# cat pids.max
2048 # max pids in cgroup

[root@localhost libpod-f0a6a3..39e3.scope]# cat pids.current
6 # current pids in cgroup
```

Conclusion

```
UID          PID    PPID  C STIME TTY          TIME CMD
1001           1       0  0 Jan21 ?        01:22:21 redis-server
*:6379
1001     2297276       0  0 12:53 pts/0    00:00:00 sh
1001     2297338 2297276  0 12:53 pts/0    00:00:00 ps -eaf
```

```
/
├── var
├── bin
├── opt
└── etc
```

```
$ hostname
container-1
```

Users

root:x:0:0:root:/root:/bin/bash
app:x:1006:1006:app:/app:/bin/bash

CPU | Memory

Filesystems
under the hood

```
/
├── var
├── run
│   └── k3s
│       └── contianerd
│           └── ...
│               └── rootfs
├── bin
├── opt
└── etc
```

PSTREE

Users

root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
.....
drad:x:1006:1006:Daniel Drack:/home/drad:/bin/bash
grep:x:1007:1007:Paul Greuter:/home/grep:/bin/bash

network
IPC
users
mount
cgroups
PID
UTS

CPU | Memory

FULL STACKS

WeAreDevelopers

Linux (Pseudo) Filesystems

The Hidden Backbone of Cloud Native

Daniel Drack

@DrackThor

CLOUD NATIVE COMPUTING FOUNDATION SILVER MEMBER

kubernetes CERTIFIED

Diamond Partner

HashiCorp SPECIALIZED

cisco Partner

Red Hat

snyk

GitLab SELECT CHANNEL PARTNER

VEEAM Value-Added Reseller Registered

aws PARTNER

splunk> a cisco company

EXOSCALE Partner