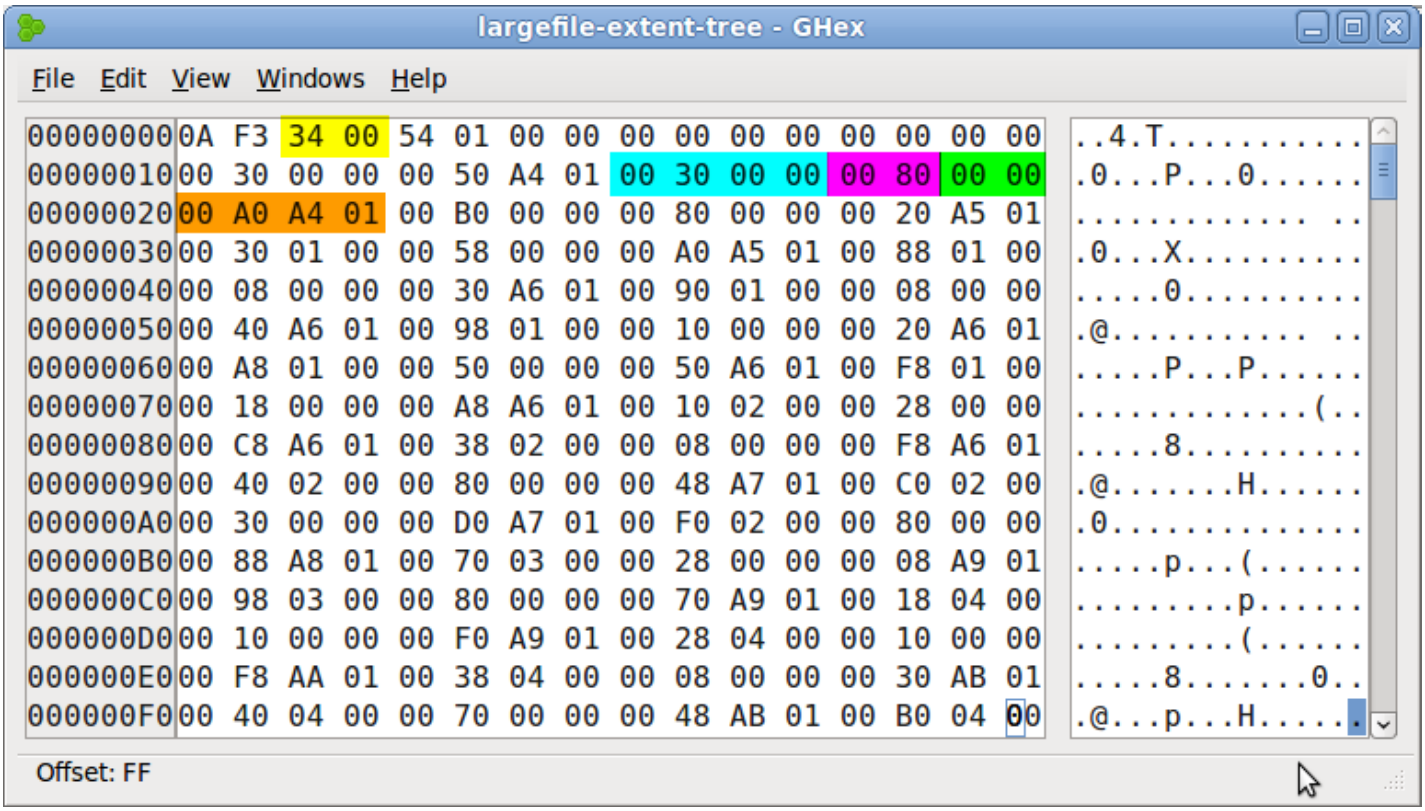
Understanding EXT4 (Part 5): Large Extents

Hal Pomeranz, Deer Run Associates

I've received a lot of positive feedback from the forensics community about this series of articles, but what's really rewarding is when other forensics researchers teach me something I didn't know. I recently received an email from a colleague in Europe who was looking at the extent trees for a large file in his EXT4 file system and saw something he couldn't explain.

To replicate the finding I created a large file-- about 4GB in size. Recall from our discussion in Part 1 of this series that there is a 16-bit field to store the size of an extent. However, the high bit in that field is reserved to mark a preallocated extent, so you can only have 32K blocks in an extent. Assuming a typical 4K block size, that means you can only have 128MB of data in a single extent. A 4GB file is therefore going to require at least 32 extents, and even that assumes you can find 32 runs of 32K contiguous blocks to use. More likely we'll have more than 32 extents, some of which don't use the full 128MB length.

After creating my 4GB file, I used the techniques described in Part 3 to decode the extent tree structure for the file and find the data block that was holding the actual extents for the file:



In fact, if you look at the number of extents field from the extent header (highlighted in yellow above) you can see that the file actually uses 52 (0x0034) extents. But what's really interesting is the second extent structure that I've highlighted above. Decoding this structure we have an extent that starts at logical offset 0x00003000 (block 12288 from the start of the file) and physical block 0x0000 01A4A000 (block number 27566080).

The thing that really surprised my colleague, however, is the extent size-- 0x8000. In binary, that's a 16-bit value with the high bit set and the lower 15 bits all zeros. Because the high-bit is used by EXT4 to mark a preallocated extent, that would mean a preallocated extent with zero bytes. And that makes no sense at all. So what's really going on here?

It's Easier When Somebody Else Does the Legwork

I received the initial email about this issue literally the day before I had to go to SANSFIRE to teach, so I wasn't able to do any research on the problem immediately. While I was dancing around in front of my students, however, my colleague in Europe was flexing his Google kung fu and found a couple of interesting links that seemed related to the behavior we were seeing.

The first was a short note in the EXT4 developers' conference call minutes:

Amit will first be merging in Andreas' patch to fallocate, which allows initialized extents to be the full 32768 blocks. Uninitialized extents are limited to 32767 blocks. Amit will also add comments to this, and have the update patches ready by tomorrow.

The second link was what appears to be the code/comments referenced in the note above, specifically:

```
-#define EXT_MAX_LEN ((1UL
```