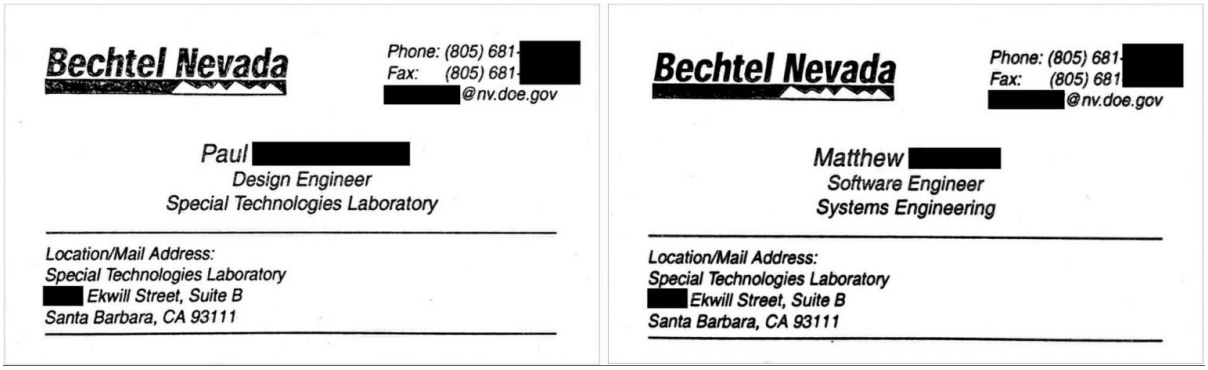
The Case of the Top Secret iPod - TidBITS

11-14 minutes

It was a gray day in late 2005. I was sitting at my desk, writing code for the next year's iPod. Without knocking, the director of iPod Software—my boss's boss—abruptly entered and closed the door behind him. He cut to the chase. "I have a special assignment for you. Your boss doesn't know about it. You'll help two engineers from the US Department of Energy build a special iPod. Report only to me."

The next day, the receptionist called to tell me that two men were waiting in the lobby. I went downstairs to meet Paul and Matthew, the engineers who would actually build this custom iPod. I'd love to say they wore dark glasses and trench coats and were glancing in window reflections to make sure they hadn't been tailed, but they were perfectly normal thirty-something engineers. I signed them in, and we went to a conference room to talk.



They didn't actually work for the Department of Energy; they worked for a division of Bechtel, a large US defense contractor to the Department of Energy. They wanted to add some custom hardware to an iPod and record data from this custom hardware to the iPod's disk in a way that couldn't be easily detected. But it still had to look and work like a normal iPod.

They'd do all the work. My job was to provide any help they needed from Apple.

I learned that an official at the Department of Energy had contacted Apple's senior vice president of Hardware, requesting the company's help in making custom modified iPods. The senior VP passed the request down to the vice president of the iPod Division, who delegated it to the director of iPod Software, who came to see me. My boss was told I was working on a special project and not to ask questions.

Background

I was the second software engineer hired for the iPod project when it started in 2001. Apple Marketing hadn't yet come up with the name iPod; the product was known by the code name P68. The first software engineer later became the director of iPod Software, the guy who gave me this special assignment. I wrote the iPod's file system and later the SQLite database that tracked all the songs. Over time, I worked on almost every part of the iPod software, except the audio codecs that converted MP3 and AAC files into audio.

(Those audio codecs were written by two engineers with advanced degrees from Berkeley and Stanford. When they weren't teasing each other about which school was better, they were writing mathematical audio code that I was scared to touch. You would no more let a regular engineer mess with code like that than you'd let a bike mechanic rebuild the transmission in a Porsche. They had an occasional poker game I played in. The only reason I didn't lose all my money was that one of them enjoyed his vodka.)

Compiling the iPod operating system from source code, loading it onto an iPod, and testing and debugging it was a fairly complex process. When a new engineer started, we typically gave them a

week to learn all this before we assigned them any actual tasks.

The iPod operating system wasn't based on another Apple operating system like Classic Mac OS, or Darwin, the underlying Unix core of macOS, iOS, iPadOS, watchOS, and tvOS. The original iPod hardware was based on a reference platform Apple bought from a company called Portal Player. Portal Player had also provided the lower levels of the iPod OS, including power management, disk drivers, and the realtime kernel (which Portal Player had licensed from another company called Quadros). Apple bought the higher levels of the iPod OS from Pixo, a company started a few years earlier by ex-Apple engineers trying to write a general-purpose cell phone operating system to sell to mobile phone companies like Nokia and Ericsson. Pixo code handled the user interface, Unicode text handling (important for localization), memory management, and event processing. Of course, Apple engineers modified all this code, and over time, rewrote much of it.

iPod OS was written in C++. Since it didn't support third-party apps, there was no external documentation on how it worked.

Finally, the iPod team developed on Windows computers. Apple didn't have working ARM developer tools yet, because this was before the iPhone shipped. The iPod team used ARM developer tools from ARM Ltd, which ran only on Windows and Linux.

My job was to get Paul and Matthew up and running on a new operating system they'd never seen before, much less developed for.

Getting Started

I requisitioned an empty office for Paul and Matthew in our building. I had IS&T (Apple's IT department) reroute the Ethernet drops in that office so they connected only to the public Internet, outside Apple's firewall, preventing them from accessing Apple's internal network. Apple's Wi-Fi network always connects outside the firewall. Even inside Apple buildings, if you're using Wi-Fi, you need a VPN to get past Apple's firewall. This wasn't a collaboration with Bechtel with a contract and payment; it was Apple doing a favor under the table for the Department of Energy. But access for that favor went only so far.

Needless to say, Paul and Matthew weren't allowed to access our source code server directly. Instead, I gave them a copy of the current source code on a DVD and explained it couldn't leave the building. Ultimately, they were allowed to keep the modified copy of the iPod OS they built, but not the source code for it.

Apple didn't provide them any hardware or software tools. I gave them the specs for the Windows computers they needed, along with the ARM compiler and JTAG debugger. They bought retail iPods to work on, several dozen at least, possibly many more.

As with all Apple buildings, everyone had to present an Apple badge to the badge reader to unlock the door and enter the iPod building. Only employees cleared for our building were allowed in. On each floor, there was another locked door and badge reader, and only people cleared for that floor were allowed in.

So every day, Paul and Matthew called me from the lobby since they didn't have Apple badges. I signed them in as guests and escorted them to their office. Eventually, I arranged to get them vendor badges, as if they were selling Apple coffee or memory chips, so I didn't have to sign them in daily. I was a programmer, not a babysitter.

Top Men

Paul and Matthew were smart—top men, even—and with a little help, they were up and running pretty quickly. I showed them how to set up the development tools, build a copy of the operating system from source, and load it into the iPod. We made some temporary changes to the user interface, so we could see that their build was actually running. I showed them how to use the JTAG hardware debugger, which was rather finicky. They dove into their work.

As they learned their way around the system, they explained what they wanted to do, at least in broad strokes. They had added special hardware to the iPod, which generated data they wanted to record secretly. They were careful to make sure I never saw the hardware, and I never did.

We discussed the best way to hide the data they recorded. As a disk engineer, I suggested they make another partition on the disk to store their data. That way, even if someone plugged the modified iPod

into a Mac or PC, iTunes would treat it as a normal iPod, and it would look like a normal iPod in the Mac Finder or Windows Explorer. They liked that, and a hidden partition it was.

Next, they wanted a simple way to start and stop recording. We picked the deepest preferences menu path and added an innocuous-sounding menu to the end. I helped them hook this up inside the code, which was rather non-obvious. In all other respects, the device functioned as a normal iPod.

At the time, the latest iPod was the fifth-generation iPod, better known as the “iPod with video.” It was relatively easy to pop open the case and close it again without leaving obvious marks, unlike the iPod nano models that became popular shortly after. Plus, the fifth-generation iPod had a 60 GB disk, so there was plenty of room to have lots of songs and still record extra data. And it was the last iPod for which Apple didn’t digitally sign the operating system.

That was important because it made the fifth-generation iPod somewhat hackable. Hobbyists enjoyed getting Linux to run on iPods, which was hard to do without the special knowledge and tools Apple possessed. We on the iPod engineering team were impressed. But Apple corporate didn’t like it. Starting with the iPod nano, the operating system was signed with a digital signature to block the Linux hackers (and others). The boot ROM checked the digital signature before loading the operating system; if it didn’t match, it wouldn’t boot.

I don’t think Paul and Matthew ever asked Apple about signing their custom operating system build so it would run on the iPod nano. I’m pretty sure Apple would have refused. The larger fifth-generation iPod was better suited to their purposes anyway.

After a few months of on-again, off-again work in their requisitioned office, Paul and Matthew finished integrating their custom hardware into the iPod and wrapped up the project. They moved their computers and debugging hardware back to Bechtel’s office in Santa Barbara. They returned the latest DVD with Apple source code to me, along with their Apple vendor badges. They said goodbye, and I never saw them again. The DVD sat on a shelf in my office for years, until I finally tossed it while cleaning up.

What Were They Doing?

The Department of Energy is huge. Its 2005 budget was $24.3 billion. It's responsible for the US nuclear weapons and nuclear power programs, including the Los Alamos National Laboratory, which was part of the Manhattan Project. As the DOE's budget request says:

> The FY 2005 budget proposes $9.0 billion to meet defense-related objectives. The budget request maintains commitments to the nuclear deterrence requirements of the Administration's Nuclear Posture Review and continues to fund an aggressive strategy to mitigate the threat of weapons of mass destruction.

My guess is that Paul and Matthew were building something like a stealth Geiger counter. Something that DOE agents could use without furtively hiding it. Something that looked innocuous, that played music, and functioned exactly like a normal iPod. You could walk around a city, casually listening to your tunes, while recording evidence of radioactivity—scanning for smuggled or stolen uranium, for instance, or evidence of a dirty bomb development program—with no chance that the press or public would get wind of what was happening. Like all other electronic gadgets, Geiger counters have gotten smaller and cheaper, and I was amused to run across the Radiation Alert Monitor 200, which looks an awful lot like a classic iPod.

Whenever I asked Paul and Matthew what they were building, they changed the subject and started arguing about where to go for lunch. Standard geeks.

The Custom iPod That Never Existed

Only four people at Apple knew about this secret project. Me, the director of iPod Software, the vice president of the iPod Division, and the senior vice president of Hardware. None of us still work at Apple. There was no paper trail. All communication was in person.

If you asked Apple about the custom iPod project and got past the stock “No comment,” the PR people would tell you honestly that Apple has no record of any such project.

But now you know.