Vulnerability Centric Exploitation Attempts in Open Source Software Systems

Sajad Mohammed Khan
Scientist B
P. G. Department of Computer Sciences
University of Kashmir

Abstract - There's been a lot of debate by security practitioners about the impact of open source approaches on security. One of the key issues is that open source exposes the source code to examination by everyone, both the attackers and defenders, and reasonable people disagree about the ultimate impact of this situation. It's been argued that a system without source code is more secure because, since there is less information available for an attacker, hence it should be harder for an attacker to find the vulnerabilities. For open source software, security attention frequently focuses on the discovery of vulnerabilities prior to release since source code can be viewed freely and users can identify, fix and make the general public aware of the vulnerabilities and thus in many people's minds, is the idea that, this makes the system more secure. In this paper we investigate this common misconception and reveal the security vulnerabilities in open source software.

Keywords- Security practitioners, open source, source code, vulnerability.

Introduction- The current world economy relies heavily on computerized information systems and security problems threaten this infrastructure. While attention focuses on issues such as system development, from our perspective, security remains important. The security issues need to be put under consideration during the development phase of the software. One important choice software developers make is the availability of the source code. Developers can choose two basic options for software source code visibility; closed source or open source. In closed source software, the developers do not make the source code publicly visible while as in open source software, the source code is publicly available for viewing. Many proponents of open source software believe that by making the source code available, the software product can be more secure. They reason that the more the people who view the code, the more likely that vulnerabilities are discovered before the software is released, which is a myth. This is opposed to proprietary software scenario where a program crashes and possibly provides a small amount of debugging information. Unless the user is prepared to reverse engineer the software in an attempt to understand how the product works or inform the vendor, detailing the vulnerability and leaving the vendor to choose how to resolve the issue.
The reality

However, despite development and testing efforts, not all vulnerabilities in software will be found before release. Instead, some will inevitably be found afterwards. While considerable research and commentary has focused on the pre-release stage benefits of open source software, the post-release stage is important as well. Generally attackers (against both open and closed programs) start by knowing about the general kinds of security problems programs have. And then use techniques to try to find those problems. We categorize the attacking techniques into two groups:

1. Dynamic techniques (where the program is run and checked for vulnerabilities)
2. Static techniques (where the program's code is examined - be it source code or machine code).

In dynamic approach, an attacker runs the program, sending it data (often problematic data), and sees if the programs' response indicates a common vulnerability. Open and closed programs have no difference here, since the attacker isn't looking at code.

In static approach, for open source software, attackers probably look at the source code and search it for patterns. For closed source software, they might search the machine code for essentially the same patterns which is a bit difficult. They might also use tools called decompilers that turn the machine code back into source code and then search the source code for the vulnerable patterns (the same way they would search for vulnerabilities in open source software) but the decompilers never produce the exact copy of the source code which gives the closed source an edge over the open source.

The Risk of Exploitation Attempt

Open source software may increase the risk of exploitation attempts based on vulnerability. Since attackers can view the source code, their effort and time to develop an exploit is reduced for open source software. Because their effort is reduced, the expected value of exploitation is strictly increased because of reduced cost. Similarly, because the time is reduced, the expected value is increased because of increased probability of exploit success. In general innovation competition, information and time advantage is important. In the context of security, the time advantage means that attackers will find fewer systems patched and defended against the vulnerability. Faced with limited resources, the rational attacker will focus on opportunities with the highest expected value of exploitation first. Therefore, we hypothesize that:

“A target firm will face a greater risk of exploitation for vulnerabilities in open source software than for vulnerabilities in closed source software.”

The Diffusion of Exploitation Attempts

Like other innovation, exploitation knowledge diffuses through the attacker community. After vulnerabilities are discovered, expert attackers build tools to exploit the vulnerability. As these tools diffuse through the attacker community, more and more organizations will experience the exploitation attempt. Then, as the potential targets install countermeasures, the value of the tool will diminish; new tools based on new vulnerabilities will offer more reward. The reduced time and effort for attackers to develop exploits translates to earlier diffusion.
Volume of Exploitation Attempts

Another metric to evaluate the relative impact of the open versus closed source choice is the overall volume of exploitation attempts. Vulnerabilities are typically found by expert attackers; they are then quickly incorporated into automated tools usable by anyone. While expert discovery will be seen in the risk of first exploitation analysis, tools are responsible for the volume of exploitation attempts. Open source code affects tools from both a supply and demand side. First, expert attackers invest their time and effort into tools that the attacker community will find most valuable. Tool value comes from increase probability of successful attack. Because the source code is visible, expert attackers require less effort and time to create tools; this time reduction increases the probability of successful attack. Therefore, more tools will be available. Second, adopters of the tools are similarly rational. They will adopt and spend effort on tools with the greatest expected value. The compounded result is greater adoption of a greater number of tools; these tools increase the number of exploitation attempts that an organization will experience.

Kernel Implants

Another serious security problem that is made worse by open source is the growing trend of operating system kernel implants. It used to be the case that if malicious code was running on a machine, generally an experienced system administrator using diagnostic tools and techniques could detect its presence. However with an open source operating system such as Linux the hackers have all the information required to develop implants to the kernel of OS. If malicious code is running at that level then it is extremely difficult to detect.

Conclusion

Our Analytical results indicate that, saying open source software is more secure than closed source software based on the argument that, more the people who view the code, the more likely that vulnerabilities are discovered before the software is released, is a myth. Thus compared with closed source software, vulnerabilities in open source software:
(a) Have increased risk of exploitation,
(b) Diffuse sooner and with higher total penetration, and
(c) Increase the volume of exploitation attempts.
(d) Have scope to get malicious code get undetected.

References


