Free Software or: How I Learned to Stop Worrying and Love the Computer

by David Sterry, TryGNULinux.com. Edited by Ali Gunduz and Ken Simeon.

In a world running on software, you have an important choice between having more control over your computer or acceptance of the status quo. It is a choice between contributing to a growing community of hopeful free software users or settling for a life of product activation, vendor lock-in, and increasing developer power.

First, a bit about why I'm writing this. I recently read 'Free as in Freedom', a book that not only described the events that led to the founding of the Free Software Foundation but also invited me to think more deeply about my own interest in free software.

I am not Richard Stallman. I don't have a beard and I don't travel the world speaking about free software. I've written enough code to know how powerful software can be but I don't write code as my primary occupation.

Still, I value my independence and wish to control my computer rather than have it control me. This is why I use free software that gives me the right to use it for any purpose, to make modifications, and to share my changes with others.

I've witnessed a lot of delays, cost and pain due to proprietary software. Beyond the usual issues of getting systems working, proprietary software presented additional and unnecessary challenges. Some of these had to do with the developer's business model, while many challenges arose from software licenses that didn't grant the freedoms to study and to modify the programs. It is for this reason that I recommend proprietary software be replaced with free software in all cases.

You can get started right now by installing a 100% free operating system like gNewSense or Trisquel both of which are distributions of GNU/Linux that include only free software. Should you choose to read on, the rest of this essay will list and elaborate on the essential ways non-free software harms you.

1. Proprietary software causes you stress

Did you ever know someone who was great at doing something but never revealed how they did it? "A magician never reveals their secret" but then again you wouldn't hire a magician to do your billing or accounting. For those tasks, you'd like to know precisely how the job is getting done. You can ask your accountant to document her billing process and then read the procedure which is better than repeating the work yourself to check the result. With most non-free software you don't have access to the source code and cannot know how the job(even billing!) is getting done. Transparency in managing your photos and browsing the web are no less important.

Psychologists have measured increased stress levels in people who had little control over a situation. Like backseat drivers, users of proprietary software have little control when a problem occurs. When a mission-critical system goes down or a show-stopping bug is triggered, stress is amplified by not knowing how a program works. Users of free software on the other hand share the virtual driver's seat. They can do testing and code review to avoid disaster or their own debugging after the fact.

2. Proprietary software doesn't respect you

To keep track of each license, proprietary software developers usually issue product keys. A product key can range from a simple serial number to several different groups of letters, dashes and numbers. They serve two purposes for the vendor and little if any purpose for users. For the vendor, requiring a product key means it's harder to share software and maintain privacy. Secondly, uniquely identifying each copy means there are more options later for detecting misuse or unauthorized copying of a program. Users instead have a responsibility to keep track of their product keys, protect them from unauthorized use, and enter them in whenever a non-free program asks.

The fact that product keys can be easily copied meant developers needed something more. The next step is product activation during which your computer contacts the developer and creates a record of an installation. Since product activation requires communication with the developer it introduces an unnecessary requirement to use the program: an Internet connection or access to a phone. The time and resources required to activate can vary but again, this unnecessary process is not productive for users. There are also serious privacy concerns with activation. Though most activation processes promise to respect your privacy, it is hard to verify such a claim. When data is sent securely by a non-free program, only the developer can know precisely what data is being sent. To use a proprietary program that uses activation, users must absolutely trust the developer.

Again, activation is not enough for vendors to stop their software from being shared. Popular programs are available with the activation code stripped out of them and so was created a new and abusive practice called automatic auditing. The most widely-known program of this sort is Windows Genuine Advantage (WGA), which Microsoft requires users to run for access to the latest security and bug fixes. What Microsoft tells us is that WGA checks the operating system, product key and other information about the computer to determine if the operating system is properly licensed. But how can we be sure? Just as with activation, data that can identify your machine is sent to Microsoft. Besides the obvious problems with non-free software developers having free reign over your programs and personal data, automatic auditing can also habituate users to invasive business practices.

Windows Genuine Advantage attempts to define software in terms more fitting for physical goods. Calling genuine software advantageous incorrectly implies that a copy of a software program is inferior to the original. It is meaningless therefore to speak of whether a program is genuine or not. If this was Microsoft's concern, they could have released a tool to determine bit-for-bit integrity without the "genuine" terminology.

The other problem with Windows Genuine Advantage is the suggestion that there should only be one developer for a piece of software. It is an understandable attempt to strengthen, in users minds, Microsoft's proprietary position. Among most free software users, a community of developers with diverse interests is seen as a strength. Within the community, the best version of a program may be distributed by different developers depending on user requirements. If beauty is in the eye of the beholder, improvement is in the eye of the user. Through automatic auditing, non-free software developers are attempting to habituate users to unreasonable search and to a distorted view of a healthy development community.

Lastly, in all but the smallest organizations that use proprietary software, one must manage copies of a non-free program by running up a licensing server. The IT department must absorb additional overhead and introduce a potential point of failure, all because the developer has chosen to restrict use of the program. Again, the complication of a licensing server and possible delays associated with it serve no productive purpose for you. They are just more reasons to switch away from proprietary software. Free software respects your rights as a user and has none of the aforementioned problems.

3. Proprietary software strips you of the power to help.

I've encountered some situations where the restrictive license of a program meant I couldn't be as helpful as I would have liked. These situations are the Reinstall CD Problem (RCDP), the Needed-Software Dilemma (NSD), and the Rare Technical Issue (RTI).

The RCDP (Reinstall CD Problem) is common with Microsoft Windows since it is distributed widely and in many versions. When replacing a failed hard drive, repairing a badly infected computer, or fixing a difficult OS-level problem, having the reinstall CD may be required and if you don't have it, you have an RCDP. This may mean waiting a couple days for a replacement or purchasing the operating system all over again. The restrictions placed on the use and distribution of non-free software means that an RCDP can last for days or weeks and require $30 - $300 in shipping or licensing costs. Contrast this with free software operating systems that can be downloaded and installed as fast as your Internet connection allows.

A closely related problem is the NSD (Needed-Software Dilemma). Often after setting up a new computer or reinstalling an operating system, I am asked, "What about (popular non-free software package)?" I must then ask the user to locate the installation media and any product keys. If the user can't find them, the situation becomes uncomfortable. Knowing the legal ramifications of distributing unlicensed copies of a proprietary program , I can't share my copy. I am put in a dilemma of whether to help the user and violate the software license or to obey the law and refuse helping. This is the perfect time to recommend free software.

Free or non-free, if you use a program long enough you will run into a bug. If you're lucky it's a common problem and the fix is documented somewhere. In the case of the RTI (Rare-Technical Issue), however, there is no documented fix. If you are short of time, you have no choice but to go straight to the developer. They may charge a support fee, be slow to respond, or roll the fix into an upgrade for which you must purchase an additional license. Without the source code and permission to modify the program, your options are limited. With free software, you'll almost always find the answer to the problem online and if not, you can submit a bug report to fix the RTI. If it doesn't get fixed, you may fix it yourself or hire someone. Often, I can submit a question to a forum and work with others to document a workaround within a day or two.

4. Proprietary software forces you to settle.

Another problem appears when you need a new or improved feature. While similar to the Rare Technical Issue, this issue escalates when your goals differ from the developer's. Some developers may see a feature request as a path to increased customer satisfaction, but others may decide that the feature isn't widely needed or that it will require too much testing and would therefore not be profitable. Free software gives you the option to make the improvements you need without the drama of an abusive user-developer relationship.

Nobody wants to be feel helpless, to have increased stress, or have their freedom taken away. I care about free software because I value not just my own time and independence but that of my friends, family and customers. If you feel the same, try a 100% free software operating system, try a replacement for your billing software, or communicate with your proprietary software vendor about why they don't publish their software under a free software license. You have a choice.