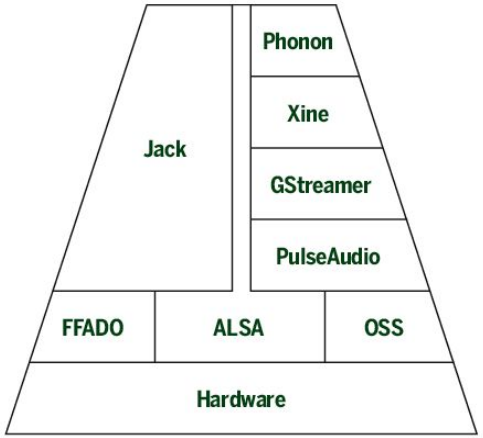
ALSA Recorder: Record What We Hear From Computer

Norrathep Rattanavipanon
Erik Efrain Henriquez

Motivation

- Learn how to record streaming data
  - Protected and unprotected
  - Video and audio
  - Same quality
- Focus on protected audio data
  - Spotify on Linux

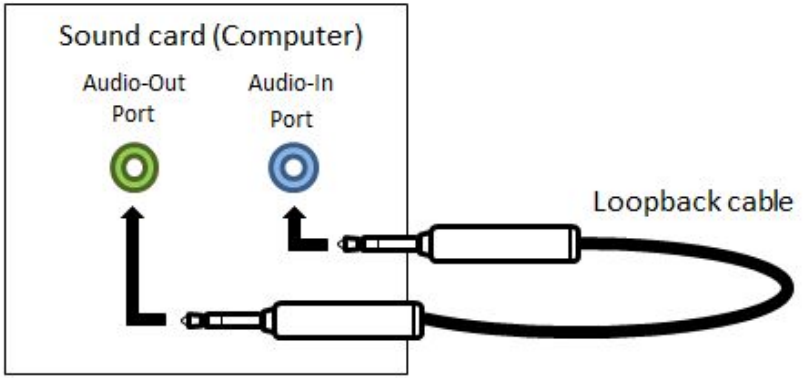
Audio Layer in Linux



- Many frontends
  - PulseAudio is popular
  - Still use underlying interfaces to communicate with hardware
- Advanced Linux Sound Architecture (ALSA)
  - Gives applications a direct interface to hardware devices (i.e. sound cards)
  - Well-established open-source interface

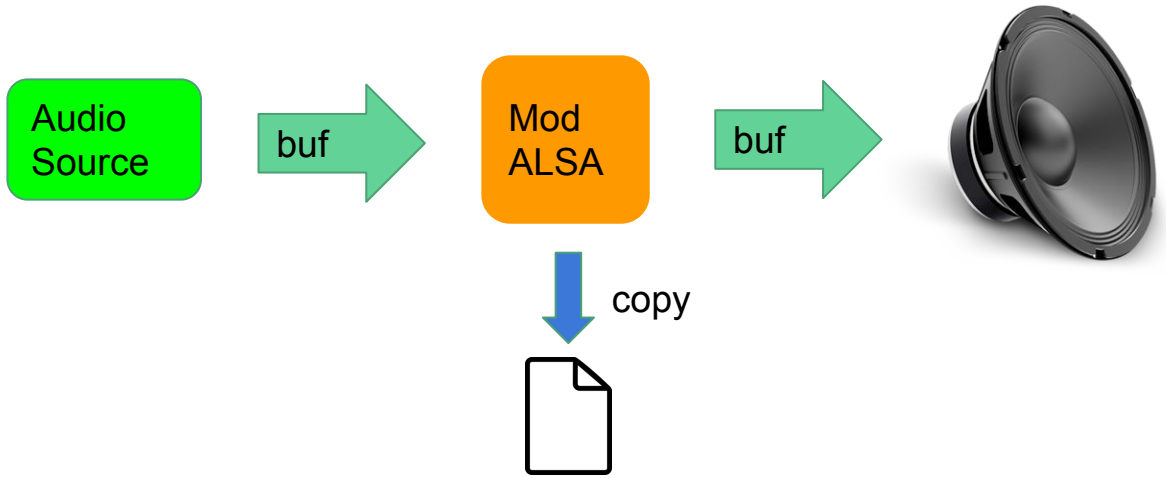
Existing method : Redirecting audio

- ALSA Configuration
  - Write Stereo Mix to a file
- Audacity
  - Manually specify the program that's being recorded



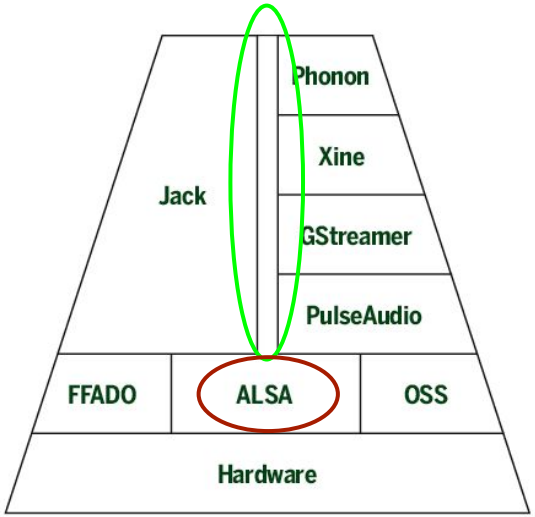
Design

- Audio data is decrypted before calling ALSA
- Modify ALSA source code
  - Whenever play an audio, buffer it and save a copy as a raw audio file.
  - Process a raw file into WAV / MP3 file later



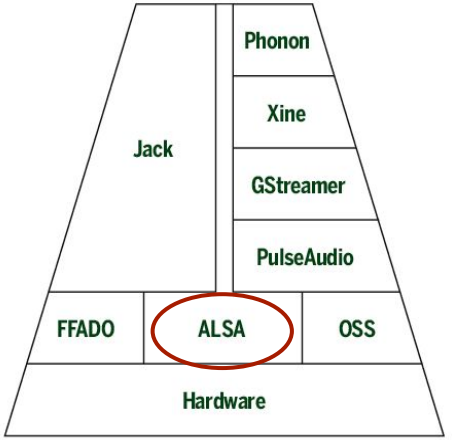
Design

- Audio data is decrypted before calling ALSA
- Modify ALSA source code
  - Whenever play an audio from buffer, save a copy as a raw audio file. Process a raw file into WAV file later
- Two methods
  - Kernel layer
  - API layer



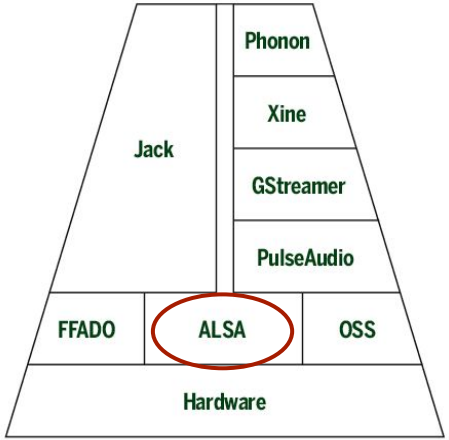
Method 1 : Kernel Layer

- **Linux/sound/core/pcm_native.c**
  - Function snd_pcm_playback_ioctl1



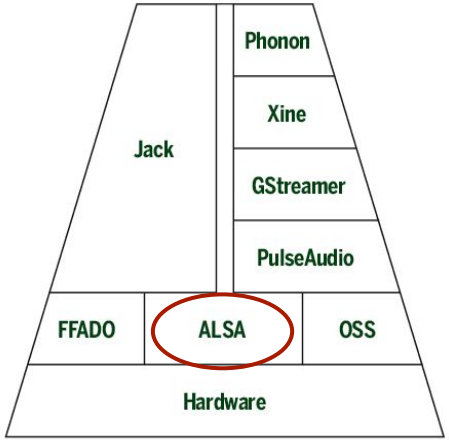
Method 1 : Kernel Layer

```
static int snd_pcm_playback_ioctl1(struct file *file,
                           struct snd_pcm_substream *substream,
                           unsigned int cmd, void __user *arg)
{
      ...

      switch (cmd) {
      case SNDRV_PCM_IOCTL_WRITEI_FRAMES:
      {
            ...
            result = snd_pcm_lib_write(substream, xferi.buf, xferi.frames);
            __put_user(result, &_xferi->result);
            ...
      }
      ...
}
```



Method 1 : Kernel Layer

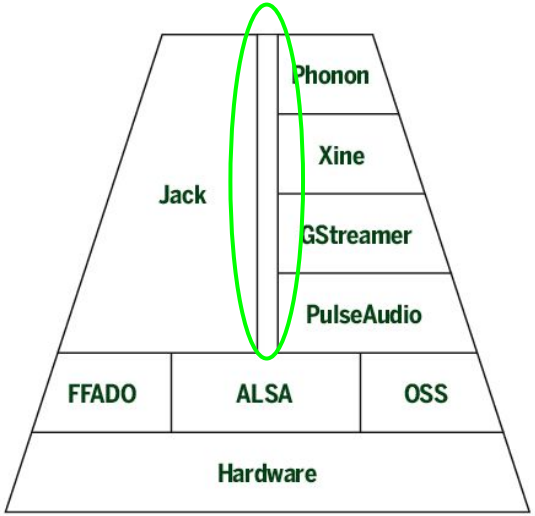
```
static int snd_pcm_playback_ioctl1(struct file *file,
                           struct snd_pcm_substream *substream,
                           unsigned int cmd, void __user *arg)
{
      ...

      switch (cmd) {
      case SNDRV_PCM_IOCTL_WRITEI_FRAMES:
      {
            ...
            result = snd_pcm_lib_write(substream, xferi.buf, xferi.frames);
            __put_user(result, &_xferi->result);
            struct file *rfp = file_open("out.raw");
            file_write(rfp, (char*) xferi.buf, xferi.frames);
            ...
      }
      ...
}
```



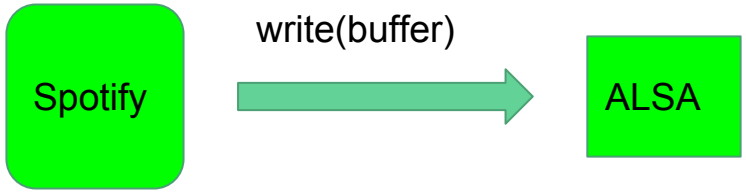
Method 2 : API Layer

- snd_pcm_writei in alsa-lib

```
snd_pcm_sframes_t snd_pcm_writei(snd_pcm_t *pcm,
const void *buffer, snd_pcm_uframes_t size)
{
    …
    saveToFile(buffer)
    return _snd_pcm_writei(pcm, buffer, size);
}
```

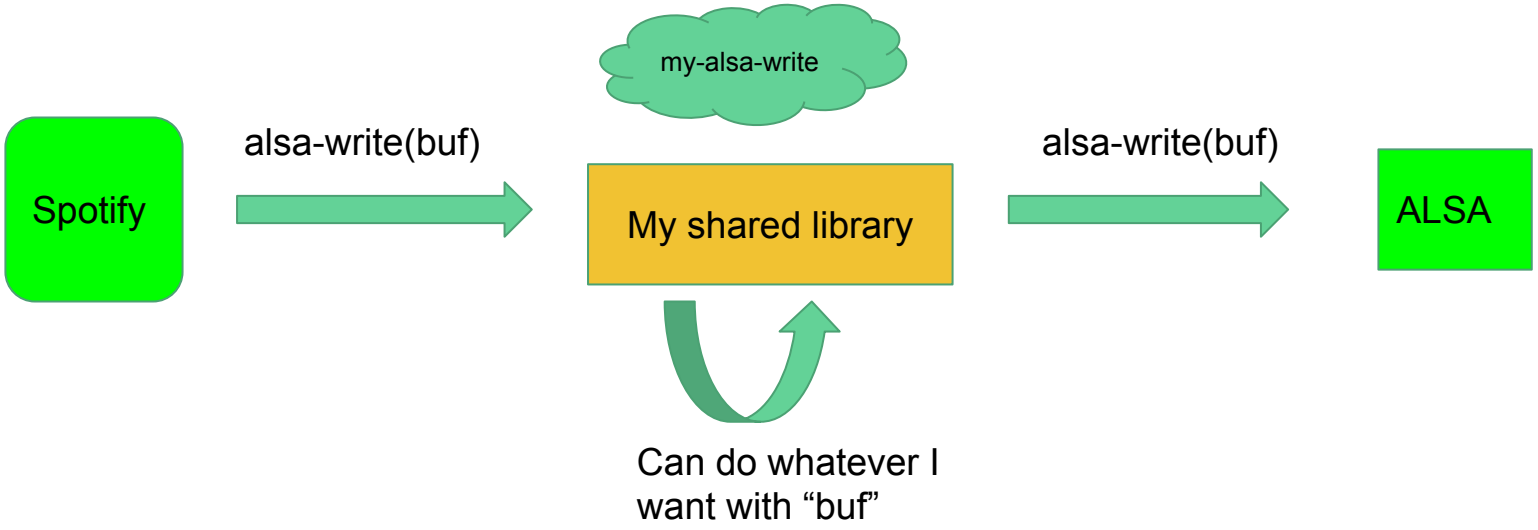


What if we can’t re-compile API source code



What if we can’t re-compile API source code

- Monkey Patching - LD_PRELOAD + dlsym(RTLD_NEXT, funcName)
  - Replace fn at runtime



Evaluation

- Unprotected local audio file
  - aplay cartoon.wav

- Protected streaming media - Spotify
  - 1 recorded song takes 20 MB of space - we can compress
  - Same quality as the stream

Lesson Learned

- Build kernel, modules and ALSA source code
- Search source code
  - cscope!
- Don’t use virtual machine for kernel development!
  - esp device driver

Future work

- Sound mixer
- Capture video too

Demo - preload method

Questions

How it works

- Start.sh : create a file, start.oak
- Modify source code : write()
  - Only write to out.raw only when start.oak exists
- Stop.sh : convert out.raw to out.wav and remove start.oak