

White Paper

Undocumented Open Source Leaves a Gap in Your Application Security Strategy



PALAMIDA[™]

Application Security for Open Source Software

215 Second Street, First Floor
San Francisco, California 94105
415.777.9400

www.palamida.com

Shifting the Security Focus to the Open Source Software in Your Application Layer

Executive Overview

Applications are more susceptible than ever in today's dynamic application development landscape. Most applications, internal and external, developed within the last five years, include at least 30% open source software and third-party components. And by 2010, open source products will be well established in 75% or more of mainstream enterprises.¹ While important to a company's bottom line, this increase in open source usage presents a huge security challenge to organizations industry-wide.

The root cause of many application security vulnerabilities lies in the application source code. The problem is that the sheer size of a code base, coupled with the number of contributing developers, makes it nearly impossible for organizations to get an accurate assessment of their software assets, much less a clear understanding of the vulnerabilities associated with the adopted code.

- What do you have?
- Where is it across your global code base?
- Where did it come from?
- What are its vulnerability risks?

Security has permeated its original boundaries of being a perimeter issue only and spread to an organization's most important asset, its software applications.

Open Source is Not Risky – Undocumented Open Source is Risky

The risks of open source software lie not in the usage of open source, but in the management *or lack thereof* of the open source itself. If a company does not keep track of the open source it has adopted, then it can be difficult and costly to identify vulnerabilities and implement patch releases associated with the adopted code.

While commercial software suppliers have put in place mechanisms by which to notify customers of updates and push them out automatically to licensed users, many open source projects do not. This puts the burden of responsibility on the *user* to monitor whether there are security patches or newer versions of the open source project available.

When organizations do not know what code is running inside their applications, they are leaving themselves open to serious vulnerabilities that can ultimately lead to data breaches. A recent Forrester study reported the cost of a data breach to run anywhere between \$90-\$305 dollars per record.²

Safeguarding Against Vulnerabilities in Undocumented Open Source

In order to effectively manage code vulnerability and compliance challenges, organizations must:

- Mitigate Risk: Secure your software procurement process at the development level.
- Create Accountability: Develop an audit trail for compliance checks.
- Continuously Monitor: Review open source inventory and evaluate vulnerability information.

Application security means much more than just firewalls and virus scanning. It requires code-level protection against security, governance and intellectual property risks and solutions robust enough to identify intellectual property challenges and known vulnerabilities in the code base.

Why Application Security?

The investment global organizations make in developing and maintaining software applications is significant. Forrester Research predicts that in 2008, organizations in North America and Europe will spend at least \$181 billion in new software application development and development for upgrades and maintenance.³ This investment in software applications indicates the growing importance and reliance on software applications and their value to a company's overall capitalization. In the mid-1980's only 20% to 30% of a company's capitalization was comprised of intangible assets, including intellectual property found in software applications. Now, 70% to 80% of its capitalization is made up of these intangible assets.⁴

Software applications help meet customer and competitive needs, but they also provide a primary avenue for attackers to evade traditional network barriers. These applications, particularly those that are externally-facing and Web-based, represent a significant opportunity and risk to every organization. According to research by Gartner Group and Symantec, close to 90% of software attacks are aimed at the application layer.⁵ Once application vulnerability has been exploited, a company is at risk not only for potential loss of vital customer or company data, but may even be open to additional attacks against other systems within the company's network.

Company	Issue	Estimated Cost ¹
Children's Hospital	In November of 2006, the hospital was victim to overseas hackers who were able to gain access two computers at the hospital, one containing private patient data, the other billing and bank information of over 200,000 patients.	Up \$61 million
Kingston Technology	In September of 2005, the computer memory vendor experienced a security breach, believed to be a result of malware attached to undocumented code. The breach may have compromised the names, addresses and credit card details of roughly 27,000 online customers.	\$2.4 million
TD Ameritrade	The names of essentially all customers have been compromised by an intruder, brokerage TD Ameritrade announced in July of 2007. The discovery was made during an investigation into stock-related spam, during which 6,500,000 credit card records compromised.	Up to \$1 billion
United States Pentagon	The Pentagon announced in April, 2006 that more than 14,000 people who registered for a 2001 Defense Department conference on health care fraud whose names, Social Security numbers, credit card information and other personal information may have been stolen due to a breach of security of the computer system of Tricare Management System, a government health-care contractor.	Up to \$2.1 million

Data Breaches: The Impact of Unsecure Code

Software security vulnerabilities are often caused by defective specification, design, and implementation. It is becoming very clear that decisions made during the software development lifecycle – from user interface design to embedded third-party components to patch management - will significantly impact the likelihood of security incidents and the success of responding to them.

It is the responsibility of security, development and IT teams to ensure that their developers use processes that produce secure software. Working together, these three groups need to insert application security into the overall security strategy for their organizations by:

- Conducting code-level security reviews, in addition to penetration tests, for their internally-developed code before deployment
- Insisting that code-level audits have been conducted by outsourced development and business partners
- Ensuring that all other third-party code included in their software applications is identified and tracked for security flaws and updated version information
- Ensuring that internally-developed applications have adequate checkpoints that enable thorough audit trails

In Deloitte's 2007 Global Security Survey, "The Shifting Security Paradigm," key finding #3 points out that "to remain competitive, companies must mitigate software security risks when they develop, acquire, outsource or host software applications."

The entire survey can be downloaded at www.deloitte.com/dtt/research/0,1002,sid=1013&cid=170582,00.html

This means that development organizations must continue to acquire the high level of security expertise required, identify processes for producing secure software, adopt them, and consistently use them when they produce, enhance, maintain, and rework the software that supports a strong application infrastructure.

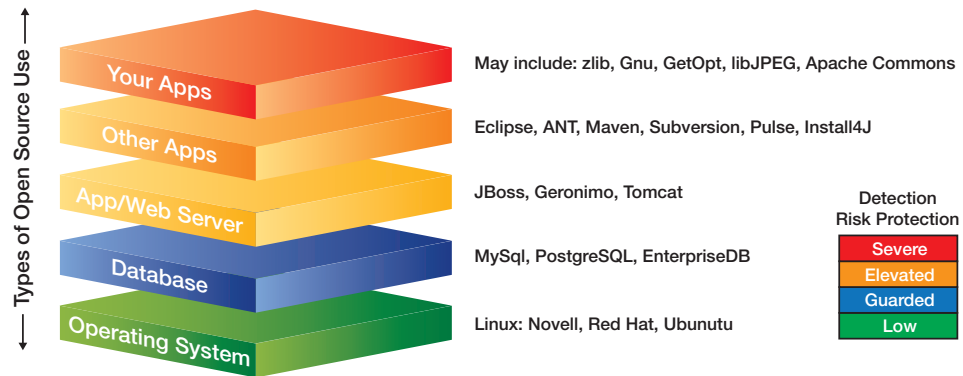
Open Source 101: It's Not Just Linux

IDC has called the use of open source the most significant, all-encompassing and longer trend that the software industry has seen since the early 1980's. In their 2006 report, "Open Source in Global Software: Market Impact," IDC Research found that open source was being used by 71% of worldwide developers and was in production at 54% of their organizations. Further illustrating this trend, a 2007 report by Gartner Research predicts that by 2008, 95% of Global 2000 organizations will have formal open source acquisition management strategies in place to address the challenges and opportunities of open source.

Undocumented Open Source Leaves a Gap in Your Application Security Strategy

The benefits of open source are readily apparent – it reduces cost of development, shortens development cycles, and can lower overall total cost of ownership of your applications if managed well. And, open source means more than just Linux and other commonly-used packaged OSS enterprise applications (e.g. SugarCRM, Pentaho, Open Office). Open source is being used up and down the application stack. While some very well-known and popular open source technology is supported by a strong commercial services community (e.g. Linux distributed by Novell, Red Hat, and Ubuntu; SQL servers such as MySQL and EnterpriseDB; and JBoss from Red Hat), only 1 out of every 10 open source projects have a commercial services community supporting it.⁶

As the use of open source has broadened behind corporate firewalls, its use requires effective management of the newly-emergent and informal “software supply chain” through which open source regularly enters the development environment. Ten to twenty years ago, when developers wanted to incorporate third-party code into the applications they were building, a joint development agreement or in-bound licensing contract would be negotiated, and the process would have included at least a development manager, procurement lead, and a lawyer. In today’s world of 24/7 and persistent network access, developers dispersed across multi-national sites can include open source, freeware, public domain, eval-ware of commercial software, etc. into the code they are writing, without triggering the usual check-points in the procurement process. And, the further up the application stack open source software is in use, the less likely its use is detected, monitored and tracked.



Open Source Software Use in the Enterprise

The result is that most software applications written in the last five years consist of at least 50% open source software code, by a line of code count. And a majority of that 50% is undocumented by the company.⁷

Size	60M lines of code
Open Source Disclosed Pre-Audit	303
Additional Undocumented Open Source Components Discovered	535
Breakdown of Code	60% open source vs. %40 proprietary code

Sample Report from Recent Global 1000 Company Audit

And as engineering departments adopt richer interactive environments (such as the use of Ajax), they become even more susceptible to vulnerabilities (i.e. Cross-site Scripting) which can be easily introduced by their developers via OSS and ultimately exploited by outsiders. As such, companies should be ensuring that they continue to leverage their use of open source technology but ensure that they have a transparent authorization and monitoring system that enables them to minimize version proliferation and ensure that they are using the most secure releases.

Project Name	Description	Common Usage
curl	curl is a command line tool for transferring files with URL syntax, supporting FTP, FTPS, HTTP, HTTPS, SCP, SFTP, TFTP, TELNET, DICT, LDAP, LDAPS and FILE.	A network transport utility.
FFmpeg	FFmpeg is a computer program that can record, convert and stream digital audio and video in numerous formats. [1] FFmpeg is a command line tool that is composed of a collection of free and open source software libraries.	Provides multimedia support to applications.
FreeType	FreeType is a software library that implements a font rasterization engine. It is used to rasterize characters into bitmaps and provides support for other font-related operations.	Helps with font layout on web-based applications.
Glibc	Glibc is used in systems which run many different OS kernels and different hardware architectures.	Provides common programming utilities. Basis of many applications' code.
Libpng	Libpng is the official PNG reference library (originally called pnglib). It is a platform-independent library that contains C functions for handling PNG images.	Provides support for PNG images.
Libtiff (Library for reading and writing Tagged Image File Format)	Utility for processing TIFF's. It is distributed in source code and can be found (on the internet) as binary builds for all kinds of platforms.	Provides support for TIFF images.
OpenSSL	OpenSSL is an open source implementation of the SSL and TLS protocols. The core library implements the basic cryptographic functions and provides various utility functions	Provides secure connections for software.
Prototype	Prototype is a JavaScript Framework that aims to ease development of dynamic web applications.	Common Ajax-related toolkit.
Script.aculo.us	script.aculo.us is a JavaScript library built on the Prototype JavaScript Framework, providing dynamic visual effects and user interface elements via the Document Object Model.	Popular Web 2.0 enabler.
zlib	zlib is a software library used for data compression. zlib is an abstraction of the DEFLATE compression algorithm used in gzip file compression program.	Most popular compression library.

The Top 10 Most Popular Open Source You're Using that Go Undocumented⁸

Is Hidden Open Source Putting Your Company at Risk?

Open source projects benefit from the public and collaborative nature of the community of users and developers that support them. A great majority of these communities respond quickly to reported vulnerability and security concerns. Open source developer communities are quick to validate, repair and post a patch or upgraded release to a project Web site.

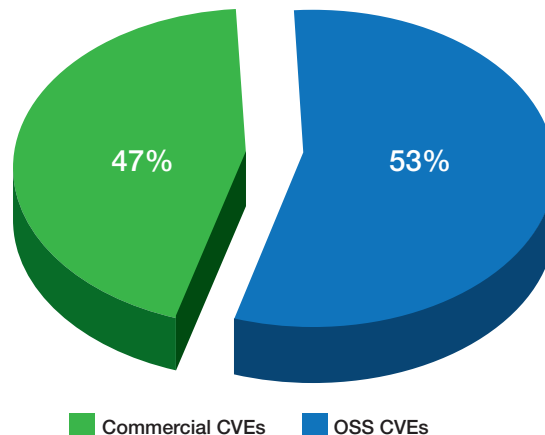
Commercially-supported open source projects and proprietary software manufacturers, on the other hand, have put in place mechanisms by which to notify customers of updates and push them out automatically to licensed users. But for most open source projects, it is up to the users to ensure that they are monitoring for the most recent versions. This puts the burden of responsibility on *organizations* to monitor whether there are security patches or newer versions of the open source project available. As such, for security and development teams, open source patch management is a crucial aspect of overall application security management.

Open Source: No More or Less Risky than Commercial Code

The National Vulnerability Database (NVD), sponsored by the Department of Homeland Security (DHS) and maintained by the MITRE Corporation, is a leading repository that tracks reported vulnerabilities, assigns standardized identification tag and provides remediation and patch information as supplied by either the software vendor or broader security community. As of end of February, 2008, the NVD referenced over 29,000 reported common vulnerabilities and exposures (CVEs) of varying severities – with approximately 14,000 related to open source projects. MITRE’s Web site describes “an information security vulnerability” as a mistake in software that can be directly used by a hacker to gain access to a system or network. The NVD considers a mistake a vulnerability if it allows an attacker to use it to violate a reasonable security policy for that system (this excludes entirely “open” security policies in which all users are trusted, or where there is no consideration of risk to the system).⁹

The NVD defines a vulnerability as a state in a computing system (or set of systems) that either:

- allows an attacker to execute commands as another user
- allows an attacker to access data that is contrary to the specified access restrictions for that data
- allows an attacker to pose as another entity
- allows an attacker to conduct a denial of service



% of CVEs on NVD Related to Open Source and Commercial Projects

The impact of the vulnerabilities in open source is equivalent to that of their commercial counterparts with CVEs ranging from Low to High Severity. Another measure is the US Department of Homeland Security open source project, which has uncovered an average of one security glitch per 1,000 lines of code in 180 widely-used open source software projects.¹⁰ Palamida recently listed the Top 5 Open Source Vulnerabilities encountered in its audit practice during 2007. The vulnerabilities are associated with the open source projects Apache Geronimo, JBoss Application Server, LibTiff, Net-SNMP and Zlib.

Project Name	CVE	Vulnerability Description
Apache Geronimo 2.0	CVE-2007-4548	Allows attackers to bypass authentication requirements and deploy arbitrary code
JBoss Application Server 3.2.4 to 4.0.5	CVE-2006-5750	Allows remote users to read or modify arbitrary files.
LibTiff before 3.8.2	CVE-2006-3464	Allows attackers ability to deploy arbitrary code
Net-SNMP 5.2.x before 5.2.2, 5.1.x before 5.1.3, 5.0.x before 5.0.10.2,	CVE-2005-4837	Allows remote attackers ability to cause denial of service
Zlib before 1.2.3	CVE-2005-2096	Allows remote attackers ability to cause denial of service

Top 5 Most Overlooked Open Source Vulnerabilities for 2007

For more information about these projects and for patch and updates, please visit www.palamida.com/blog.

Zlib and LibTiff are two projects that are good examples of the prevalence and potential impact of open source software vulnerabilities.

Prevalence and Impact of Open Source Software Vulnerabilities

The zlib library has been a fundamental open source software component for almost 15 years and can be found in almost every Linux and Unix system. The security flaw in earlier versions of the zlib impact more than just other open source projects. According to the zlib project page, there are at least 756 applications from both open source and *commercial* vendors that uses zlib. Major vendors of these applications include Adobe, Microsoft, Cisco, CA and VMware.

The impact of open source software vulnerabilities can be felt on even the latest consumer gadgets. In November 2007, Apple posted a security update version 1.1.2 for the iPhone and the iPod Touch to patch the well-known vulnerability in Libtiff.

The vulnerability leads to a security flaw that can be triggered when viewing a malicious TIFF image leading to multiple stack-based buffer overflows, which can cause denial of service and the possible execution of arbitrary code. This has led to the Jailbreakme 1.1.1 vulnerability that allows hackers root access to the iPhone to read the email and install and run applications.

The breadth and scope of open source vulnerabilities seem consistent with their commercial counterparts. New industry regulations, along with increased market awareness, are increasing the pressure within organizations to ensure that their application security policies extend to their use of open source.

Conclusion

The identification, monitoring and management of open source security vulnerabilities in order to ensure that no hidden third-party code is embedded in applications or running behind firewalls, to date has not been managed well. But with the increasing scrutiny of the problems associated with application-layer attacks, increased customer data breaches, and the need to increase the reliability of InternetWeb-facing systems, the need for automated application security and governance systems for open source and other third-party code use becomes a part of engineering, IT and application security top priorities. By implementing these new application security software systems, organizations will be able to increase customer and partner confidence, maintain competitive advantage and ensure regulatory compliance and compliance with corporate policies.

About Palamida, Inc.

Palamida is the industry's first application security solution exclusively for Open Source Software that uses component-level analysis to quickly identify and track undocumented code and associated security vulnerabilities as well as intellectual property and compliance issues, enabling development organizations to cost-effectively manage and secure mission critical applications and products.

For more information visit www.palamida.com

¹Mark Driver, Gartner Group, February 2007

²Dignan, Larry, "What that Data Breach Will Really Cost You," ZDNet Between the Lines May 2007: 8

³Forrester Research, "Business Data Services Enterprise And SMB Software Survey, North America And Europe, Q3 2007"

⁴Herrald, Heather "Getting your buck's worth from intellectual property," InfoWorld October 2001: 16

⁵Briggs, Linda L. "Application Security Comes Under Attack," Application Development Trends June 2006: 1

⁶Based on Palamida research conducted February 29, 2008 - March 4, 2008, examining support structure for 3,168 popular open source projects

⁷In 2007, Palamida Services team audited between 300M to 400M lines of code for F500 to venture-backed companies across, multiple industries. Results were tabulated based on blind audit results.

⁸"Top 5 Most Overlooked Open Source Software Vulnerabilities for 2007" by Palamida. Detailed information at www.palamida.com/blog

⁹Common Vulnerabilities and Exposures cve.mitre.org

¹⁰Broersma, Matthew, "Open Source Security Bugs Uncovered," PC World January 2008: 29