Global Information Assurance Certification Paper

Is Open Source Software Really More Secure?

Bud Rogers

January 21, 2000

We have all heard the claim that Open Source software is by its nature more secure. Open Source advocates claim that access to source code allows anyone to look for bugs or malicious code, so problems tend to be identified and fixed much more quickly than in proprietary closed source software. Many of us take for granted the truth of these assertions. Is our faith in Open Source justified?

The arguments in favor of open source are many and varied, but most revolve around the concept of peer review. Open source advocates maintain that if source code is made public and anyone can examine it, the odds are much greater that security flaws will be promptly identified and fixed. The idea is perhaps best summarized by Eric S. Raymond's famous phrase "Given enough eyes, all bugs are shallow." [1]

In contrast, users of proprietary software are totally dependent on the vendor both to identify vulnerabilities and to fix them. While the vendor has some motivation to find and fix bugs, there are often much stronger incentives to enhance their software in other ways that are more immediately visible to the customer. If the choice is between squashing an obscure bug and providing new features, features win almost every time.

Peter G. Neumann, Chairman of the ACM Committee on Computers and Public Policy, cites some of the risks associated with closed source proprietary software:

- Unavailability of source code reduces on-site adaptability and repairability
- Inscrutability of code prohibits open peer analysis (which otherwise might improve reliability and security).
- Lack of interoperability and composability often induces inflexible monolithic solutions.
- Proprietary interface standards complicate system integration.

He goes on to say that

> "In critical applications, an enormous amount of untrustworthy code may have to be taken on faith." and that "Open-source software offers an opportunity to surmount these risks of proprietary software.[2]

The President's Information Technology Advisory Committee noted a "growing national vulnerability based on the inadequacies of the current system to build reliable and secure software" and went on to recommend the Open Source software development model. [3]

The Information Technology Advisory Agency of the German Interior Ministry has proposed that German federal offices adopt open source software standards as a matter of policy. The proposal cites, among other things, security issues relating to proprietary data formats, interoperability, and source code access. The proposal raises the specific question, "Is Open-Source Software secure?" It concludes that

> "Making the software open source alone does not mean per se that it is secure [but that] the fundamental precondition for evaluating the security of software is certainly for its source code to be made open."[4]

Three French Members of Parliament have proposed a law requiring open standards and access to source code as a matter of policy by the French government. They cite similar concerns about proprietary data formats, interoperability, and accessibility to source code. [5]

In the specialized area of cryptographic software, independent peer review has long been recognized as an essential element of the development process. Bruce Schneier flatly states that peer review is the only way to guarantee the reliability of cryptographic algorithms:

> "The only way to find security flaws in a piece of code is to evaluate it . . . And the best way to facilitate that is to publish the source code."[6]

Open source software written to open, published standards also facilitates interoperability with other software. Both the German and French policy proposals as well as the PITAC Letter to the President mention this point. Interoperability is not a security issue in itself, but it can have security implications. Many proprietary software packages have interfaces which limit interoperability to a vendor's own software suite or to a narrow range of other software products. If that limited range of choice does not meet the users' security requirements, then it becomes a security issue.

There is also the issue of features in proprietary software, included by the vendors themselves, which raise security concerns. Both Microsoft and Real Networks raised the ire of privacy groups when it was discovered that they were using Globally Unique Identifiers

built into Windows software, to collect potentially sensitive private information about users. Each company first denied and later admitted that they were building large databases of information tied to individual users and PC's through their GUID's. Microsoft maintains the GUID is meant to help support personnel track software problems and that they were unaware of the larger privacy issues. [7] Real Networks claimed the GUID's had been included in their software inadvertently and besides, they weren't using the information for anything anyway.

> "David Brotherton, a spokesperson for Real Networks, said the company never used the information for anything. 'We weren't even aware it was there,' he said." [8]

Protestations of innocence aside, the very presence of GUID's in proprietary software raises real privacy and security issues. The Microsoft and Real Networks flaps are two widely publicized examples of undocumented "features" which expose potentially sensitive information without users' knowledge or consent, but they are by no means the only such cases. The French proposal of law specifically mentions this concern as a national security issue:

> "In order to guarantee national security, it is required to use systems which do not contain pieces of code which may be used to take remote control of a computer or to unwillingly transmit information to a third party."[9]

Critics of open source argue that access to source code makes it easier for attackers to identify vulnerabilities. [10] The problem with this argument is that attackers seem to be able to find vulnerabilities in proprietary software about as well as they do in open source software.[11]

Open source critics also argue that open source can lead to a false sense of security. They say that just because the source code is available doesn't guarantee that anyone is reading it. [12] Nor does it mean that all the bugs have been found and fixed. Many users install and use open source software without ever looking at the code. They assume someone else has already scanned it for possible vulnerabilities. Undetected bugs have lingered in some popular open source packages for years. [13] This is a legitimate concern.

Critics of open source also point out that the average user may not be competent to inspect source code and identify vulnerabilities. [14] Particularly in the case of large, complex software packages, this can be a real concern. Bruce Schneier makes this point with respect to cryptographic software. Even while making the case for open source crypto, he cautions that the code needs to be examined by expert cryptoanalysts:

> "And you can't just have anyone evaluate the code, you need experts in security software evaluating the code." [15]

Another concern is the growing popularity, even within the open source movement, of packages that include precompiled binaries. Users who install precompiled binaries have no guarantee that the executables have any direct relationship to the published source code. They are essentially placing their trust in the good will and good security practices of the site from which they downloaded the packages. In that respect they are in essentially the same position as users of proprietary software. Only if they compile from source themselves can they be assured that the executable they get matches the source code. This is also a legitimate concern.

Another concern raised by critics of open source is the issue of liability and risk management. Users of proprietary software purchased from reputable vendors feel they have some recourse in the event of failure. The most common Open Source license, the GNU General Public License, explicitly disavows any warranty. [16] Users basically assume all the risk of software failure on their own. For some users, particularly business users, that may not be an acceptable risk. [17]

There does not seem to be very much solid evidence to settle the argument. Among the few known attempts to quantify the issue are a pair of studies done by the University of Wisconsin. The first was done in 1990. A test program called fuzz was used to feed random input to a number of common UNIX utilities to test their susceptibility to input buffer overflows. The study found that many of the utilities in common use on UNIX systems of the day were indeed vulnerable. [18]

The study was repeated in 1995. The second study found that many then-current versions of the same utilities still exhibited the same vulnerabilities. The commercial UNIX distributions exhibited roughly the same vulnerabilities seen in the earlier study, while versions of the same utilities packaged with distributions of Linux were somewhat less vulnerable. The Open Source GNU versions of the same utilities, while not perfect, were generally the least vulnerable of all.[19] While not conclusive proof, the studies seem to lend credence to the claims of Open Source advocates.

SecurityPortal.com did a study which attempted to compare response time to security vulnerabilities between Open Source and proprietary operating systems. The study measured the time between general awareness of a security flaw and the availability of a patch. They compared response times between Redhat Linux, Microsoft NT, and Sun Microsystems Solaris. They found that the Open Source OS was generally able to provide security patches faster that either of the commercial vendors.[20]

Can we draw any conclusions from any of this? Everyone seems to have an opinion on Open Source. There are plenty of arguments, both pro and con, and almost everyone has a favorite story to prove their point.

Proponents argue that access to source code allows anyone to examine the code. With lots of eyes looking at the code, bugs are likely to be quickly found and fixed. Open source software written to open standards facilitates interoperability. Users can pursue a strategy of mix and match to find the best combination of software for their particular needs.

Critics contend that access to the source code makes it easier for attackers to find vulnerabilities. They argue that having access to the source code doesn't mean anyone is actually looking at it, or that all the bugs have been found and fixed. They remind us that open source users who install precompiled binaries are essentially running closed source software since they can't be sure the executables match the source code. And they raise the issue of accountability -- when the software breaks, who is responsible?

Certainly publishing source code for independent peer review increases the likelihood that security flaws will be identified and fixed in a timely manner. And the counter argument, "security through obscurity," has been pretty well debunked. Access to source code is probably a good first step toward better security.

But access to source code does not by itself guarantee better security. Someone has to actually look at the code to find and fix the bugs. The first part of Eric Raymond's famous phrase is after all, "Given enough eyes." And a certain level of competence is required as well. A million clueless newbies reading source code they don't understand probably won't do much for security. Thousands of competent coders carefully reviewing each others' work can probably improve security for all.

---

1 Eric S.Raymond, "The Cathedral and The Bazaar ", URL: http://www.tuxedo.org/~esr/writings/cathedral-bazaar/

2 Peter G.Neumann, "Robust Open-Source Software", Communications of the ACM, 41:2 (February, 1998): 128

3 Raj Reddy and Irving Wladawsky-Berger, "President's Information Technology Advisory Committee Letter to the President", URL: http://www.ccic.gov/ac/pitac_ltr_sep11.html

4 Rudolf E. Bahr, Ralf Reilander, and Egon Troles "Open-Source Software in the Federal Administration", KBSt Letter No. 2/2000, February 24, 2000, URL: http://linux.kbst.bund.de/02-2000/brief2-2000-en.pdf

5 Jean-Yves Le Deaut et al, "A Proposal of Law", URL: http://www.osslaw.org/motifs_en.html

6 Bruce Schneier, Secrets & Lies, (New York: John Wiley & Sons, 2000), 344

7 John Markoff, "Microsoft to Alter Software in Response to Privacy Concerns", URL: http://www.nytimes.com/library/tech/99/03/biztech/articles/07soft.html

8 Bob Sullivan, "More privacy concerns for Real", URL: http://www.msnbc.com/news/436070.asp

9 Jean-Yves Le Deaut et al, Ibid.

10 Rudolf Schreiner, "Open Source Software Security", URL: http://www.objectsecurity.com/whitepapers/open_source/open_source_security.html

11 Jay Beale and Kurt Seigfried, "Open Source - Why it's Good for Security", URL: http://www.securityportal.com/topnews/os20000417.html

12 Elias Levy, "Wide Open Source", URL: http://www.securityfocus.com/commentary/19

13 Bruce Schneier, Ibid. 345

14 Elias Levy, Ibid.

15 Bruce Schneier, Ibid. 344

16 GNU GENERAL PUBLIC LICENSE, URL: http://www.gnu.org/copyleft/gpl.txt

17 Scott Berinato, "Experts debate the merits of opens source for security", URL: http://www.zdnet.co.uk/news/2000/12/ns-14378.html

18 B. P. Miller et al, "An Empirical Study of the Reliability of UNIX Utilities", Communications of the ACM, 33:12 (December 1990), 32-44

19 B. Miller et al, "Fuzz revisited: A re-examination of the reliability of unix utilities and services". Technical report, Computer Sciences Department, University of Wisconsin, 1995.

20 "Linux vs Microsoft: Who solves security problems faster?", URL: http://securityportal.com/cover/coverstory20000117.html

Last Updated: May 28th, 2017

Upcoming Training

SANS Stockholm 2017	Stockholm, Sweden	May 29, 2017 - Jun 03, 2017	Live Event
SANS San Francisco Summer 2017	San Francisco, CA	Jun 05, 2017 - Jun 10, 2017	Live Event
Security Operations Center Summit & Training	Washington, DC	Jun 05, 2017 - Jun 12, 2017	Live Event
SANS Houston 2017	Houston, TX	Jun 05, 2017 - Jun 10, 2017	Live Event
Community SANS Ottawa SEC401	Ottawa, ON	Jun 05, 2017 - Jun 10, 2017	Community SANS
SANS Rocky Mountain 2017	Denver, CO	Jun 12, 2017 - Jun 17, 2017	Live Event
SANS Charlotte 2017	Charlotte, NC	Jun 12, 2017 - Jun 17, 2017	Live Event
SANS Rocky Mountain 2017 - SEC401: Security Essentials Bootcamp Style	Denver, CO	Jun 12, 2017 - Jun 17, 2017	vLive
Community SANS Portland SEC401	Portland, OR	Jun 12, 2017 - Jun 17, 2017	Community SANS
SANS Secure Europe 2017	Amsterdam, Netherlands	Jun 12, 2017 - Jun 20, 2017	Live Event
SANS Minneapolis 2017	Minneapolis, MN	Jun 19, 2017 - Jun 24, 2017	Live Event
SANS Columbia, MD 2017	Columbia, MD	Jun 26, 2017 - Jul 01, 2017	Live Event
SANS Cyber Defence Canberra 2017	Canberra, Australia	Jun 26, 2017 - Jul 08, 2017	Live Event
SANS Paris 2017	Paris, France	Jun 26, 2017 - Jul 01, 2017	Live Event
SANS London July 2017	London, United Kingdom	Jul 03, 2017 - Jul 08, 2017	Live Event
Cyber Defence Japan 2017	Tokyo, Japan	Jul 05, 2017 - Jul 15, 2017	Live Event
SANS Cyber Defence Singapore 2017	Singapore, Singapore	Jul 10, 2017 - Jul 15, 2017	Live Event
Community SANS Minneapolis SEC401	Minneapolis, MN	Jul 10, 2017 - Jul 15, 2017	Community SANS
SANS Los Angeles - Long Beach 2017	Long Beach, CA	Jul 10, 2017 - Jul 15, 2017	Live Event
Community SANS Phoenix SEC401	Phoenix, AZ	Jul 10, 2017 - Jul 15, 2017	Community SANS
SANS Munich Summer 2017	Munich, Germany	Jul 10, 2017 - Jul 15, 2017	Live Event
Mentor Session - SEC401	Macon, GA	Jul 12, 2017 - Aug 23, 2017	Mentor
Mentor Session - SEC401	Ventura, CA	Jul 12, 2017 - Sep 13, 2017	Mentor
Community SANS Atlanta SEC401	Atlanta, GA	Jul 17, 2017 - Jul 22, 2017	Community SANS
Community SANS Colorado Springs SEC401	Colorado Springs, CO	Jul 17, 2017 - Jul 22, 2017	Community SANS
SANSFIRE 2017	Washington, DC	Jul 22, 2017 - Jul 29, 2017	Live Event
SANSFIRE 2017 - SEC401: Security Essentials Bootcamp Style	Washington, DC	Jul 24, 2017 - Jul 29, 2017	vLive
Community SANS Charleston SEC401	Charleston, SC	Jul 24, 2017 - Jul 29, 2017	Community SANS
Community SANS Fort Lauderdale SEC401	Fort Lauderdale, FL	Jul 31, 2017 - Aug 05, 2017	Community SANS
SANS San Antonio 2017	San Antonio, TX	Aug 06, 2017 - Aug 11, 2017	Live Event
SANS Prague 2017	Prague, Czech Republic	Aug 07, 2017 - Aug 12, 2017	Live Event