

Which Platform: The Myths and Realities of Software Security

Goh Seow Hiong*
June 2005

In today's software industry, few arguments have generated as lively a debate as the virtues of open source software versus their commercial, or proprietary, counterparts.

While the debate makes for entertaining reading, certain myths and misconceptions have been perpetuated in the process. To help decision makers discern the facts, this article will take a clear-eyed look at some key aspects of the debate, and attempt to separate fact from fiction.

A good starting point is the issue of software security, as it is an industry-wide concern. We will see below that vulnerabilities affect all complex software programs and are not more or less prevalent for software developed under either an open source or commercial software model. The issue is in how to minimize and remedy the vulnerabilities, not which licensing model leads to more secure software. Criminal attacks against software remain a critical consideration in the minds of all users, and it is therefore important to understand the underlying principles in this area.

Myth #1: Known Security Mechanism Means More Secure

It is sometimes argued that a computer program whose source code is not disclosed cannot, by definition, be secure. This view also holds that if the source code of a product is made available, the product is, by definition, more secure. But is this, in fact, the case?

Good security is not dependent on whether the mechanism (in this case, source code) is known or published, but on how the mechanism is designed, implemented and managed. More specifically, security depends on whether qualified persons have reviewed and tested the mechanism to minimize the number of vulnerabilities that might be exploited by criminals.

While the source code in certain cases may be available for people to read, in reality, few code reviewers or writers are trained or skilled in identifying security vulnerabilities. A good programmer does not necessarily make a good security specialist.

In the open source community, where large volumes of source code are available, it is not always possible for every single line of the code to be scrutinized by a wide range of security experts. Users who obtain open source packages often install and use the software even before taking a look at the source code or recompiling it, even though it may have been provided along with the software.

Just as there are vulnerabilities in proprietary, or commercial, software, so too are there vulnerabilities in community-developed open source software, some remaining undiscovered for years notwithstanding the availability of source code.

Myth #2: Speed of Security Fix

The speed through which vulnerabilities and bugs are addressed and remedied is also a point of contention among proponents of different software development models.

Some have assumed that open source software developers respond more quickly to vulnerabilities or bugs than commercial software providers. However, comparative studies of the security of products indicate that neither development model necessarily resolves vulnerabilities faster than the other.

Moreover, the assumption that a quick fix is a good fix may not always be the case. For some customers, it is important that updates or patches for identified vulnerabilities be thoroughly tested in different environments and configurations

before they are rolled out, lest they lead to system stability issues.

For others, it is essential for the customer to have accountability from the software provider for the quality of the updates or patches provided. Such customers may not want just any third party to provide an update or patch, if that third party does not bear ultimate responsibility for the success of the fix, or the problems that may arise from applying the fix.

The Reality: Implementing Security

For years, many developers focused on coding software for scalability, availability, manageability and serviceability. In recent years, new and increasingly sophisticated criminal attacks have caused security to factor significantly more into the work of developers, resulting in better and more secure code.

The issue is not what kind of software is being developed – commercial versus open source – but rather, is security being implemented at every stage of the development process or simply as an after-thought? Today we see software licensed under open source and commercial models that has been developed with security in mind. Developers are using methodologies known to reduce vulnerabilities through up front assessments, rigorous and organized testing, and post release response centers that assess vulnerabilities and provide updates. How the software is licensed is not a significant factor in this process.

While the design of security features matters significantly, total security depends just as much on how well the software is deployed, configured, updated and maintained, including whether product vulnerabilities are discovered and resolved through appropriate updates.

Many security compromises arise from the lack of proper configuration of the systems and software deployed, and the lack of a timely application of security updates, patches and fixes. The user's role in ensuring that the computing environment is

properly configured and maintained cannot be under-emphasized.

Lapses from configuration errors or poor system maintenance are by far the more common mistakes and causes of security breaches than malicious attacks resulting from known vulnerabilities. The *impact* of software exploits also has more to do with the popularity of the software than with its design or method of development.

Neither Is Inherently Superior

Given these considerations, it is wrong to assume that either open source or commercial software is inherently superior in security. Conclusions about the security of a product should not be made on the basis of its software development model or its licensing model.

Instead, the three key determinants of software security are the quality of the developers, the techniques and tools used by the development team to reduce vulnerabilities, and the strength of the relationship between the customer and the software provider in addressing problems that arise. None of these factors are directly dependent or pre-determined by the method of software licensing or distribution.

Software developers across the industry are redoubling their efforts to address consumer concerns on security. Ultimately, good code is good code, regardless of whether the source is open or not.

* Mr GOH Seow Hiong (shgoh@bsa.org) is the Director of Software Policy for Asia of the Business Software Alliance (www.bsa.org). BSA is the voice of the world's commercial software industry and its hardware partners before governments and in the international marketplace. Its members represent one of the fastest growing industries in the world. BSA programs foster technology innovation through education and policy initiatives that promote copyright protection, cyber security, trade and e-commerce. BSA members include Adobe, Apple, Autodesk, Avid, Bentley Systems, Borland, Cadence Design Systems, Cisco Systems, CNC Software/Mastercam, Dell, Entrust, HP, IBM, Intel, Internet Security Systems, Macromedia, McAfee, Microsoft, PTC, RSA Security, SAP, SolidWorks, Sybase, Symantec, UGS and VERITAS Software.