2 iPod Antiforensics

by Travis Goodspeed

In my lecture introducing Active Disk Antiforensics at 29C3, I presented tricks for emulating a disk with self defense features using the Facedancer board. This brief article will show you how to build your own antiforensics disk out of an iPod by patching the Rockbox framework.

To quickly summarize that lecture: (1) USB Mass Storage is just a wrapper for SCSI. We can implement these protocols and make our own disks. (2) A legitimate host will follow the filesystem and partition data structure, while a malicious host—that is to say, a forensics investigator's workstation—will read the disk image from beginning to end. There are other ways to distinguish hosts, but this one is the easiest and has fewest false positives. (3) By overwriting its contents as it is being imaged, a disk can destroy whatever evidence or information the forensics investigator wishes to obtain.

There are, of course, exceptions to the above rules. Some high-end imaging software will image a disk backward from the last sector toward the first. A law-enforcement forensics lab will never mount a volume before imaging it, but an amateur or a lab less concerned with a clean prosecution might just copy the protected files out of the volume.

Finally, there is the risk that an antiforensics disk might be identified as such by a forensics investigator. The disk's security relies upon the forensics technician triggering the erasure, and it won't be sufficient if the technician knows to work around the defenses. For example, he could revert to the recovery ROM or read the disk directly.

2.1 Patching Rockbox

Rockbox exposes its hard disk to the host through USB Mass Storage, where handler functions implement each of the different SCSI commands needed for that protocol. To add antiforensics, it is necessary only to hook two of those functions: READ(10) and WRITE(10).

In `firmware/usbstack/usb_storage.c` of the Rockbox source code, blocks are read in two places. The first of these is in handle_scsi(), near the SCSI_READ_10 case. At the end of this case, you should see a call to send_and_read_next(), which is the second function that must be patched.

In *both* of these, it is necessary to add code to both (1) observe incoming requests for illegal traffic and (2) overwrite sectors as they are requested after the disk has detected tampering. Because of code duplication, you will find that some data leaks out through send_and_read_next() if you only patch handle_scsi(). (If these function names mean nothing to you, then you do not have the Rockbox code open, and you won't get much out of this article, now will you? Open the damn code!)

On an iPod, there will never be any legitimate reads over USB to the firmware partition. For our PoC, let's trigger self-destruction when that region is read. As this is just a PoC, this patch will provide nonsense replies to reads instead of destroying the data. Also, the hardcoded values might be specific to the 2048-byte sector devices, such as the more recent iPod Video hardware.

The following code should be placed in the SCSI_READ_10 case of handle_scsi(). `tamperdetected` is a static bool that ought to be declared earlier in `usb_storage.c`. The same code should go into the send_and_read_next() function.

```
//These sectors are for 2048-byte sectors.
//Multiply by 4 for devices with 512-byte sectors.
if(cur_cmd.sector>=10000 && cur_cmd.sector<48000)
  tamperdetected=true;

//This is the legitimate read.
cur_cmd.last_result = storage_read_sectors(
  IF_MD2(cur_cmd.lun,) cur_cmd.sector,
  MIN(READ_BUFFER_SIZE/SECTOR_SIZE, cur_cmd.count),
  cur_cmd.data[cur_cmd.data_select]
);

//Here, we wipe the buffer to demo antiforensics.
if(tamperdetected){
  for(i=0;i<READ_BUFFER_SIZE;i++)
    cur_cmd.data[cur_cmd.data_select][i]=0xFF;
  //Clobber the buffer for testing.
  strcpy(cur_cmd.data[cur_cmd.data_select],
         "Never gonna let you down.");

  //Comment the following to make a harmless demo.
  //This writes the buffer back to the disk,
  //eliminating any of the old contents.
  if(cur_cmd.sector>=48195)
    storage_write_sectors(
         IF_MD2(cur_cmd.lun,)
         cur_cmd.sector,
         MIN(WRITE_BUFFER_SIZE/SECTOR_SIZE, cur_cmd.count),
         cur_cmd.data[cur_cmd.data_select]);
}
```

2.2 Shut up and play the single!

Neighbors who are too damned lazy to read this article and implement their own patches can grab my Rockbox patches from `https://github.com/travisgoodspeed/`.

2.3 Bypassing Antiforensics

This sort of an antiforensics disk can be most easily bypassed by placing the iPod into Disk Mode, which can be done by a series of key presses. For example, the iPod Video is placed into Disk Mode by holding the Select and Menu buttons to reboot, then holding Select and Play/Pause to enter Disk Mode. Be sure that the device is at least partially charged, or it will continue to reboot. Another, surer method, is to remove the disk from the iPod and read it manually.

Further, this PoC does not erase evidence of its own existence. A full and proper implementation ought to replace the firmware partition at the beginning of the disk with a clean Rockbox build of the same revision and also expand later partitions to fill the disk.

2.4 Neighborly Greetings

Kind thanks are due to The Grugq and Int80 for their work on traditional antiforensics of filesystems and file formats. Thanks are also due to Scott Moulton for discretely correcting a few of my false assumptions about real-world forensics.

Thanks are also due to my coauthors on an as-yet-unpublished paper which predates all of my active antiforensics work but is being held up by the usual academic nonsense.