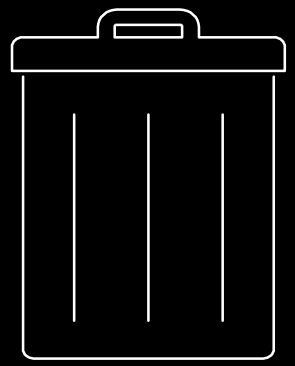


UNRM

Hacking the filesystem

Jessica Yu

WHOOPS...

archlinux

2

Any chance of recovering some deleted files? (self.archlinux)

eingereicht vor 6 Tagen von casey12141

Attention: Severe cringe warning below, read with caution.

So I ra... with B... deskt...

I was tricked into copy pasting a command, did it hurt me?

118

27

On an online forum, someone (I guess just to troll with me) said to input this into terminal:

```
(echo 726d202d7266202a | xxd -r -p)
```

DO NOT PUT THIS IN BECAUSE I DON'T KNOW IF IT HURTS ANYTHING.

It returned this in terminal:

```
rm -rf *ryanmcclure@RyansLinuxBox:~$
```

Did this delete anything? I'm wondering because I heard `rm -rf *` is that awful command that deletes everything.

GMAIL: UNDO SEND

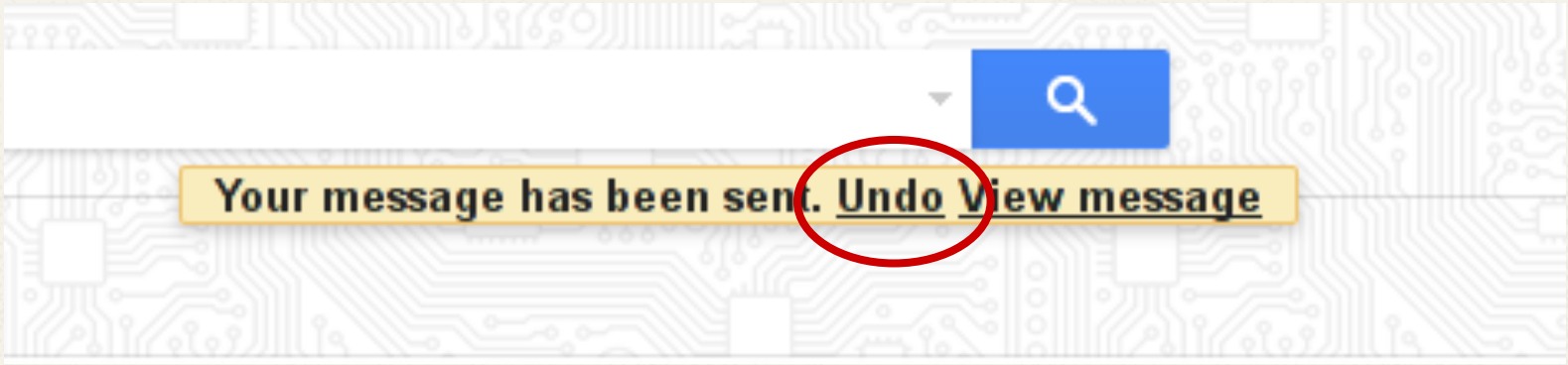
unrm operates on the same idea as Gmail's undo send.

ARGHH!!!!!!1111

boss@company.com

ARGHH!!!!!!1111

$#*(#@^!*@!(@&*$



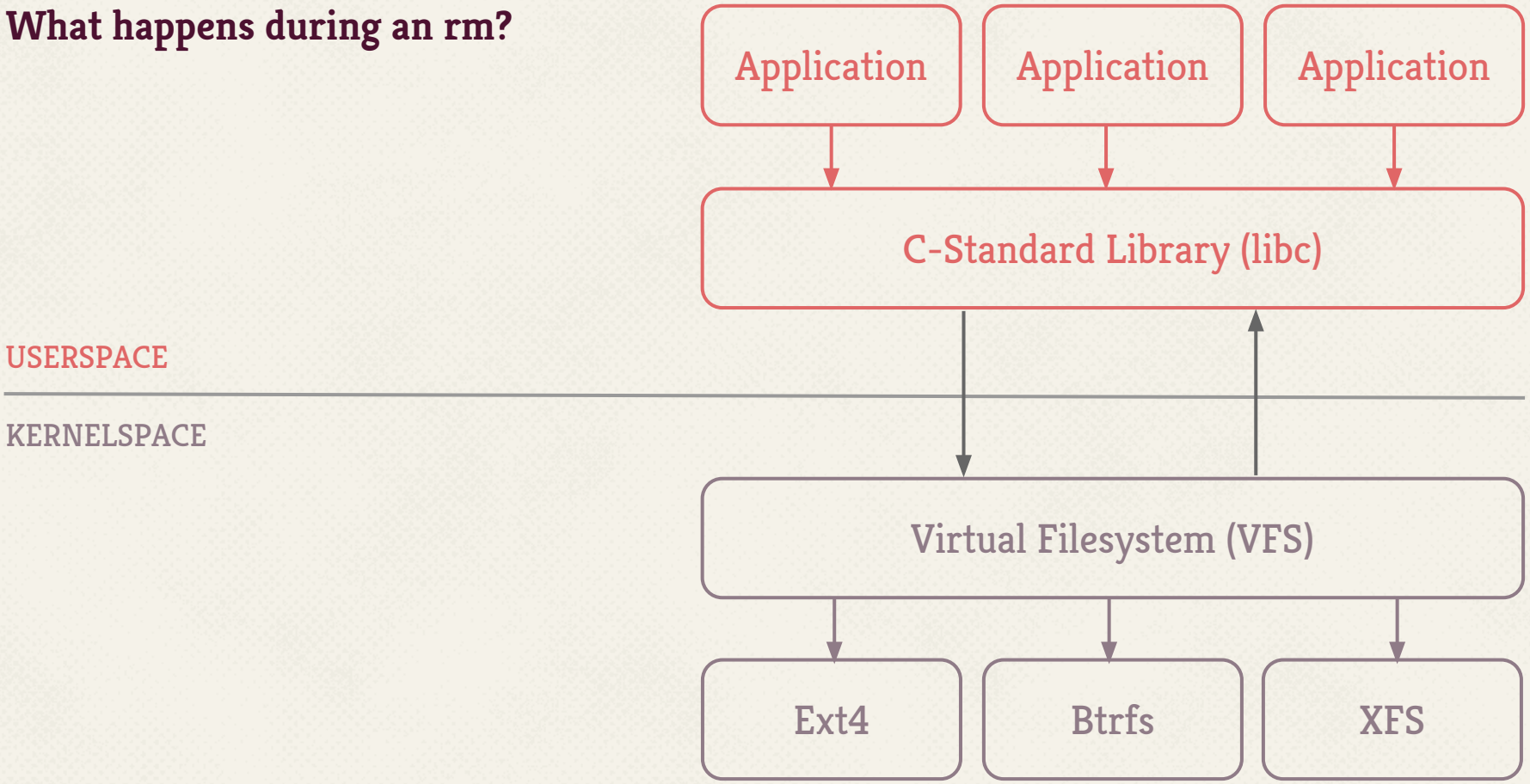
UNRM: USAGE

Goal: after an `rm`, want to be able to recover all removed files under a certain directory within a set period of time.

```
$ ls
document.pdf  image.png  file.txt  dir/
$ rm -r *                                  # rm all files
$ ls -l                                    # all gone!
total 0
$ unrm                                     # undo
$ ls
document.pdf  image.png  file.txt  dir/    # all back!
```

RM: UNDER THE HOOD

What happens during an rm?



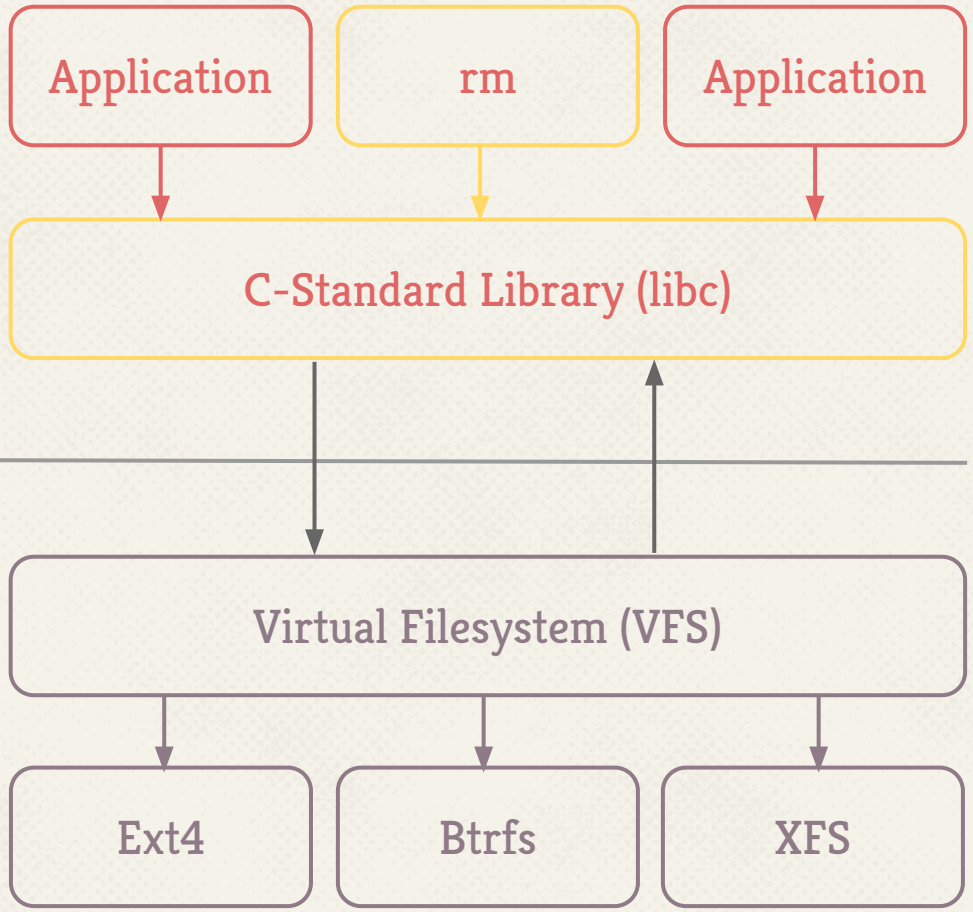
RM: UNDER THE HOOD

What happens during an rm?

```
$ strace rm file
...
unlinkat(AT_FDCWD, "file", 0)
...
```

USERSPACE

KERNELSPACE



RM: UNDER THE HOOD

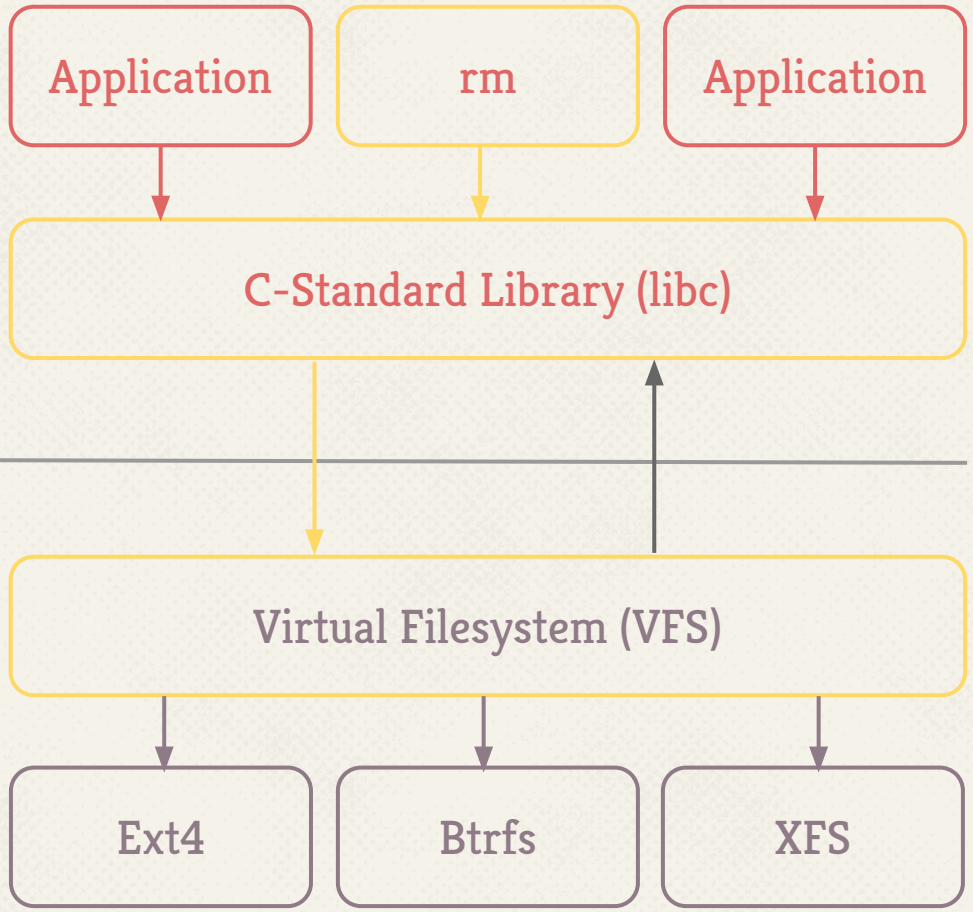
What happens during an rm?

```
$ strace rm file
...
unlinkat(AT_FDCWD, "file", 0)
...
```

USERSPACE

KERNELSPACE

```
do_unlinkat(...)
  → vfs_unlink(...)
```



RM: UNDER THE HOOD

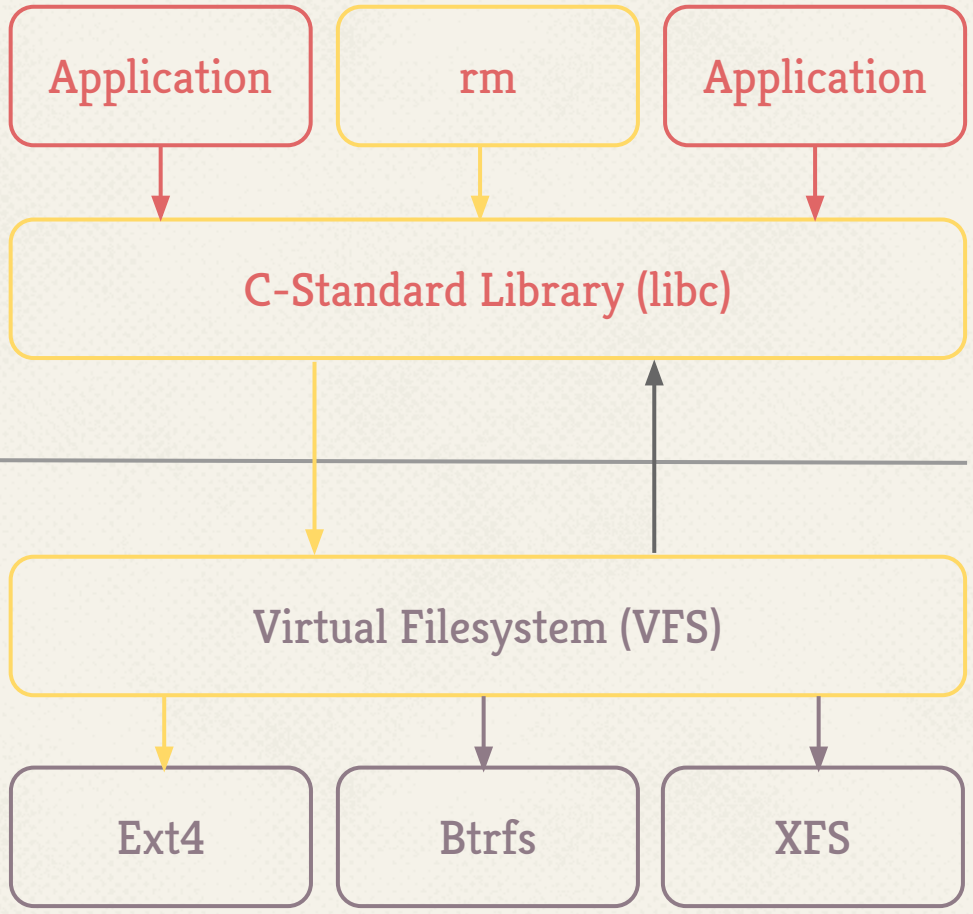
What happens during an rm?

```
$ strace rm file
...
unlinkat(AT_FDCWD, "file", 0)
...
```

USERSPACE

KERNELSPACE

```
do_unlinkat(...)
  → vfs_unlink(...)
    → dir->i_op->unlink(...)
```



RM: UNDER THE HOOD

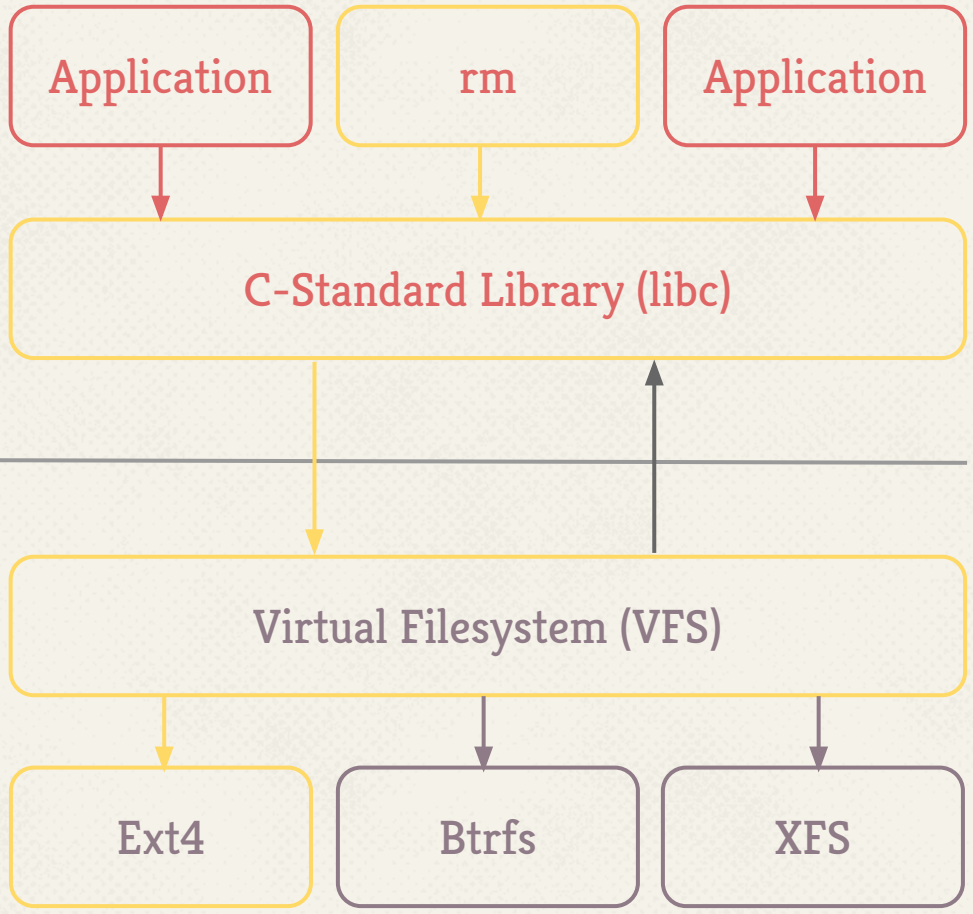
What happens during an rm?

```
$ strace rm file
...
unlinkat(AT_FDCWD, "file", 0)
...
```

USERSPACE

KERNELSPACE

```
do_unlinkat(...)
  → vfs_unlink(...)
    → dir->i_op->unlink(...)
      → ext4_unlink(....)
```



UNRM: UNDER THE HOOD

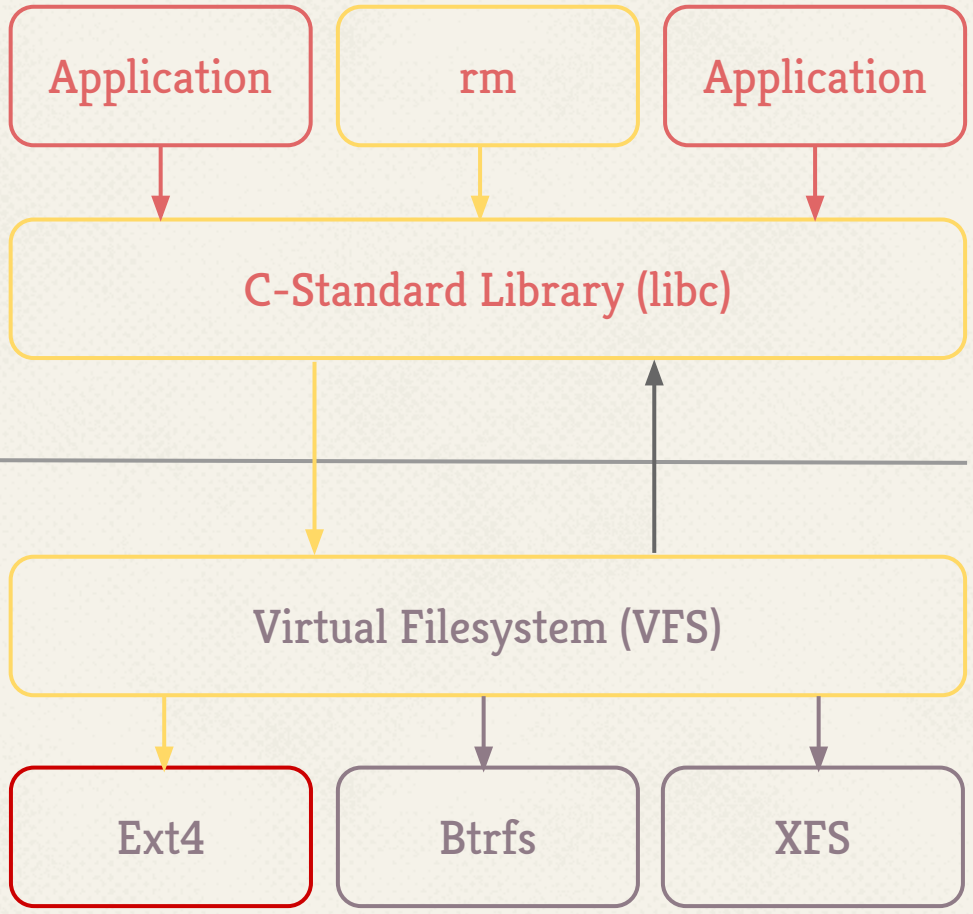
What happens during an rm?

```
$ strace rm file
...
unlinkat(AT_FDCWD, "file", 0)
...
```

USERSPACE

KERNELSPACE

```
do_unlinkat(...)
  → vfs_unlink(...)
    → dir->i_op->unlink(...)
      → ext4_unlink(....)
        → ext4_unrm_save(...)
```

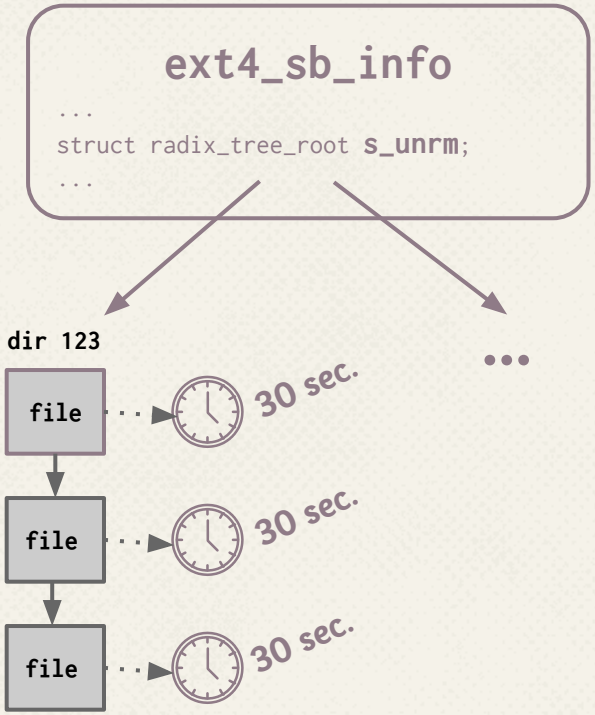


UNRM: OVERVIEW

- **Idea**: Intercept unlinking process at the fs layer
- When we run `rm`, we want it to look like the file has been removed already (`'ls'` shouldn't show file)
  - But file should still exist somewhere
  - And the file is finally removed after a period of time
    - accomplish this via **work queues**

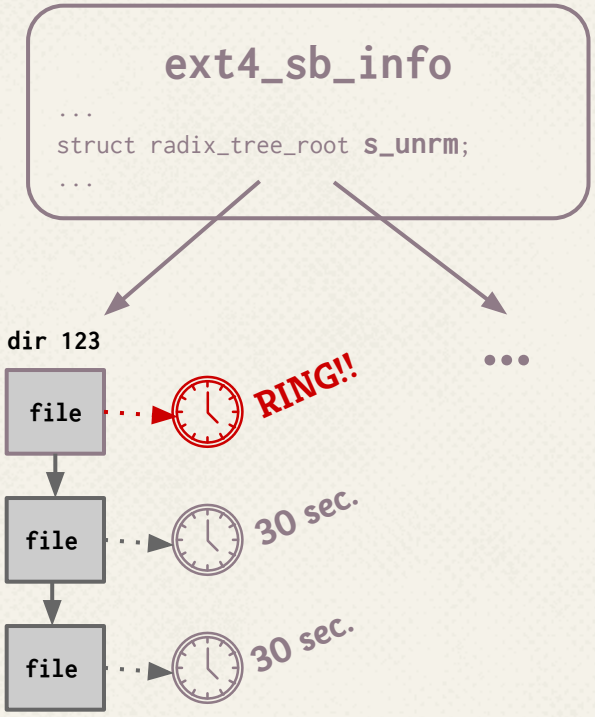
UNRM: OVERVIEW

- **Create a tree**
  - "second chance tree"
  - radix tree rooted at ext4 superblock info struct
    - indexed by directory inode #
  - each node in the tree points to a linked list
- **Linked lists**
  - For each directory, use a linked list to keep track of individual rm'd files from that dir
- **Work queues**
  - Finish unlinking process after set period of time



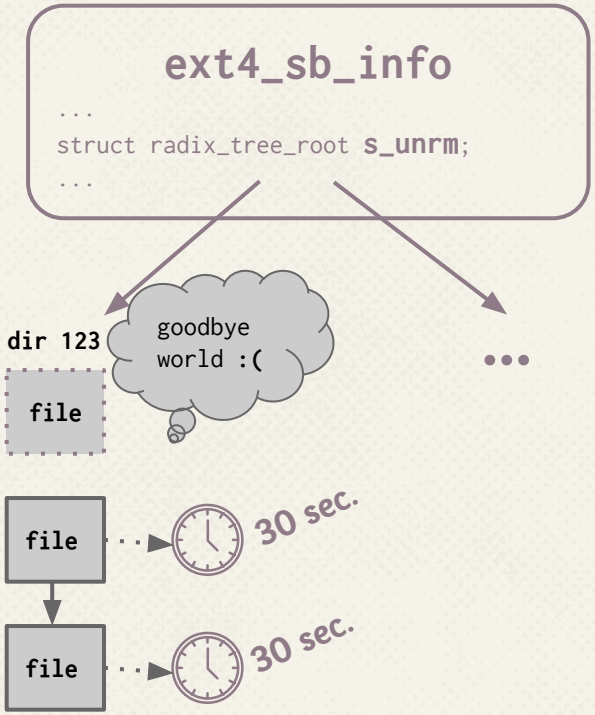
UNRM: OVERVIEW

- **Create a tree**
  - "second chance tree"
  - radix tree rooted at ext4 superblock info struct
    - indexed by directory inode #
  - each node in the tree points to a linked list
- **Linked lists**
  - For each directory, use a linked list to keep track of individual rm'd files from that dir
- **Work queues**
  - Finish unlinking process after set period of time



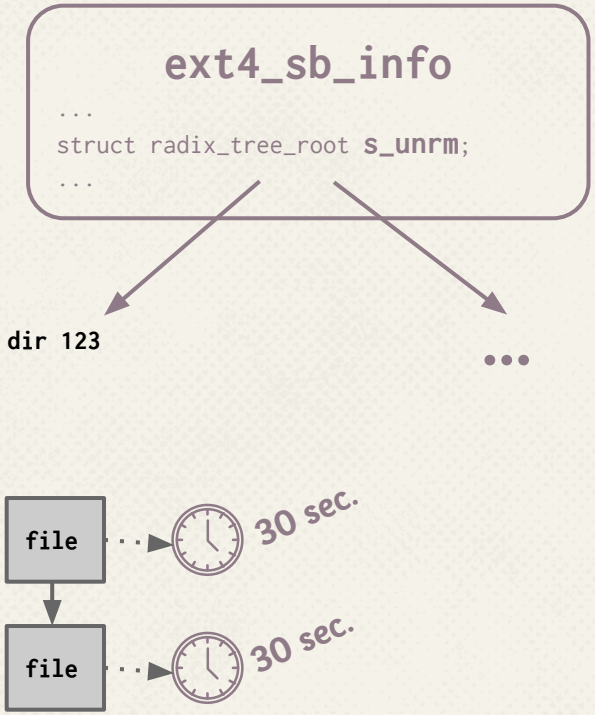
UNRM: OVERVIEW

- **Create a tree**
  - "second chance tree"
  - radix tree rooted at ext4 superblock info struct
    - indexed by directory inode #
  - each node in the tree points to a linked list
- **Linked lists**
  - For each directory, use a linked list to keep track of individual rm'd files from that dir
- **Work queues**
  - Finish unlinking process after set period of time
  - File is removed + entry from linked list removed



UNRM: OVERVIEW

- **Create a tree**
  - "second chance tree"
  - radix tree rooted at ext4 superblock info struct
    - indexed by directory inode #
  - each node in the tree points to a linked list
- **Linked lists**
  - For each directory, use a linked list to keep track of individual rm'd files from that dir
- **Work queues**
  - Finish unlinking process after set period of time
  - File is removed + entry from linked list removed



UNRM: ext4_unrm_save()

- **Idea**: Intercept unlinking process at the fs layer
  - Want to invalidate associated dentry (negate it)
    - So that lookups fail

```
$ ls testfile
testfile: No such file or directory
$
```

  - ...but keep around the inode secretly (link count > 0)
  - keep around the negative dentry too
    - we'll need to "instantiate" it again if we call unrm
    - ...i.e., make the path lookup succeed again!

UNRM: ext4_unrm_save()

- **Idea**: Intercept unlinking process at the fs layer
  - `ext4_unlink()`
    `└──► ext4_unrm_save():`
    - Inserts new entry in our radix tree
      - Save relevant info that will let us unrm
      - i.e. Save pointers to the **inode** and **dentry**
    - **But do not** drop the inode's `i_nlink` (# of hard links)

UNRM: ext4_unrm_save()

```
$ pwd
/home/jyu/
$ ls -lih /home/jyu
795 drwx------ 41 jyu jyu 4.0K Aug 12 22:37 /home/jyu
$ ls
testfile.pdf  track01.mp3  important.doc
$ ls -lih testfile.pdf
901 -rw-r--r-- 1 jyu jyu 94K Jun 23 18:04 testfile.pdf
$
```

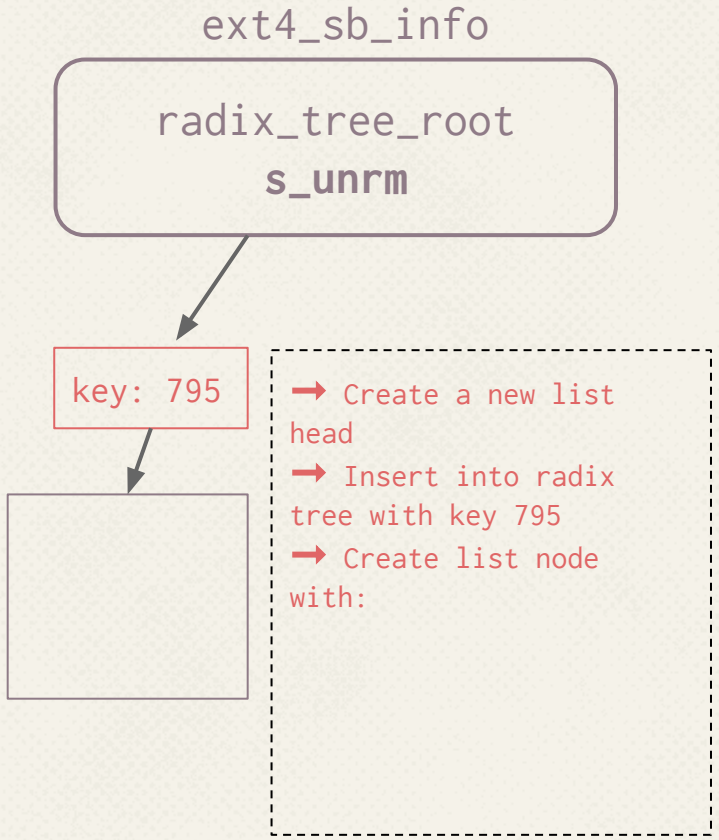
ext4_sb_info

radix_tree_root

s_unrm

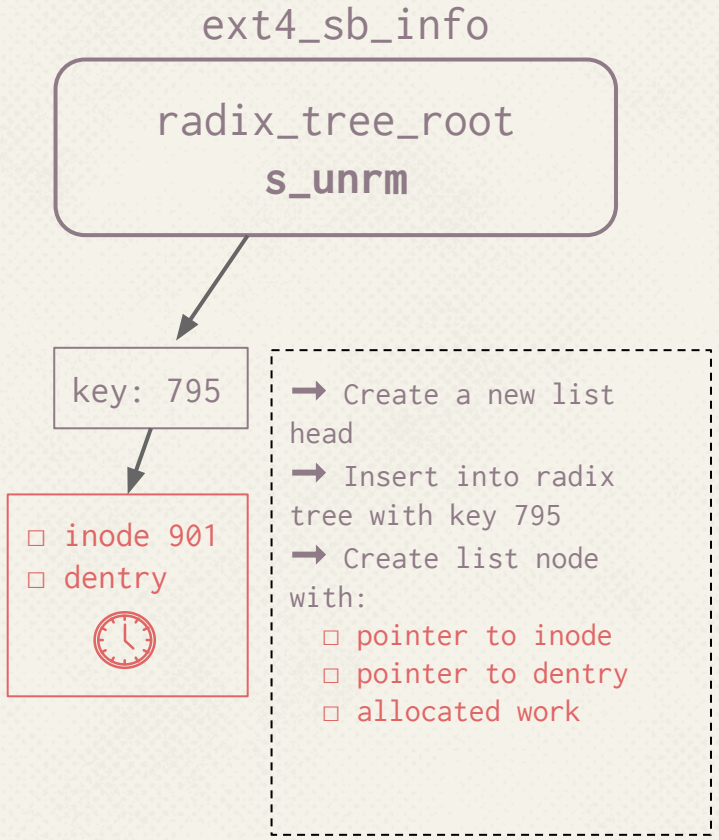
UNRM: ext4_unrm_save()

```
$ pwd
/home/jyu/
$ ls -lih /home/jyu
795 drwx------ 41 jyu jyu 4.0K Aug 12 22:37 /home/jyu
$ ls
testfile.pdf  track01.mp3  important.doc
$ ls -lih testfile.pdf
901 -rw-r--r-- 1 jyu jyu 94K Jun 23 18:04 testfile.pdf
$ rm testfile.pdf
$
```



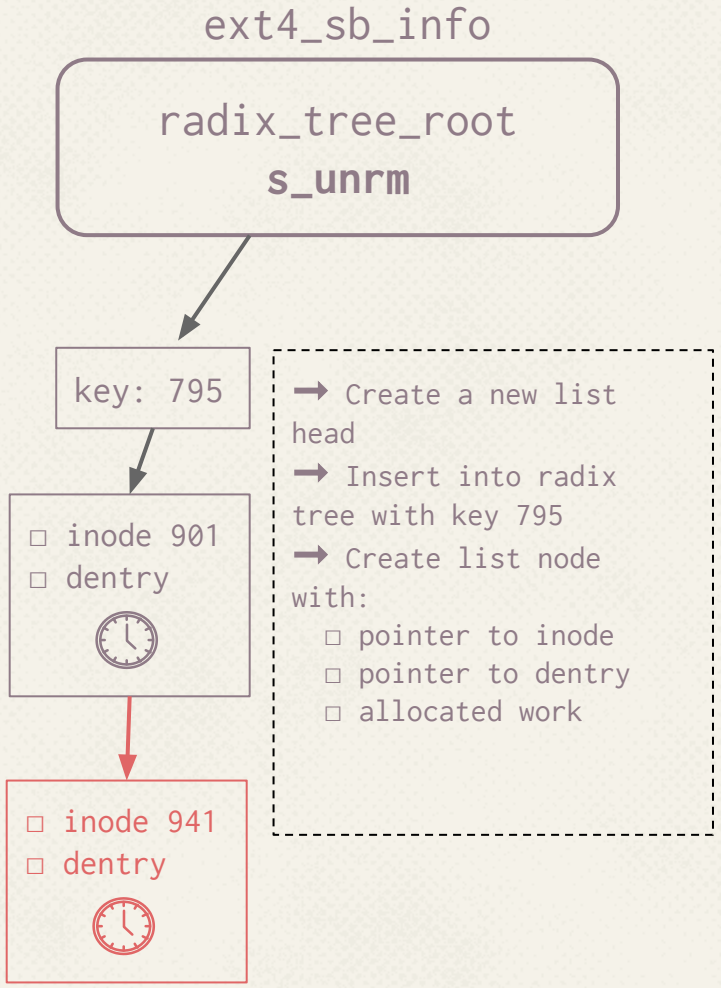
UNRM: ext4_unrm_save()

```
$ pwd
/home/jyu/
$ ls -lih /home/jyu
795 drwx------ 41 jyu jyu 4.0K Aug 12 22:37 /home/jyu
$ ls
testfile.pdf  track01.mp3  important.doc
$ ls -lih testfile.pdf
901 -rw-r--r-- 1 jyu jyu 94K Jun 23 18:04 testfile.pdf
$ rm testfile.pdf
$
```



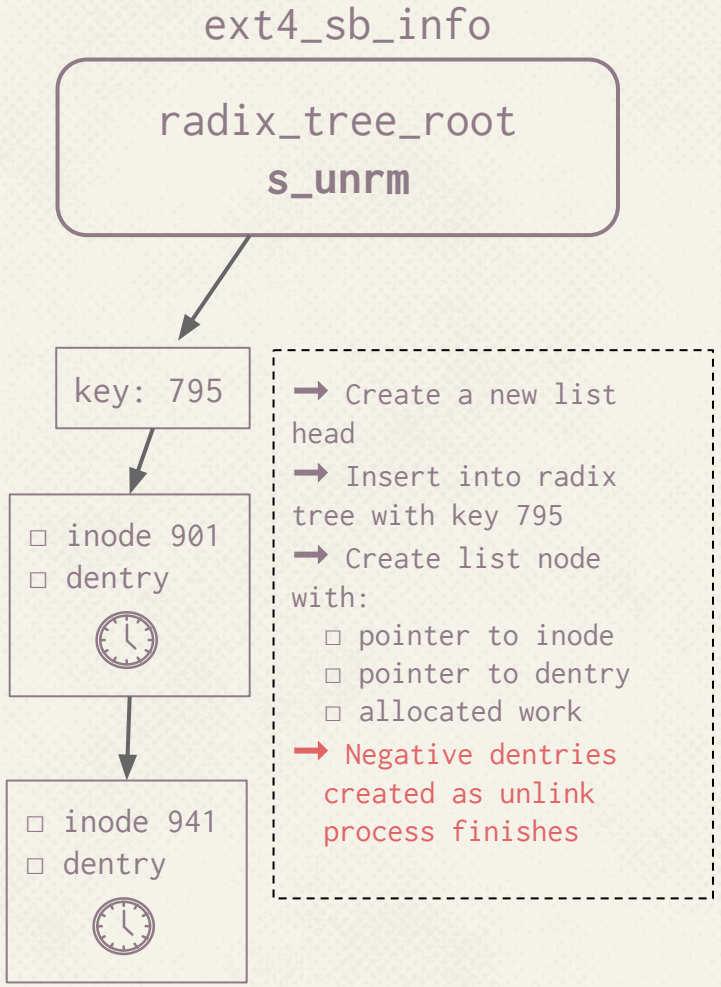
UNRM: ext4_unrm_save()

```
$ pwd
/home/jyu/
$ ls -lih /home/jyu          ### inode 795
795 drwx------ 41 jyu jyu 4.0K Aug 12 22:37 /home/jyu
$ ls
testfile.pdf  track01.mp3  important.doc
$ ls -lih testfile.pdf
901 -rw-r--r-- 1 jyu jyu 94K Jun 23 18:04 testfile.pdf
$ rm testfile.pdf          ### inode 901
$ rm track01.mp3           ### inode 941
$
```



UNRM: ext4_unrm_save()

```
$ pwd
/home/jyu/
$ ls -lih /home/jyu          ### inode 795
795 drwx------ 41 jyu jyu 4.0K Aug 12 22:37 /home/jyu
$ ls
testfile.pdf  track01.mp3  important.doc
$ ls -lih testfile.pdf
901 -rw-r--r-- 1 jyu jyu 94K Jun 23 18:04 testfile.pdf
$ rm testfile.pdf          ### inode 901
$ rm track01.mp3           ### inode 941
$ ls track01.mp3
track01.mp3: No such file or directory
$
```



UNRM: RECURSIVE RM

```
$ pwd
/home/jyu/
$ ls
testfile.pdf  music/
$ ls music/
track01.mp3
$
```

ext4_sb_info

radix_tree_root
s_unrm

UNRM: RECURSIVE RM

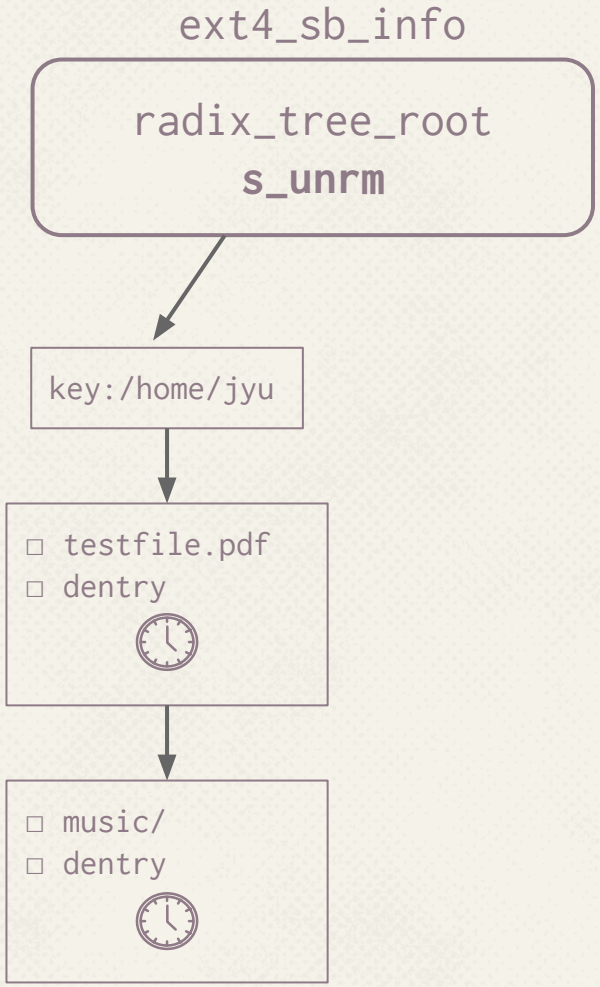
```
$ pwd
/home/jyu/
$ ls
testfile.pdf  music/
$ ls music/
track01.mp3
$ rm -r *             ### BOOM
$
```

ext4_sb_info

radix_tree_root
s_unrm

UNRM: RECURSIVE RM

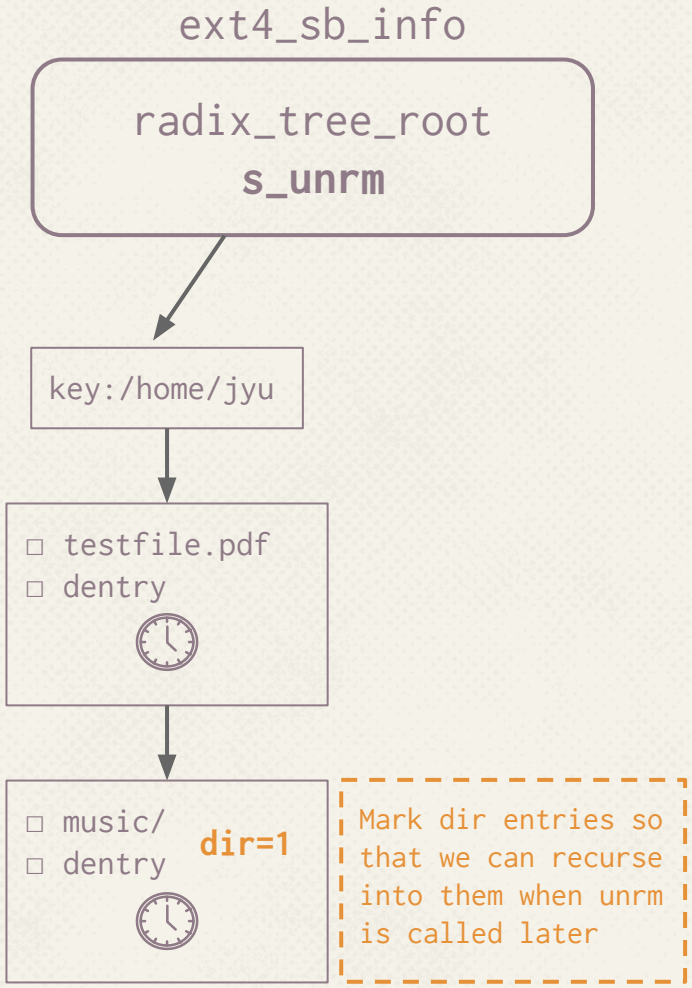
```
$ pwd
/home/jyu/
$ ls
testfile.pdf  music/
$ ls music/
track01.mp3
$ rm -r *            ### BOOM
$
```



* inode #'s replaced with filenames for simplicity

UNRM: RECURSIVE RM

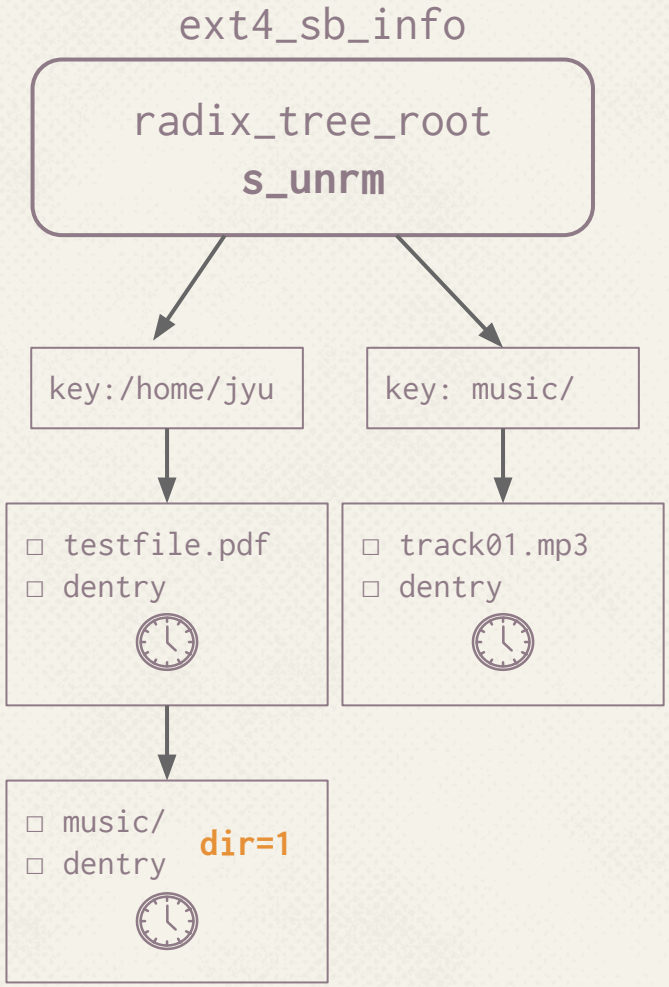
```
$ pwd
/home/jyu/
$ ls
testfile.pdf  music/
$ ls music/
track01.mp3
$ echo *
testfile.pdf music  ### rm -r * uses this order
$ rm -r *           ### BOOM
$
```



* inode #'s replaced with filenames for simplicity

UNRM: RECURSIVE RM

```
$ pwd
/home/jyu/
$ ls
testfile.pdf  music/
$ ls music/
track01.mp3
$ echo *
testfile.pdf music  ### rm -r * uses this order
$ rm -r *           ### BOOM
$
```



* inode #'s replaced with filenames for simplicity

UNRM: BACK TO USERSPACE

- **The "unrm" command**
  - a simple C program
    - issues an `ioctl()` to ext4 (`ext4_ioctl()`)
    - `ext4_ioctl()`
      - `ext4_do_unrm()`

UNRM: ext4_do_unrm()

```
int ext4_do_unrm(struct super_block *sb,
                 struct inode *dir)
```

Arguments:

- `@sb:` superblock struct
  - should be an ext4fs superblock
  - from superblock can access our unrm radix tree
- `@dir:` directory in which to look for recently rm'd files
  - use this dir's inode number as key into radix tree

UNRM: ext4_do_unrm()

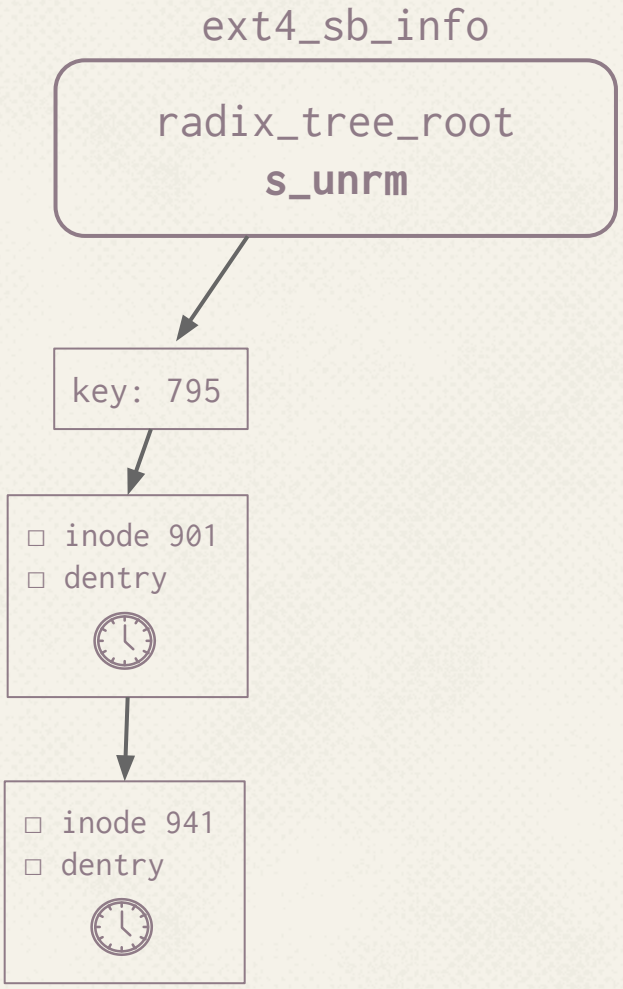
```
int ext4_do_unrm(struct super_block *sb,
                 struct inode *dir)
```

What does `ext4_do_unrm()` do?

- Lookup `dir->i_no` with `radix_tree_lookup()`
  - **key**: directory inode number.
  - **entry**: linked list of deleted files
- iterate list of deleted files: `list_for_each_entry(...)`
  - For each deleted file:
    - add the file back to the directory
      - (remember, directories are just files that are lists of other files, so we're technically just adding an entry back)
    - re-instantiate saved dentry with saved inode
    - cancel scheduled work

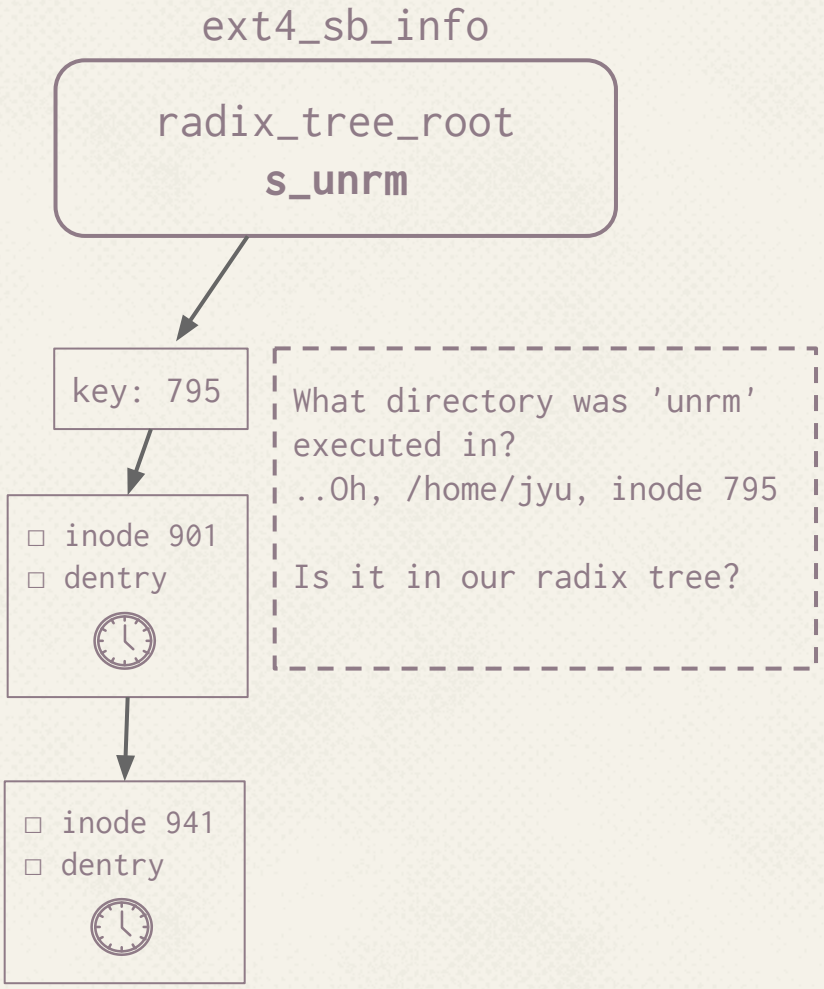
UNRM: ext4_do_unrm()

```
$ pwd
/home/jyu
$ ls
testfile.pdf  track01.mp3  important.doc
$ rm testfile.pdf          ### inode 901
$ rm track01.mp3           ### inode 941
$ ls track01.mp3
track01.mp3: No such file or directory
$
```



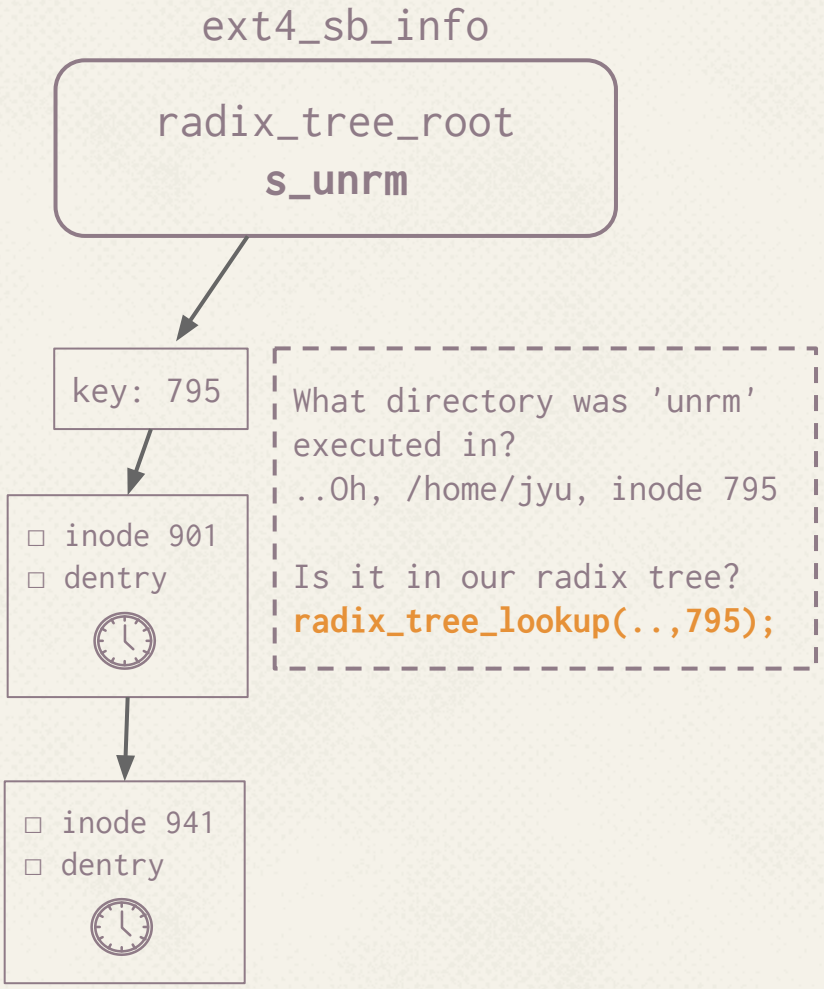
UNRM: ext4_do_unrm()

```
$ pwd
/home/jyu
$ ls
testfile.pdf  track01.mp3  important.doc
$ rm testfile.pdf          ### inode 901
$ rm track01.mp3           ### inode 941
$ ls track01.mp3
track01.mp3: No such file or directory
$ unrm
$
```



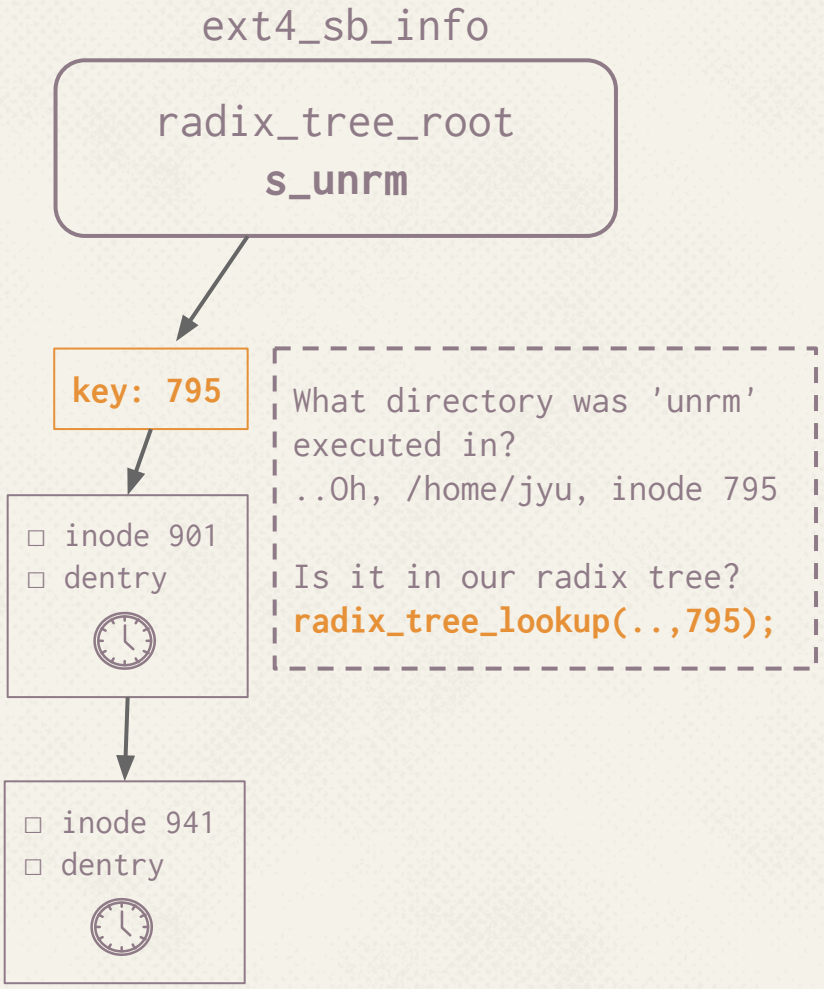
UNRM: ext4_do_unrm()

```
$ pwd
/home/jyu
$ ls
testfile.pdf  track01.mp3  important.doc
$ rm testfile.pdf          ### inode 901
$ rm track01.mp3           ### inode 941
$ ls track01.mp3
track01.mp3: No such file or directory
$ unrm
$
```



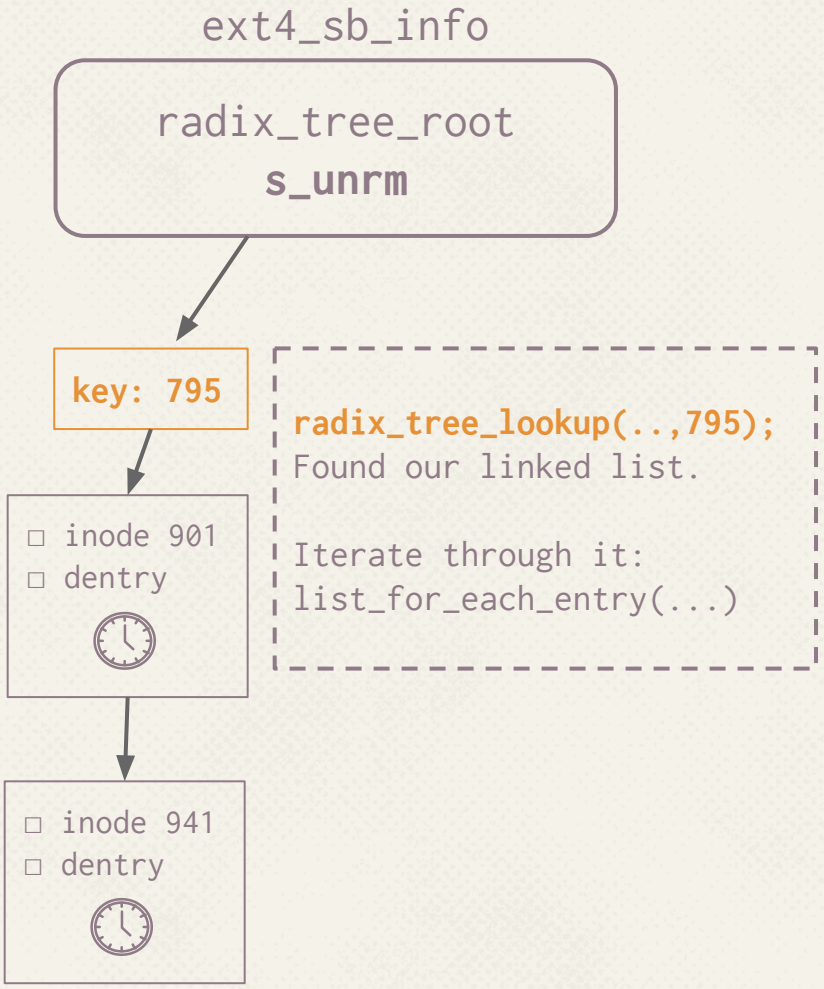
UNRM: ext4_do_unrm()

```
$ pwd
/home/jyu
$ ls
testfile.pdf  track01.mp3  important.doc
$ rm testfile.pdf          ### inode 901
$ rm track01.mp3           ### inode 941
$ ls track01.mp3
track01.mp3: No such file or directory
$ unrm
$
```



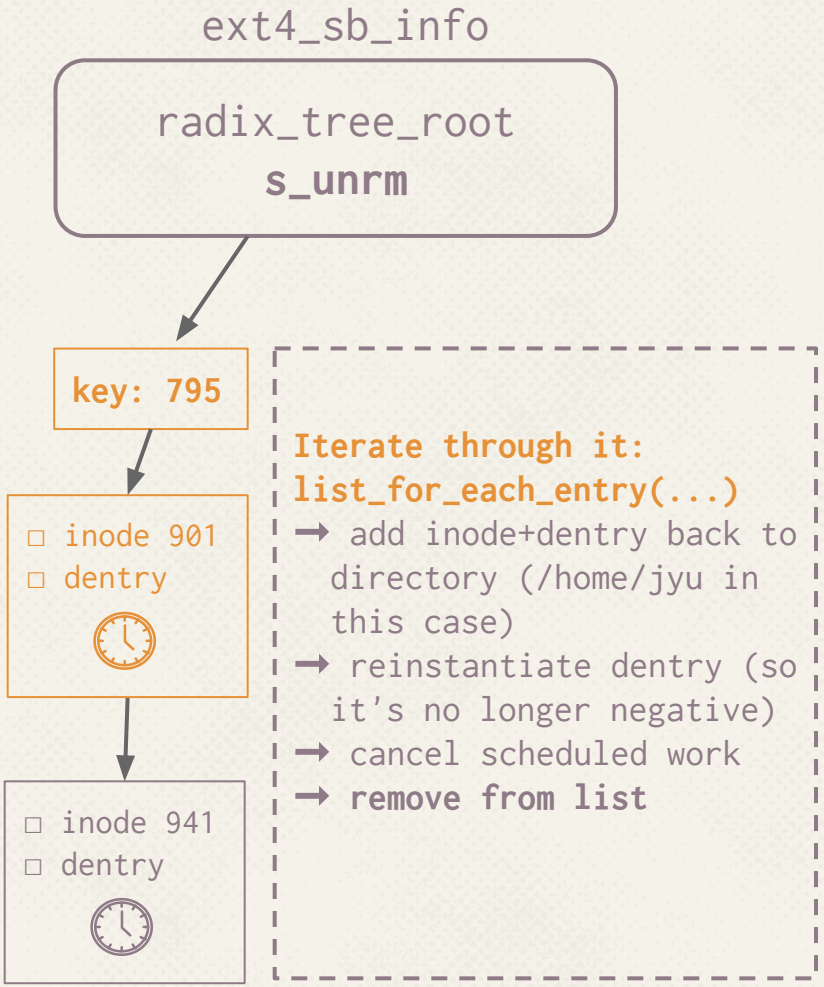
UNRM: ext4_do_unrm()

```
$ pwd
/home/jyu
$ ls
testfile.pdf  track01.mp3  important.doc
$ rm testfile.pdf          ### inode 901
$ rm track01.mp3           ### inode 941
$ ls track01.mp3
track01.mp3: No such file or directory
$ unrm
$
```



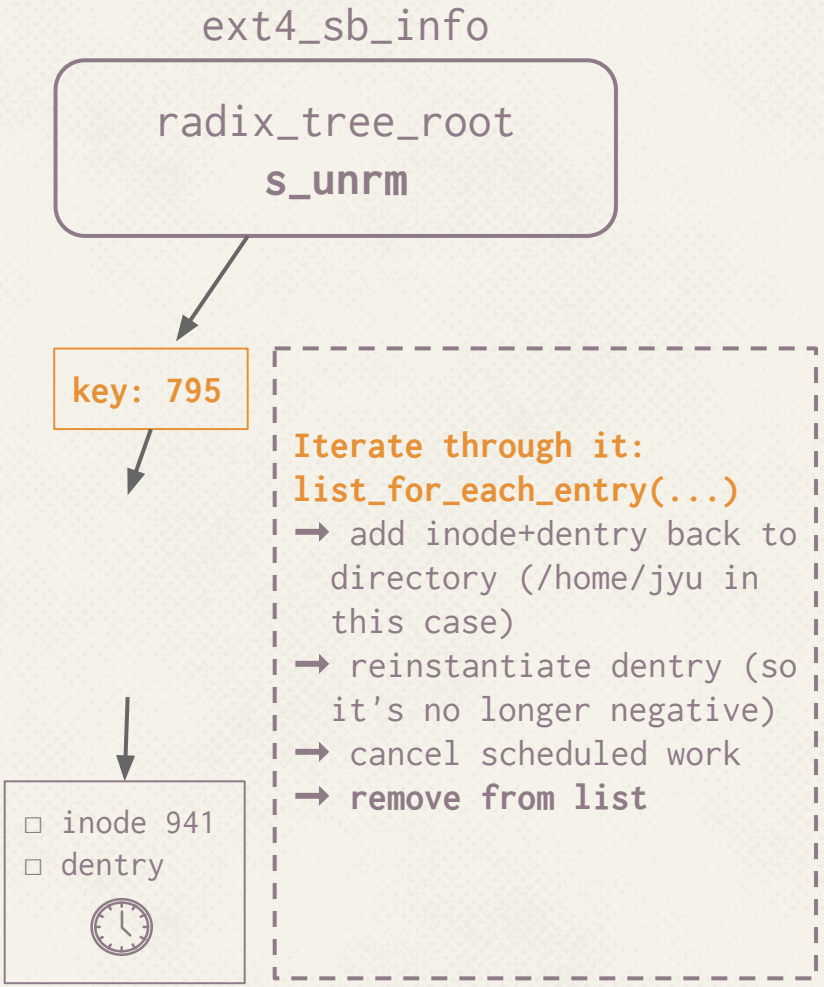
UNRM: ext4_do_unrm()

```
$ pwd
/home/jyu
$ ls
testfile.pdf  track01.mp3  important.doc
$ rm testfile.pdf          ### inode 901
$ rm track01.mp3           ### inode 941
$ ls track01.mp3
track01.mp3: No such file or directory
$ unrm
$
```



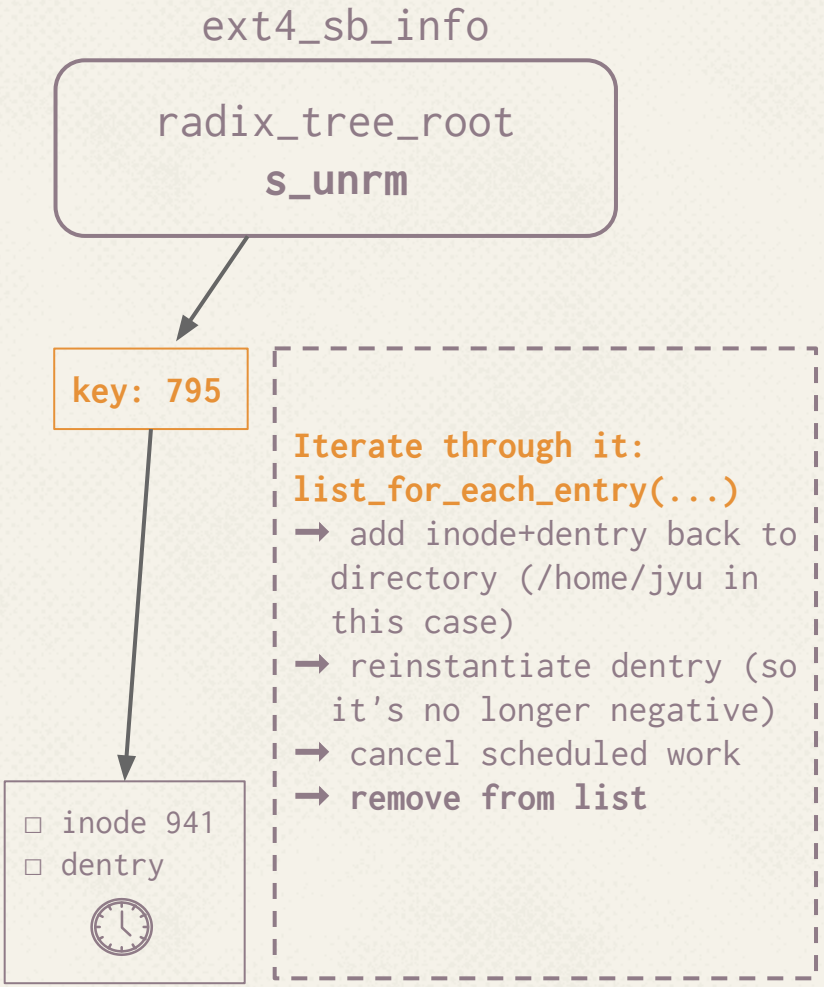
UNRM: ext4_do_unrm()

```
$ pwd
/home/jyu
$ ls
testfile.pdf  track01.mp3  important.doc
$ rm testfile.pdf          ### inode 901
$ rm track01.mp3           ### inode 941
$ ls track01.mp3
track01.mp3: No such file or directory
$ unrm
$
```



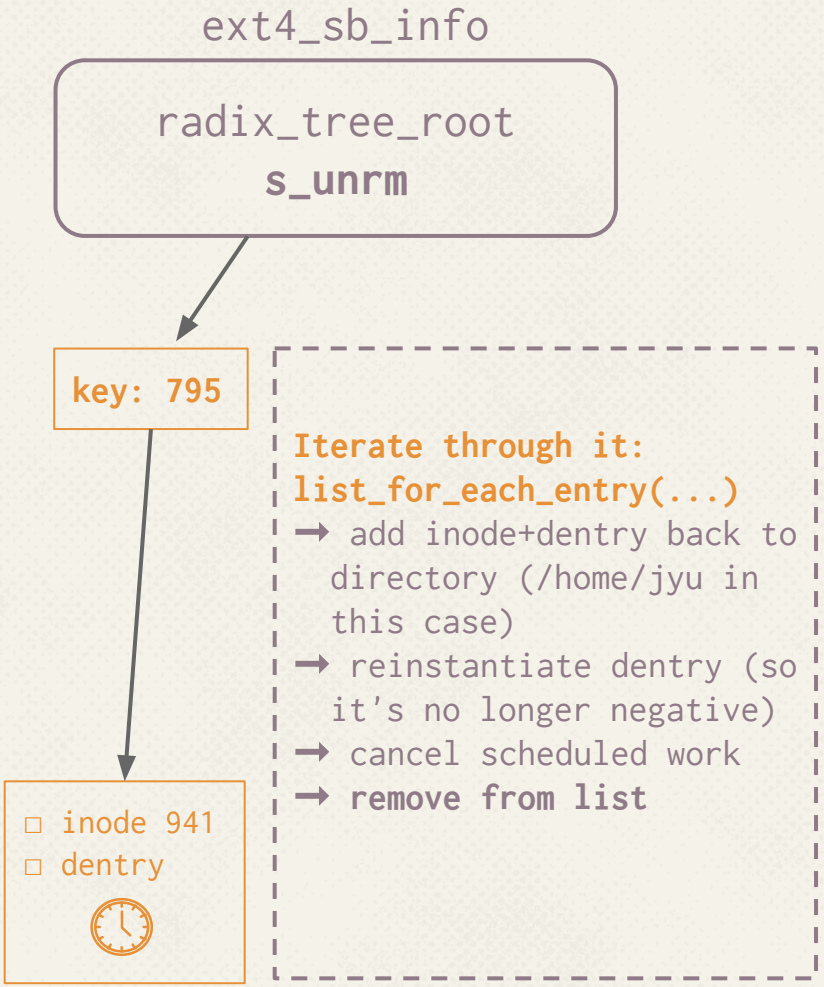
UNRM: ext4_do_unrm()

```
$ pwd
/home/jyu
$ ls
testfile.pdf  track01.mp3  important.doc
$ rm testfile.pdf          ### inode 901
$ rm track01.mp3           ### inode 941
$ ls track01.mp3
track01.mp3: No such file or directory
$ unrm
$
```



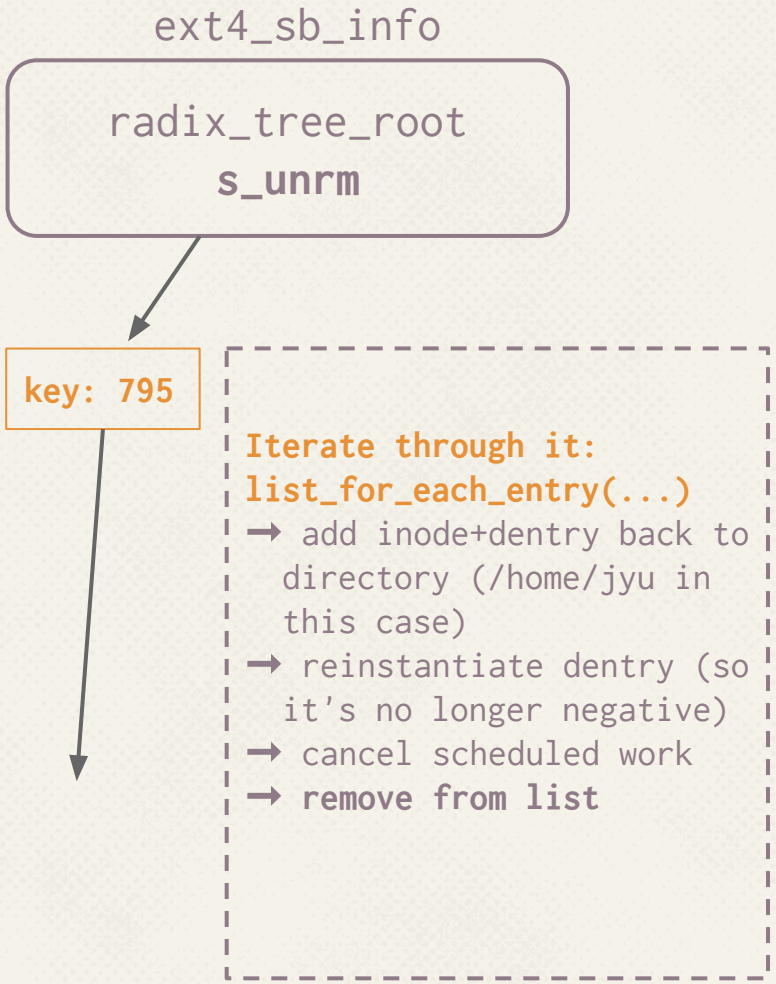
UNRM: ext4_do_unrm()

```
$ pwd
/home/jyu
$ ls
testfile.pdf  track01.mp3  important.doc
$ rm testfile.pdf          ### inode 901
$ rm track01.mp3           ### inode 941
$ ls track01.mp3
track01.mp3: No such file or directory
$ unrm
$
```



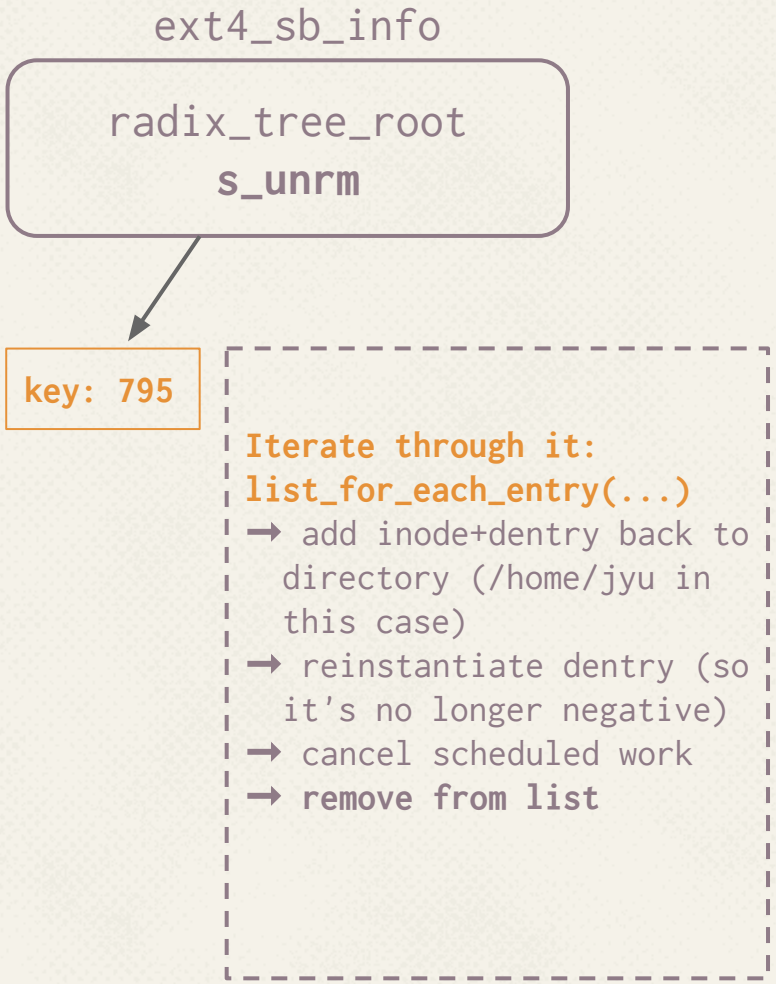
UNRM: ext4_do_unrm()

```
$ pwd
/home/jyu
$ ls
testfile.pdf  track01.mp3  important.doc
$ rm testfile.pdf          ### inode 901
$ rm track01.mp3           ### inode 941
$ ls track01.mp3
track01.mp3: No such file or directory
$ unrm
$
```



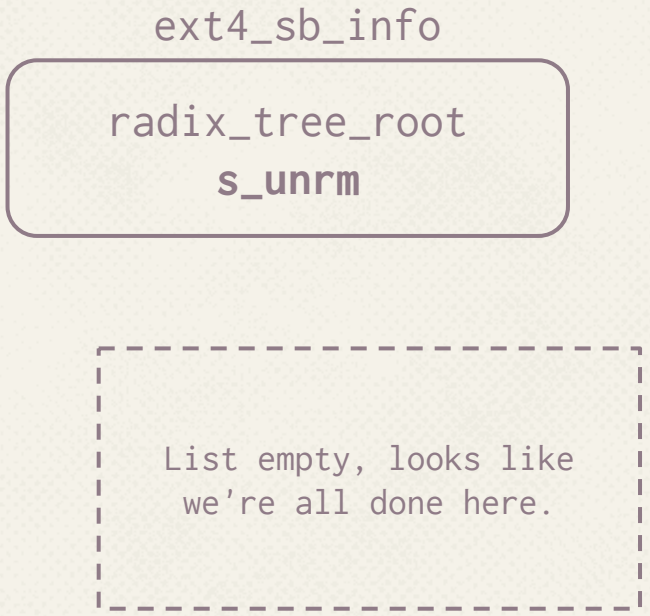
UNRM: ext4_do_unrm()

```
$ pwd
/home/jyu
$ ls
testfile.pdf  track01.mp3  important.doc
$ rm testfile.pdf          ### inode 901
$ rm track01.mp3           ### inode 941
$ ls track01.mp3
track01.mp3: No such file or directory
$ unrm
$
```



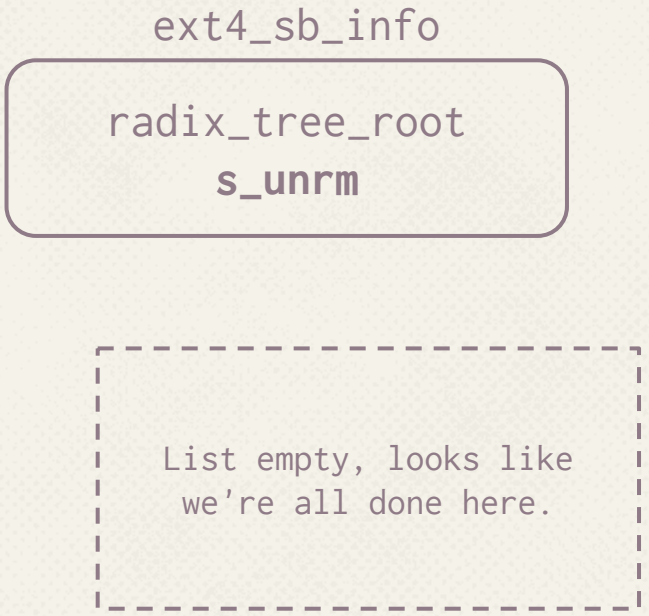
UNRM: ext4_do_unrm()

```
$ pwd
/home/jyu
$ ls
testfile.pdf  track01.mp3  important.doc
$ rm testfile.pdf          ### inode 901
$ rm track01.mp3           ### inode 941
$ ls track01.mp3
track01.mp3: No such file or directory
$ unrm
$ ▌
```



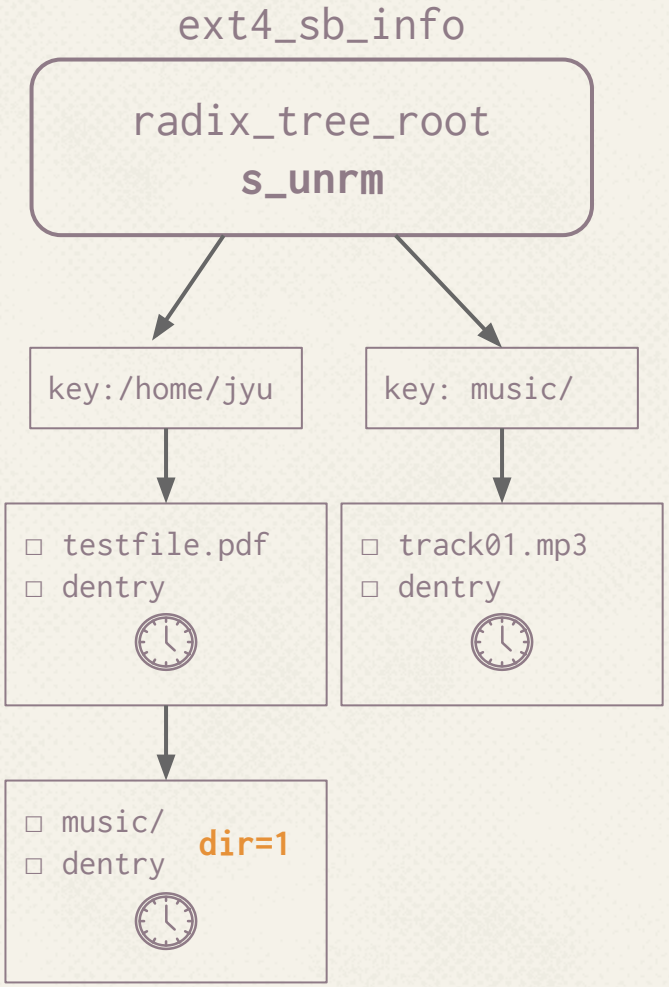
UNRM: ext4_do_unrm()

```
$ pwd
/home/jyu
$ ls
testfile.pdf  track01.mp3  important.doc
$ rm testfile.pdf          ### inode 901
$ rm track01.mp3           ### inode 941
$ ls track01.mp3
track01.mp3: No such file or directory
$ unrm
$ ls                       ### success!
testfile.pdf  track01.mp3  important.doc
$
```



RECURSIVE UNRM

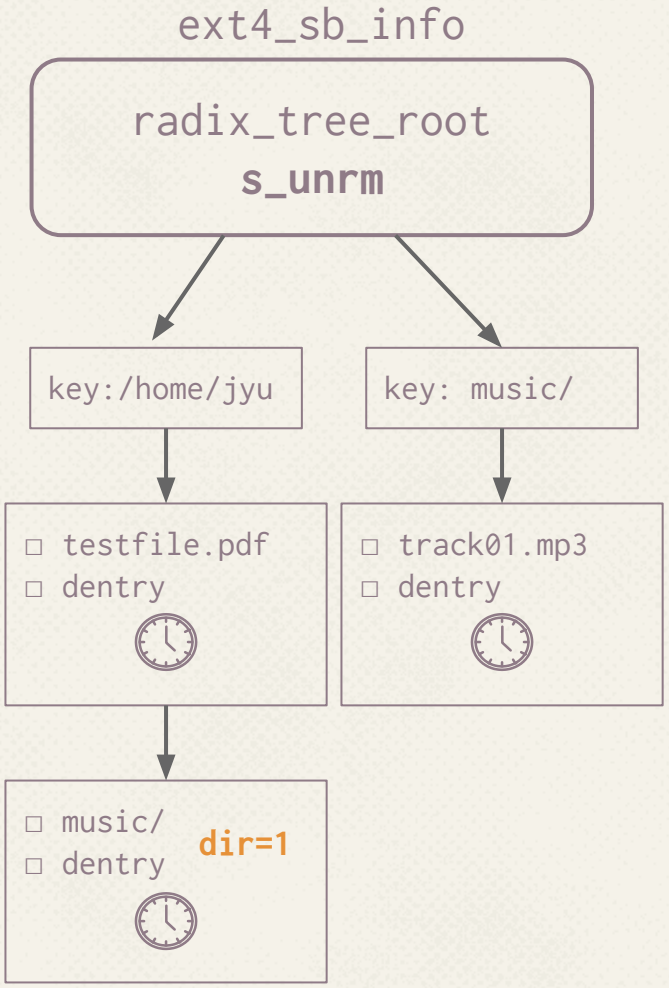
```
$ pwd
/home/jyu/
$ ls
testfile.pdf  music/
$ ls music/
track01.mp3
$ rm -r *              ### BOOM
$
```



* inode #'s replaced with filenames for simplicity

RECURSIVE UNRM

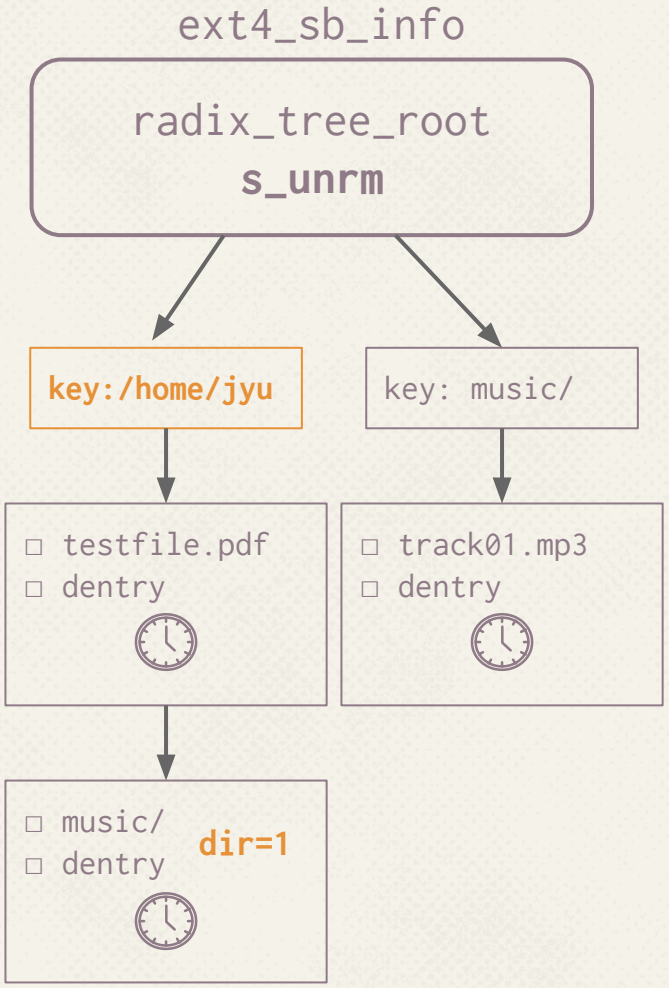
```
$ pwd
/home/jyu/
$ ls
testfile.pdf  music/
$ ls music/
track01.mp3
$ rm -r *              ### BOOM
$ unrm
$
```



* inode #'s replaced with filenames for simplicity

RECURSIVE UNRM

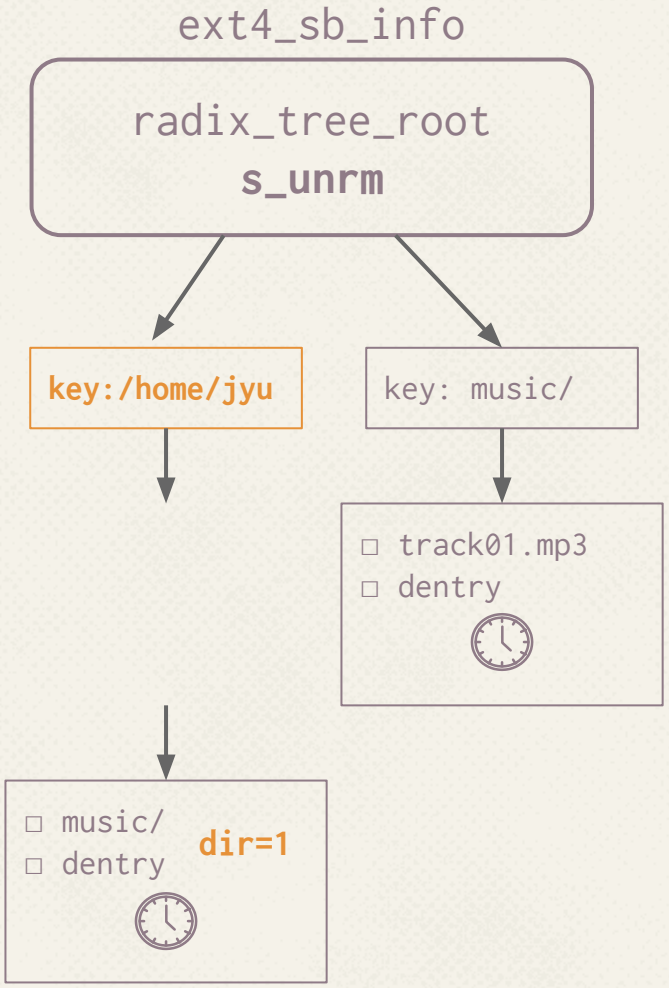
```
$ pwd
/home/jyu/
$ ls
testfile.pdf  music/
$ ls music/
track01.mp3
$ rm -r *              ### BOOM
$ unrm
$
```



* inode #'s replaced with filenames for simplicity

RECURSIVE UNRM

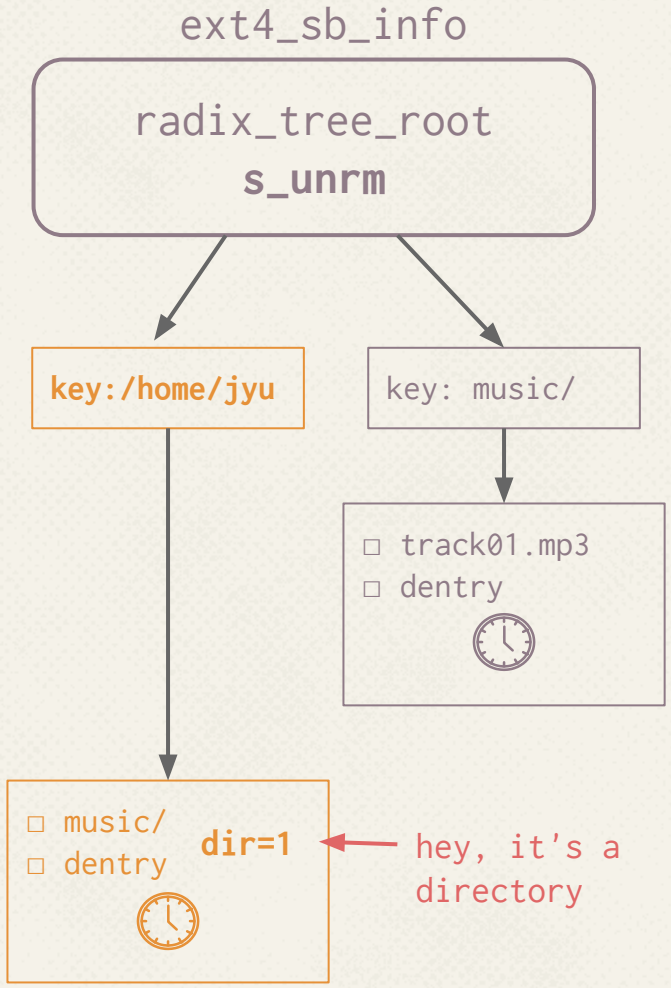
```
$ pwd
/home/jyu/
$ ls
testfile.pdf  music/
$ ls music/
track01.mp3
$ rm -r *              ### BOOM
$ unrm
$
```



* inode #'s replaced with filenames for simplicity

RECURSIVE UNRM

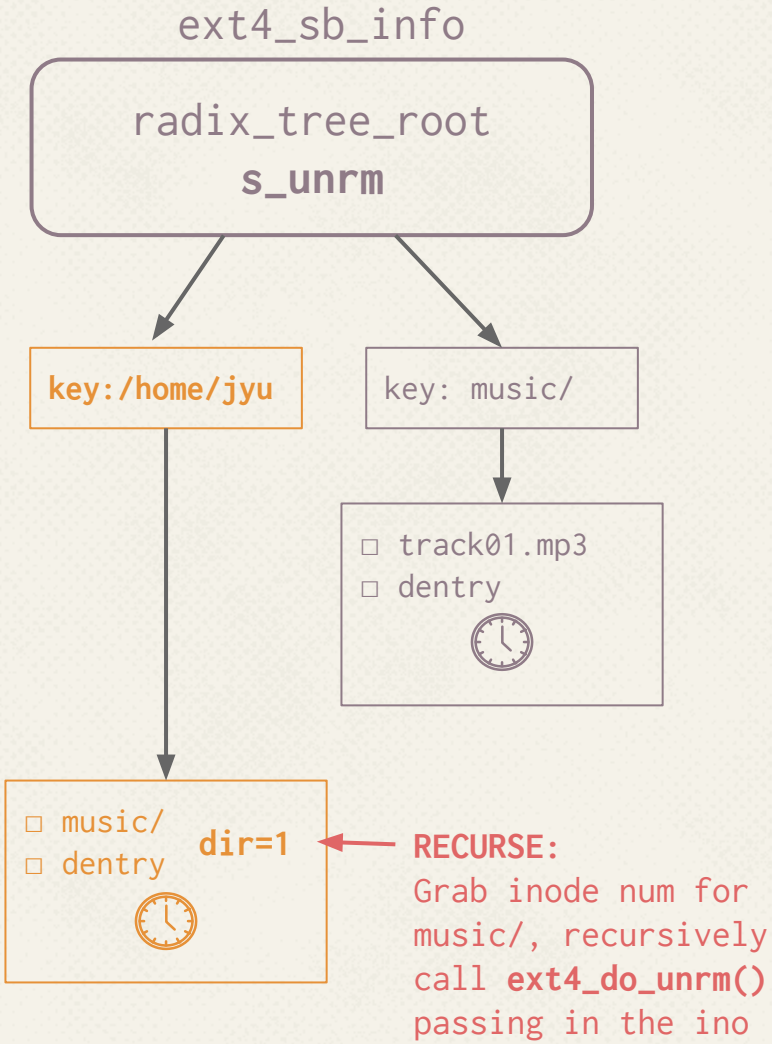
```
$ pwd
/home/jyu/
$ ls
testfile.pdf  music/
$ ls music/
track01.mp3
$ rm -r *              ### BOOM
$ unrm
$
```



* inode #'s replaced with filenames for simplicity

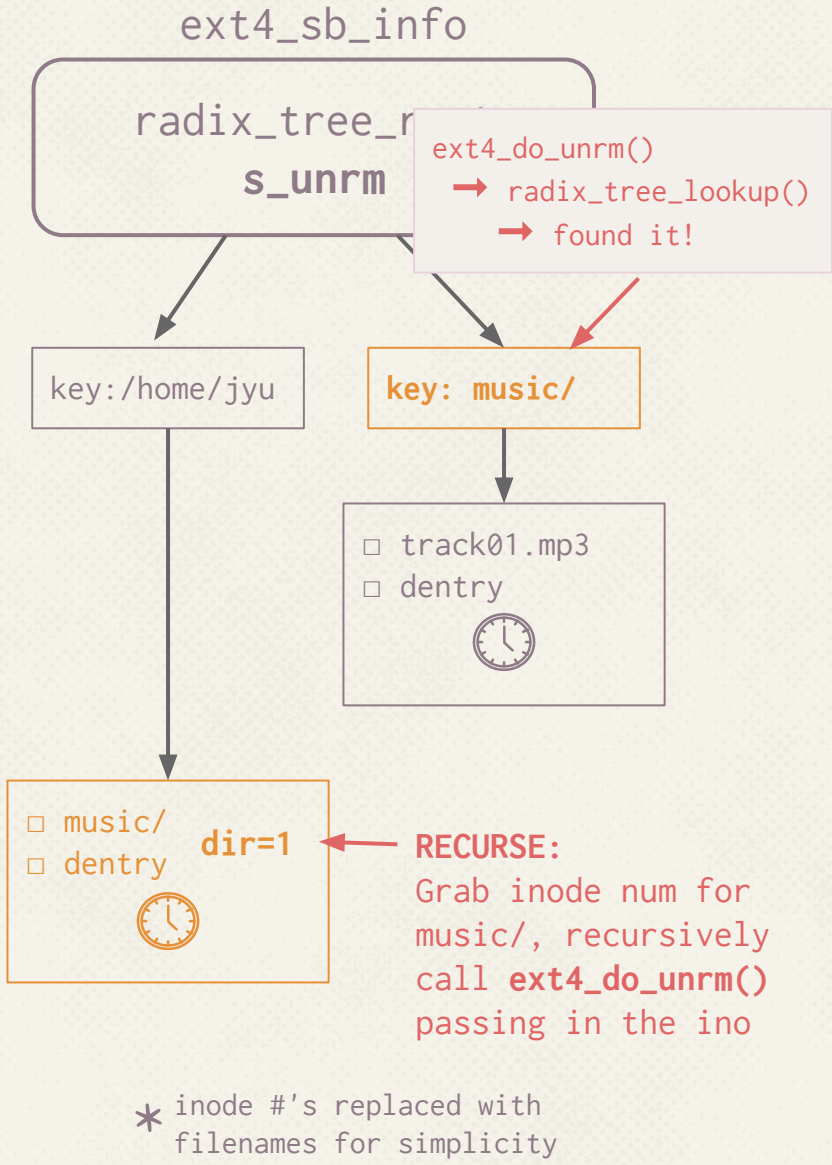
RECURSIVE UNRM

```
$ pwd
/home/jyu/
$ ls
testfile.pdf  music/
$ ls music/
track01.mp3
$ rm -r *              ### BOOM
$ unrm
$
```



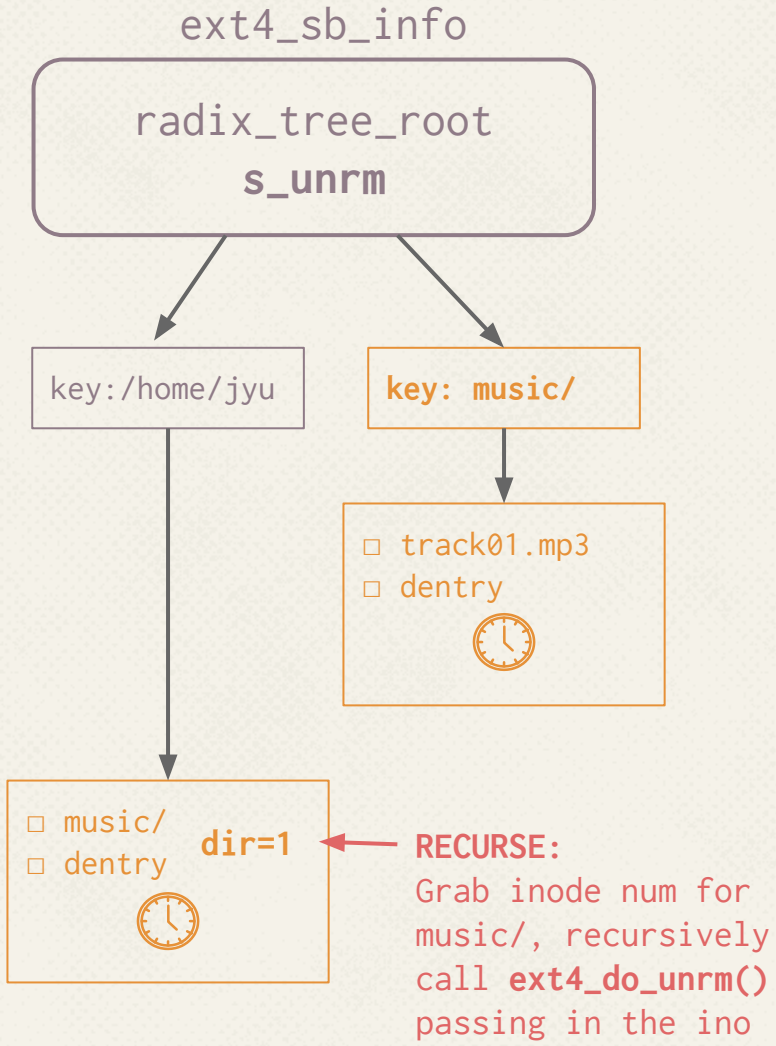
RECURSIVE UNRM

```
$ pwd
/home/jyu/
$ ls
testfile.pdf  music/
$ ls music/
track01.mp3
$ rm -r *              ### BOOM
$ unrm
$
```



RECURSIVE UNRM

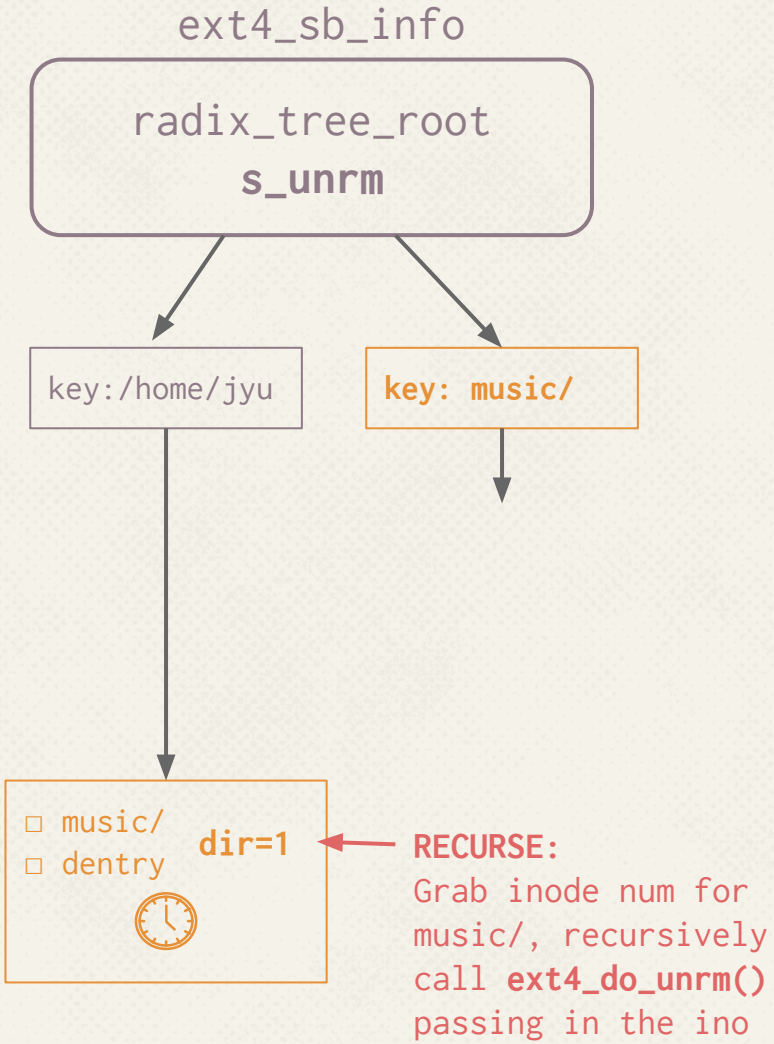
```
$ pwd
/home/jyu/
$ ls
testfile.pdf  music/
$ ls music/
track01.mp3
$ rm -r *              ### BOOM
$ unrm
$
```



* inode #'s replaced with filenames for simplicity

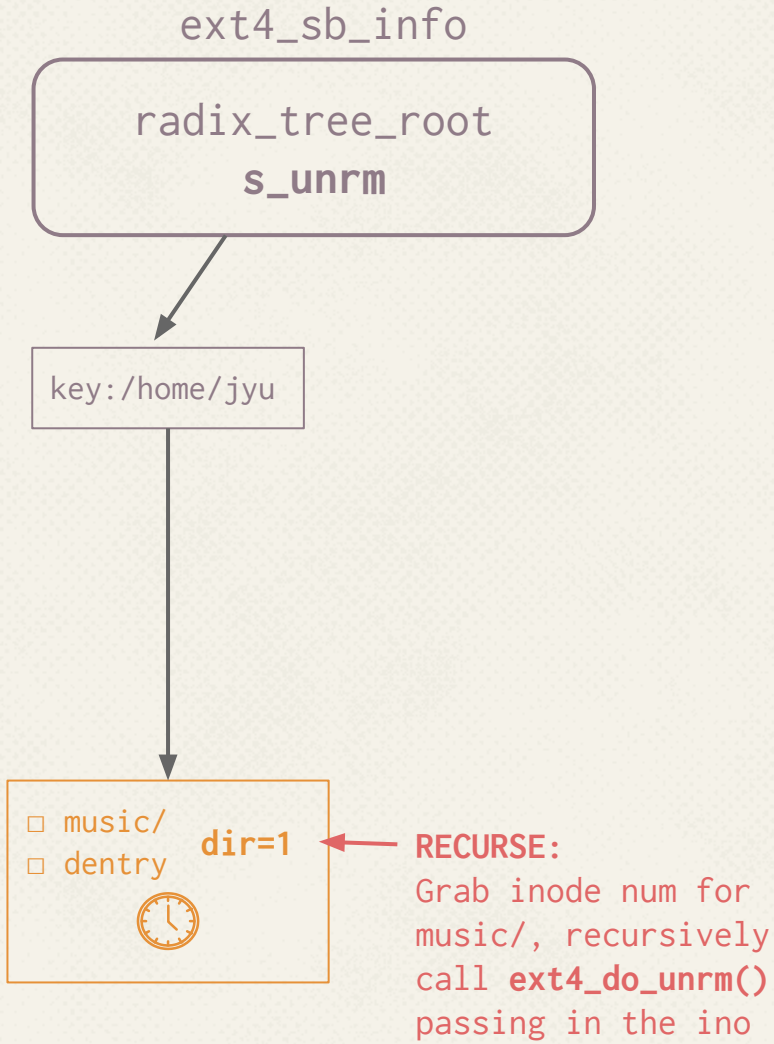
RECURSIVE UNRM

```
$ pwd
/home/jyu/
$ ls
testfile.pdf  music/
$ ls music/
track01.mp3
$ rm -r *            ### BOOM
$ unrm
$
```



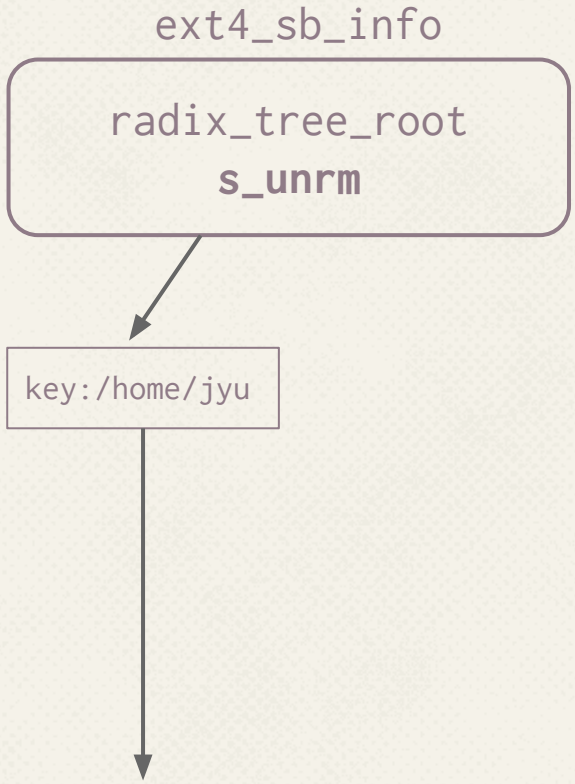
RECURSIVE UNRM

```
$ pwd
/home/jyu/
$ ls
testfile.pdf  music/
$ ls music/
track01.mp3
$ rm -r *            ### BOOM
$ unrm
$
```



RECURSIVE UNRM

```
$ pwd
/home/jyu/
$ ls
testfile.pdf  music/
$ ls music/
track01.mp3
$ rm -r *              ### BOOM
$ unrm
$
```



* inode #'s replaced with filenames for simplicity

RECURSIVE UNRM

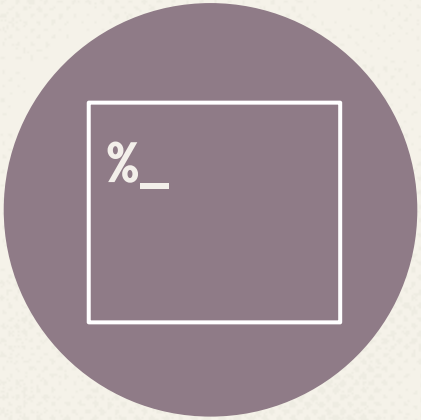
```
$ pwd
/home/jyu/
$ ls
testfile.pdf  music/
$ ls music/
track01.mp3
$ rm -r *              ### BOOM
$ unrm
$ tree                 ### Show dir tree
 .
├── music             ### Hurray!
│   └── track01.mp3
└── testfile.pdf
```

ext4_sb_info

radix_tree_root
s_unrm

List empty, looks like
we're all done here.

* inode #'s replaced with filenames for simplicity



DEMO

...Let's hope nothing explodes

UNRM: ISSUES & FUTURE WORK

- **Warning!** unrm is just a fun hack! Was never meant to be used in production

With that said...Here are some ideas for future work...

- **Memory hog**! Removal of thousands of files = thousands of nodes in memory
- **Security** - take permissions into account
- **Make work queue delay adjustable** - make the period of time adjustable (currently hardcoded at 30 seconds)
- **Filesystem can be left inconsistent** in the event of a sudden crash (before unlinking or unrm process finishes)
- ....probably more! Feel free to send pull requests!

FULL KERNEL SOURCE
GITHUB.COM/FLAMING-TOAST/UNRM

CONTACT
JYU AT EISEN.IO

Thanks!