BLACKDUCK

OPEN SOURCE SECURITY ANALYSIS
The State of Open Source Security in Commercial Applications

By Mike Pittenger, *Vice President, Security Strategy*

Black Duck's On-Demand business conducts audits of customers' software, often in merger or acquisition situations. Typically the audits include commercial software that has been in the market for a number of years.

From a legal standpoint, customers want to confirm their software is not subject to unnecessary intellectual property (IP) risk through the use of open source software under restrictive licenses (e.g., GPL, LGPL), or from improper reporting of the open source licenses used.

From a security standpoint, customers want to understand the security profile of their software. While many of these companies have internal security programs and deploy security testing tools such as static and dynamic analysis, those tools are not effective at identifying the types of vulnerabilities disclosed every day in popular open source components. More importantly, if a customer is not aware of all of the open source in use, they cannot defend against common attacks against known vulnerabilities in those components.

This study covers more than 200 applications reviewed by Black Duck On-Demand over the six months from October 2015 through March 2016. The age of the applications tested (e.g., how old the codebase is) varies widely. All of the companies submitting code for audit review had conducted manual reviews, and all data was anonymized prior to Black Duck's analysis.

YOU'RE USING OPEN SOURCE, AND MORE THAN YOU THINK

Everyone uses open source. Black Duck finds it in over 95% of the applications we analyze for clients. It's easy to understand why. Open source adds needed functionality while lowering development costs and accelerating time to market.

Open source can enter a code base in a variety of ways. We commonly think of a developer who recognizes a need for specific functionality, and pulls in an open source component that meets the re-

105
Average number of open source components in each application

quirements. While this represents the classic example, open source enters in other ways as well. Commercial components typically include open source that may or may not be disclosed. Additionally, outsourced development teams are highly motivated to use open source for lowering development costs and speeding time to market. In other cases, open source is built into reusable components that are used internally.

2x On average the companies were using 100% more open source than they originally believed

Our review found that open source comprises over 35% of the average commercial application, and represents over 100 unique open source components in each application. When considering these numbers, it is important to remember that we are reviewing commercial applications as opposed to code developed for internal use. In the latter category, we expect to see open source comprising a much higher percentage of the application (75%+ is not unusual), though with a smaller total number of components.

If the number of unique components is surprising to a reader, it is also surprising to our customers. Those who provide a listing of the components (bill of material) they expect to be in the applications when the audit begins are often only aware of 45% of the actual components used. In other words, while customers may believe they are using (on average) 60-70 components, they are actually using over 140.

67% of applications reviewed contained open source security vulnerabilities

IF YOU'RE USING OPEN SOURCE, CHANCES ARE YOU ARE LIKELY INCLUDING VULNERABILITIES KNOWN TO THE WORLD AT LARGE

Without visibility into the open source they use, a company is unable to protect itself from known vulnerabilities in those components. Even when components are identified, it can be difficult to track the various projects and comb publicly available databases for changes to the code, including newly disclosed vulnerabilities. Since 2014 alone, the National Vulnerability Database (NVD) has reported over 6,000 new vulnerabilities in open source software.

Do the math. If the average commercial application tested includes over 100 unique open source components, manual tracking of the components in a single application is clearly burdensome. Multiply that by the hundreds or thousands of applications in a large enterprise, and the ability to track risk manually is impossible.

THIS ISN'T AN ISOLATED INCIDENT

While a single component with an exploitable vulnerability can be a problem, the issue is more widespread according to our data. On average, we identified five vulnerable components in every application.

Of course, this doesn't mean customers should stop using open source. It does indicate, however, that visibility into the components that are included in their code base is required. This would provide the ability to switch to newer (or at least less vulnerable) versions of the same components.

22.5

Average number of open source component vulnerabilities in each application

A single vulnerability may or may not warrant an organization's attention. It could be minor or unreachable by an attacker. What we found, however, goes far beyond a single vulnerability in a single component. On average, each vulnerable component included multiple, unique vulnerabilities. When we look at the total number of vulnerabilities per project, the numbers are daunting – over 22 individual vulnerabilities in any single application.

1,894 days

Average age of open source component vulnerabilities at scan time

BAD NEWS, THIS ISN'T A NEW PROBLEM

When a security issue is disclosed in an open source component, it is often (but not always) accompanied by an update or patch that remediates the issue. This allows development teams to address the vulnerability by updating the component. With some components, of course, this is not a simple fix. Those with multiple APIs will require more planning and testing to ensure the fix doesn't break other functionality or add new vulnerabilities.

The planning process still doesn't explain the age of vulnerabilities found in our study. On average, the vulnerabilities we identified had been disclosed more than five years before our analysis. This indicates that the organizations didn't know about the vulnerabilities, either because they didn't know the component was present, or had not checked public resources for vulnerability information.

This represents a significant risk to organizations deploying these applications. The longer a vulnerability is known, the more likely that an attacker can leverage it. And attackers often move quickly. The 2015 Verizon Data Breach Investigation Report found that "half of the CVE's (Commonly Vulnerabilities and Exposures) exploited in 2014 fell within two weeks", and "99.9% of the exploited vulnerabilities were compromised more than a year after the CVE was published".

In other words, you need to know where vulnerabilities exist in your code (and the open source in your custom code) and add this to your incident-response program.

EVEN WELL-PUBLICIZED VULNERABILITIES ARE NOT GETTING FIXED

Vulnerabilities in open source are particularly attractive to attackers. The ubiquity of the affected components, the public disclosure of vulnerabilities (often with sample exploits) and access to the source code make the attacker's job simpler. In addition, without a traditional support model, users are typically unaware of new updates and vulnerabilities.

10%
of the applications included the Heartbleed vulnerability

Some vulnerabilities garner a lot of news, while others fly under the radar. However, when something like Heartbleed is talked about in the mainstream media, including the nightly news, one would expect companies to take note.

Our study found over 10% of the applications tested included the Heartbleed vulnerability (disclosed a minimum of 18 months prior to our analysis), and almost 10% included POODLE. LogJam and FREAK each affected almost 5% of the applications.

This illustrates the difficultly organizations have in managing open source components, and hence the attractiveness to attackers of vulnerabilities in these components. Without a comprehensive list of the components used, it is nearly impossible to map new vulnerabilities to specific applications that use the affected components. With popular components like OpenSSL, attackers see a target-rich environment when new vulnerabilities (and associated exploits) are disclosed.

NOR IS BAD NEWS SOMETHING YOU CAN SAFELY IGNORE

% of High Severity (CVSS Base Score >= 7.0	39.55%
% Medium Severity (CVSS Base Score >=4, <=6.9	52.10%
% Low Severity	8.35%

40%
of open source vulnerabilities in each application were rated "severe"

As noted previously, vulnerabilities vary in severity and will warrant different responses from an organization. The analysis tell us, again, that awareness of the vulnerabilities was low (or non-existent). Common Vulnerability Scoring System (CVSS) base scores (scored 1-10) are calculated by looking a combination of the simplicity of exploiting the vulnerability and its impact to confidentiality, integrity, and availability for the component. Our study found almost 40% of the vulnerabilities had CVSS base scores greater than 7, and over 90% had base scores greater than 4.

CONCLUSION

Open source lowers development costs while accelerating time to market, and we expect its adoption to continue to grow. This report highlights the fact that, even for companies that conducted manual reviews, unknown (to the companies) open source permeated commercial applications.

Vulnerabilities in open source are particularly attractive to attackers. The ubiquity of the affected components provides a target-rich environment; the vulnerabilities are publicly disclosed (often with sample exploits); and without a traditional support model, users are typically unaware of new updates and vulnerabilities.

It's obvious that it is impossible to defend against a threat that you don't know exists. Organizations can address this with three simple steps:

1. CREATE OPEN SOURCE USAGE POLICIES: Understand the characteristics that are important to your organization for each type of application you build, including license obligations, acceptable security risk and open source community support.

2. TRACK USAGE AND ENFORCE POLICIES: Automated tools like Black Duck Hub automatically identify and track open source through integration with build tools, highlighting known vulnerabilities and components that violate company policy. This inventory, or bill of materials, is updated with every build and can be used to map new vulnerabilities to your applications.

3. MONITOR FOR NEW VULNERABILITIES: Public sources, like the National Vulnerability Database, provide information on publicly disclosed vulnerabilities in open source and commercial software.

OSSA 2016

ABOUT BLACK DUCK SOFTWARE

Organizations worldwide use Black Duck Software's industry-leading products to automate the processes of securing and managing open source software, eliminating the pain related to security vulnerabilities, open source license compliance and operational risk. Black Duck is headquartered in Burlington, MA, and has offices in San Jose, CA, London, Frankfurt, Hong Kong, Tokyo, Seoul and Beijing. For more information, visit www.blackducksoftware.com.

CONTACT

To learn more, please contact: sales@blackducksoftware.com or +1 781.891.5100
Additional information is available at: www.blackducksoftware.com