CS 1550

Week 11 – Project 4

Teaching Assistant

Xiaoyu(Veronica) Liang

Overview

- FUSE is a **Linux kernel extension** that allows for a user space program to provide the implementations for the various file-related syscalls

- Goal: Use FUSE to create our own file system

Installation of FUSE

- Kernel is installed
- Install libraries and example programs

```
cd /u/OSLab/USERNAME
cp /u/OSLab/original/fuse-2.7.0.tar.gz .
tar xvfz fuse-2.7.0.tar.gz
cd fuse-2.7.0
./configure
make
```

Installation of FUSE

- Kernel is installed
- Install libraries and example programs

  cd /u/OSLab/USERNAME
  cp /u/OSLab/original/fuse-2.7.0.tar.gz .
  tar xvfz fuse-2.7.0.tar.gz
  cd fuse-2.7.0
  ./configure
  make  This compiles the examples.

Setting up the Environment Variables

- cd ~
- chmod u+w .bash_profile
  Gives you the write permission to .bash_profile
- nano .bash_profile
- Scroll down to the end of the file until you see the line:
  "# Define your own private shell functions and other commends here"
- Add the following lines (spacing around '[' and ']' characters need to be there!)
  if ["$HOSTNAME" = "thoth.cs.pitt.edu"]; then
  source /opt/set/specific_profile.sh;
  fi
- Save the file and quit
- chmod u-w .bash_profile
- .bash_profile will not run until the next time you log in
- source /opt/set/specific_profile.sh

FUSE Example

- cd fuse-2.7.0/example
- mkdir testmount (create mount point)

  A mount point is a location in the UNIX hierarchical file system where a new device or file system is located
- ls -al testmount

  Empty, only see: . and ..
- ./hello testmount
- ls -al testmount

  Should see 3 entiries: ., .., hello
- fusermount –u testmount

  Unmount the file system we just used when we are done, or need to make changes to the program. Always need to do.

Debug Mode

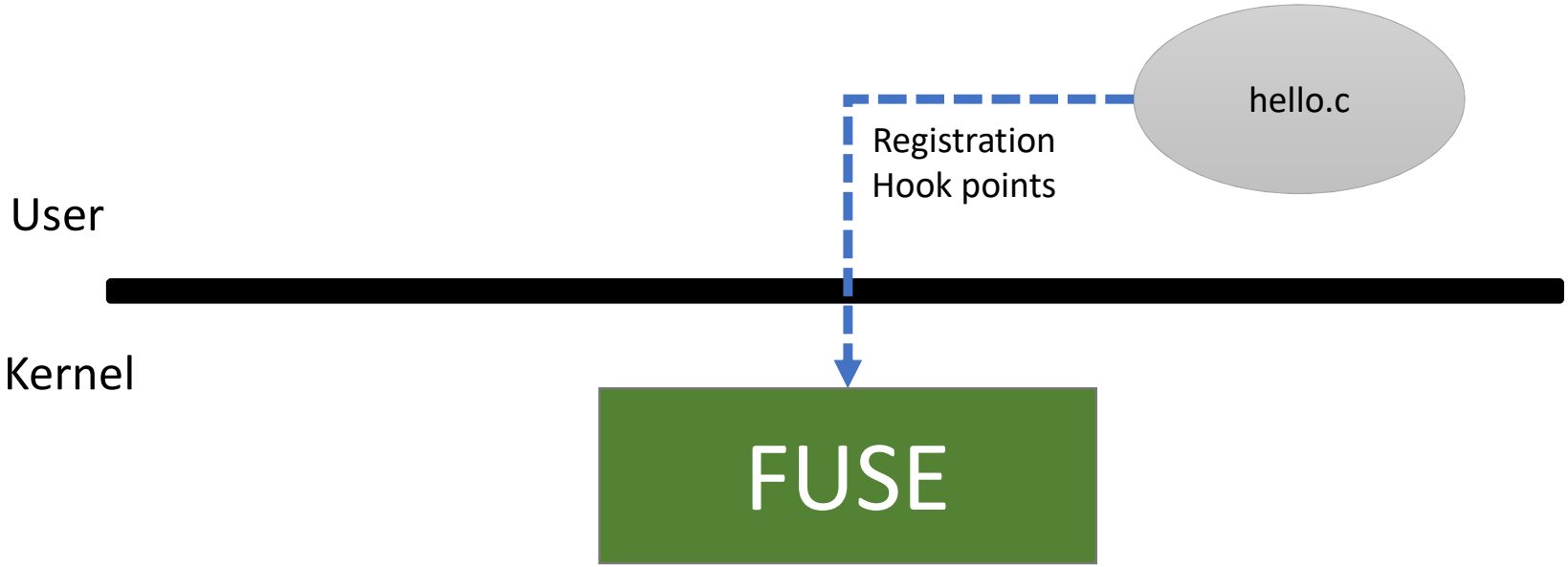
- Testing is to launch a FUSE application with the –d option

  ./hello –d testmount

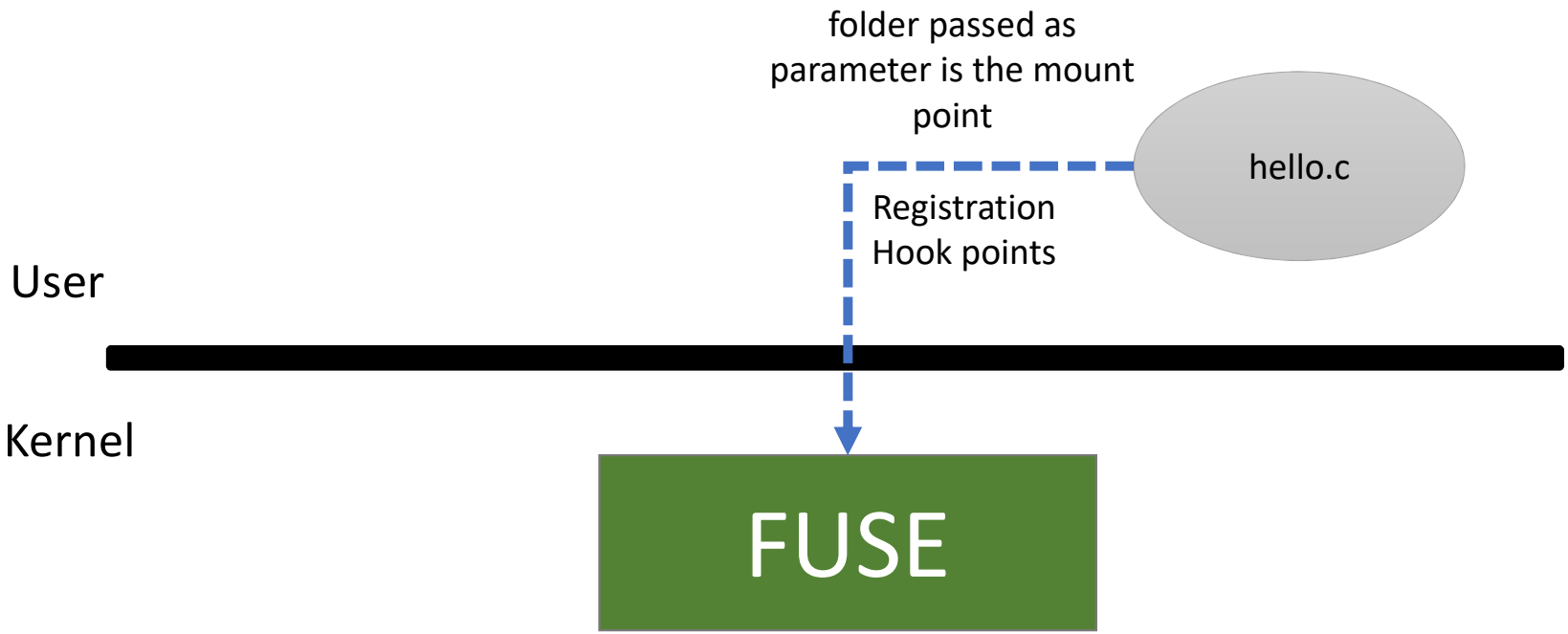
  This will keep the program in the foreground, and it will print out every message that the application receives, and interpret the return values that you're getting back.
- Open a second terminal window and try your testing procedures.
- If you do a CTRL + C in the first window, you may not need to unmount the file system.
- IMPORTANT: if your program crashes or you abort it, you definitely need to do the fusermount. Otherwise, you will get a confusing "Transport endpoint not connected" message the next time you try to mount the system.

Overview

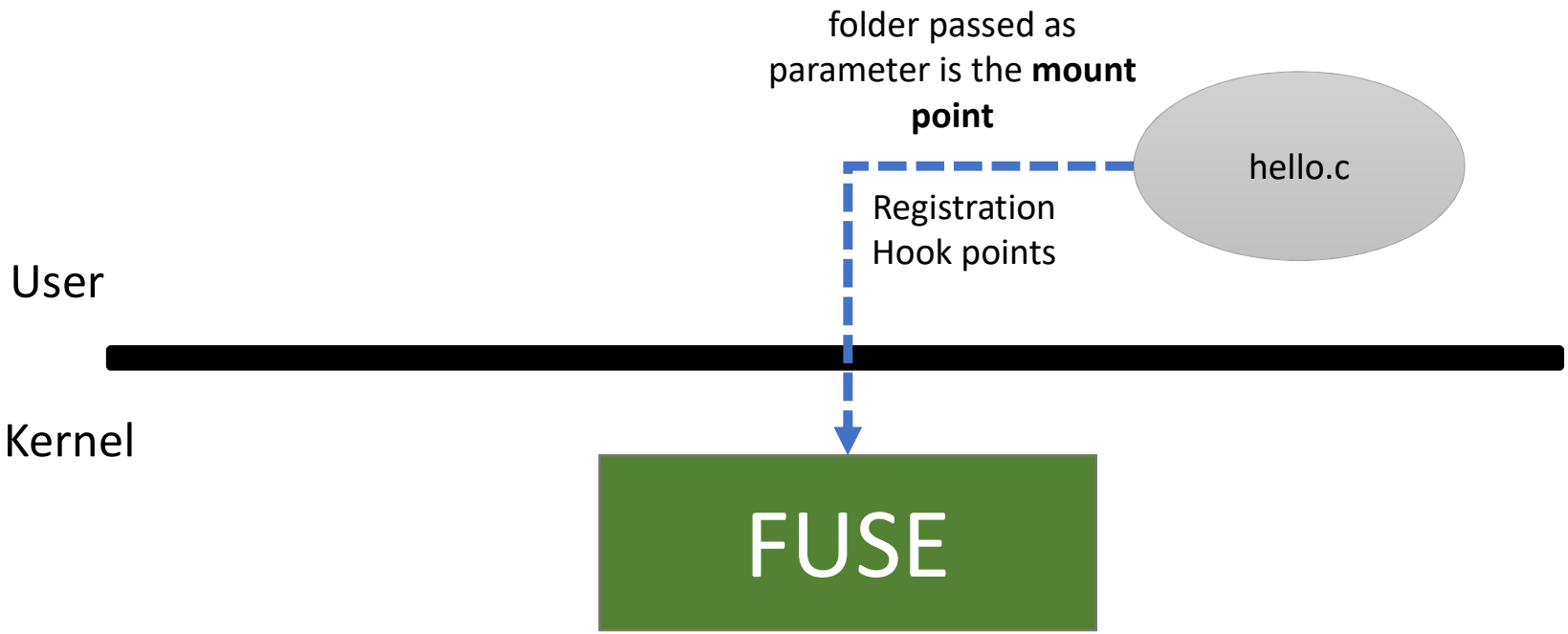
Overview



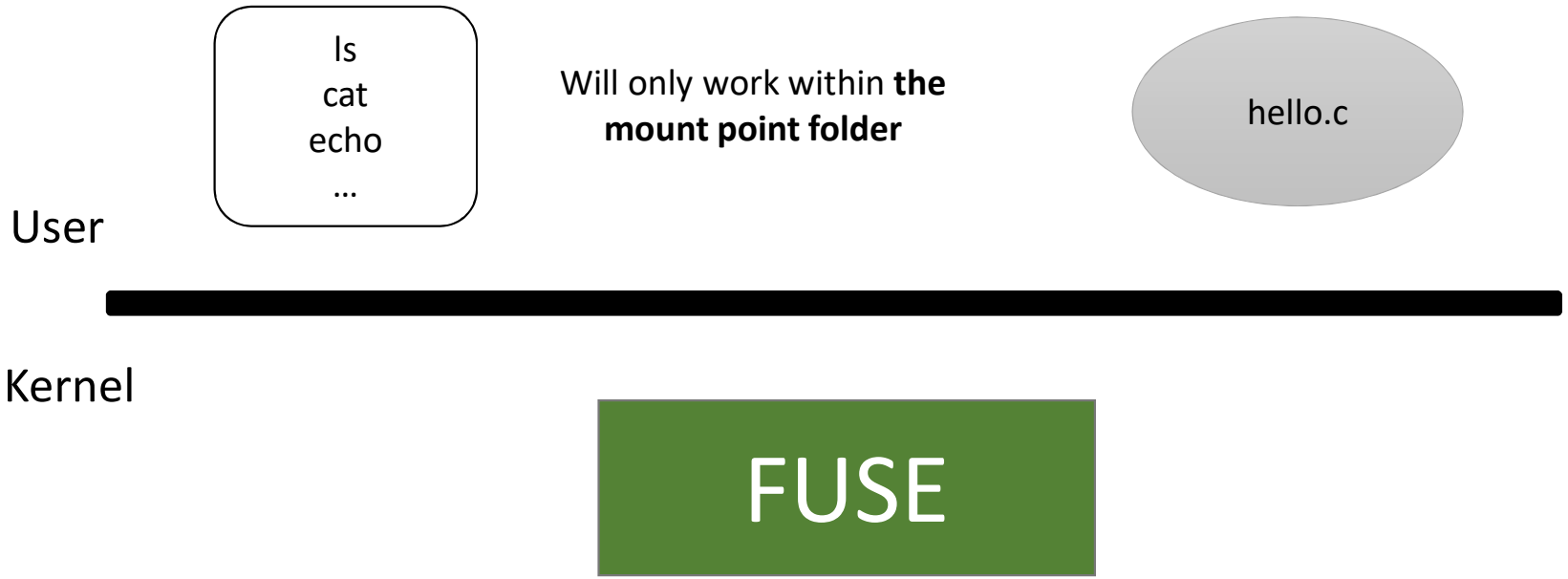
Overview


folder passed as parameter is the mount point
hello.c
Registration Hook points
User
Kernel
FUSE

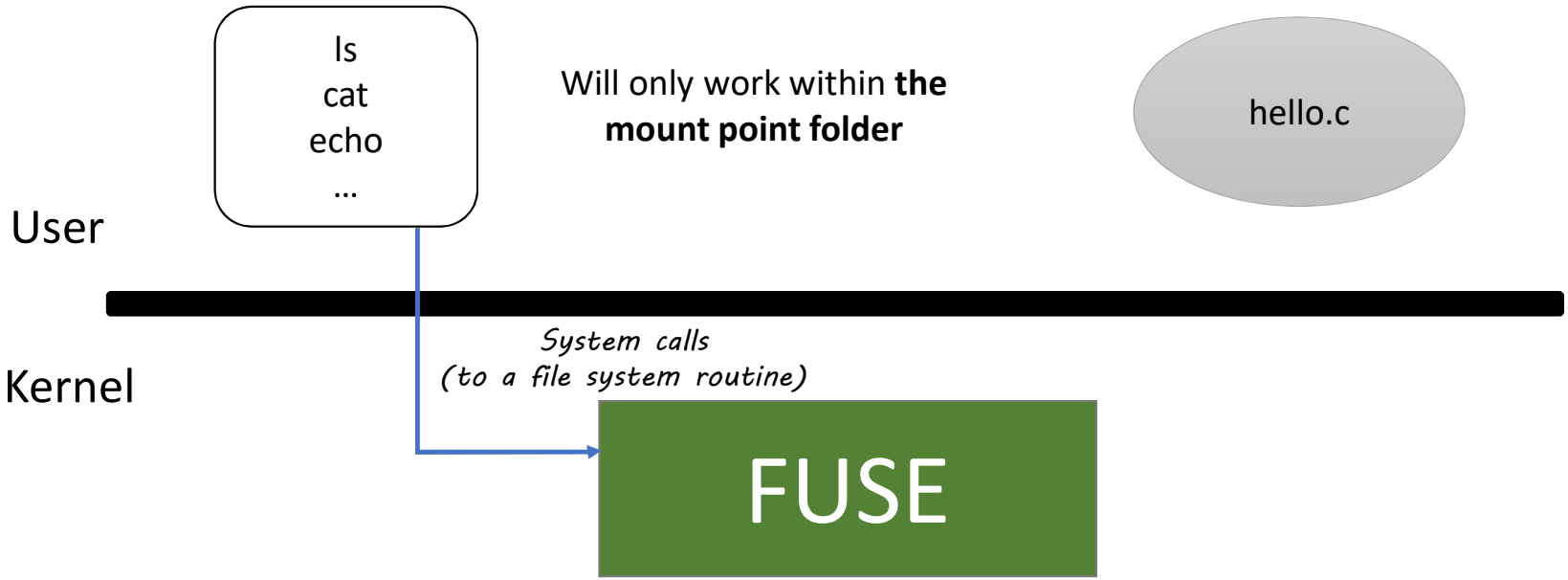
Overview


folder passed as parameter is the **mount point**
hello.c
Registration Hook points
User
Kernel
FUSE

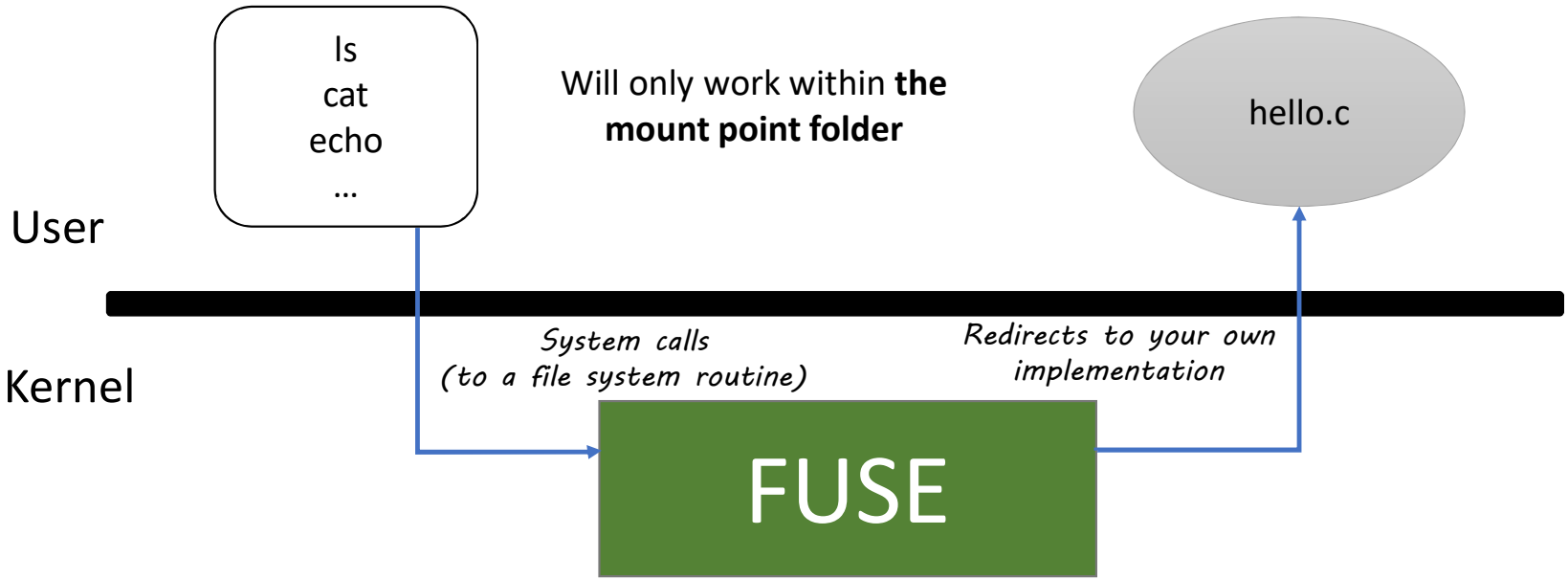
Overview


ls
cat
echo
…
Will only work within **the mount point folder**
hello.c
User
Kernel
FUSE

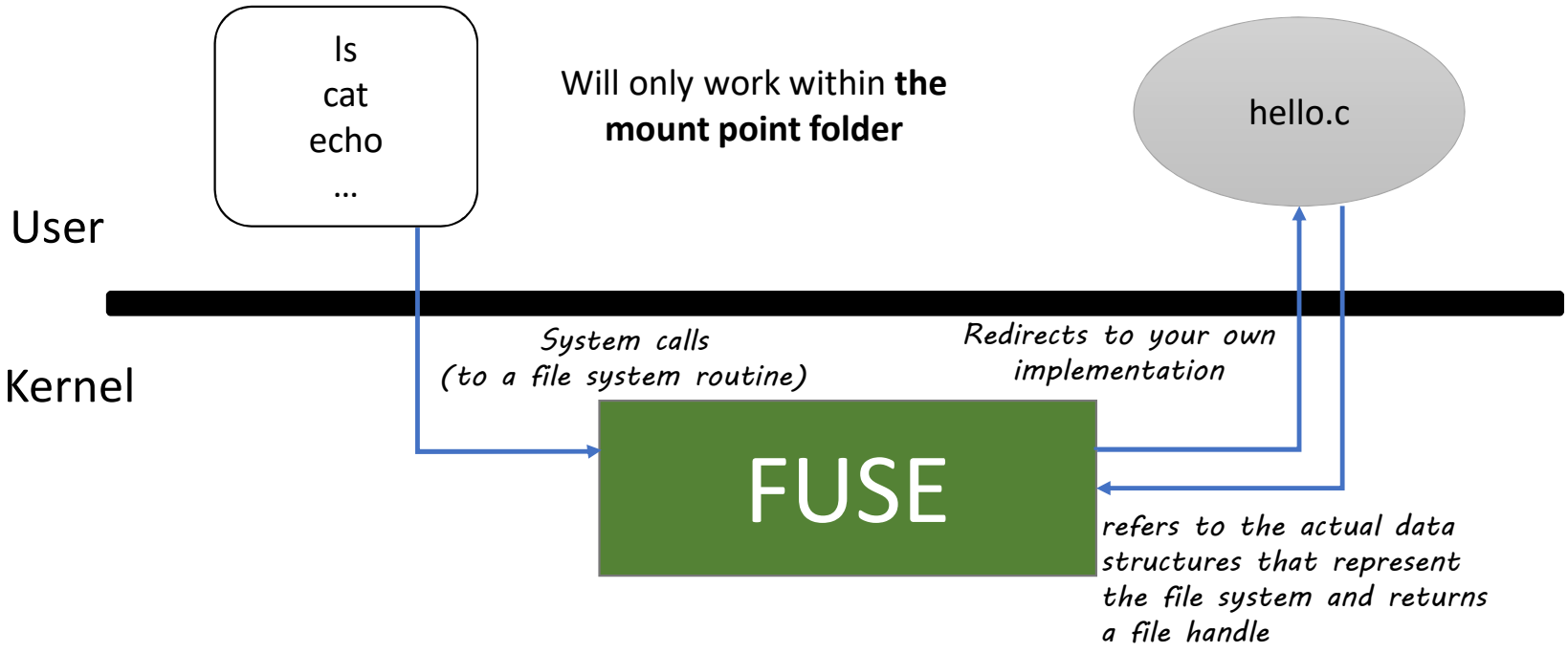
Overview


ls
cat
echo
…
Will only work within **the mount point folder**
hello.c
User
Kernel
System calls
(to a file system routine)
FUSE

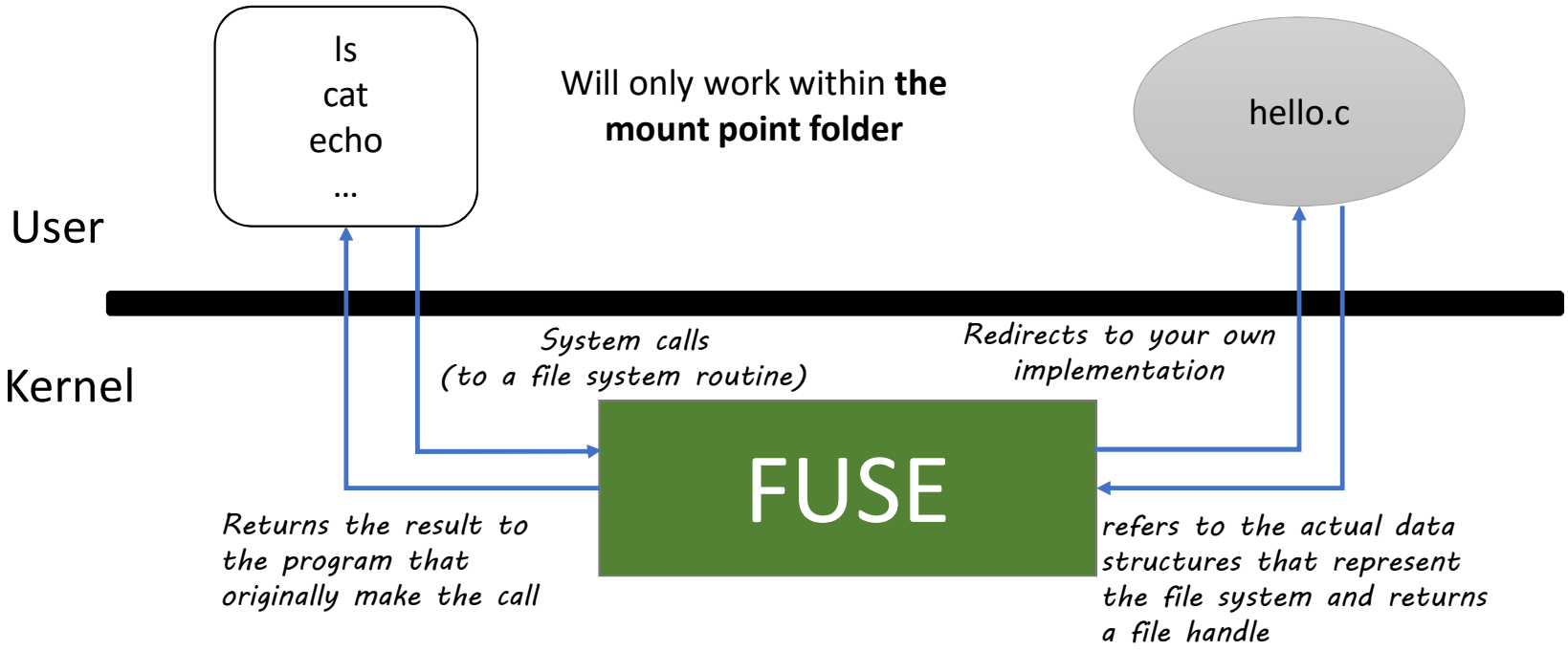
Overview


ls
cat
echo
…
Will only work within **the mount point folder**
hello.c
User
Kernel
System calls
(to a file system routine)
Redirects to your own implementation
FUSE

Overview


ls
cat
echo
...
Will only work within **the mount point folder**
hello.c
User
Kernel
System calls
(to a file system routine)
Redirects to your own implementation
FUSE
refers to the actual data structures that represent the file system and returns a file handle

Overview


ls
cat
echo
...
Will only work within **the mount point folder**
hello.c
User
Kernel
System calls
(to a file system routine)
Redirects to your own implementation
FUSE
Returns the result to the program that originally make the call
refers to the actual data structures that represent the file system and returns a file handle