OPTIMIZING OSS MANAGEMENT TO MINIMIZE OPERATIONAL EXPOSURE IN FINANCIAL SERVICES

For banks, delivering innovation and differentiation to clients is critical. Yet many banks devote more time and resources than necessary to implementing proprietary software that delivers neither.

To combat this, financial service organizations are turning to open source software (OSS), which offers significant improvements in cost, schedule, and features. With the increased use of OSS comes greater innovation, improved code quality, and quicker deployment. It also means less code to maintain.

Gartner estimates that by 2016, OSS will be included in mission-critical software portfolios within 99 percent of Global 2000 enterprises. As OSS use increases, so does the need to track its use. Most financial institutions maintain a catalogue of OSS components but they rarely have adequate visibility into where these components are used and by whom.

It's critical for financial services organizations to understand the operational risks associated with inconsistent software component compliance, policies, workflows, and procedures. Of particular importance are the following issues:

- Transparency into OSS use
- Building the "find and fix" model
- Automatically detecting security and other vulnerabilities in OSS
- The challenge of maintaining outdated or multiple versions of OSS components
- Automating and managing the selection, use, and governance of software components, which enables developers to build better software faster
- The impact of non-compliance

OSS: SHAPING THE FUTURE OF SOFTWARE DEVELOPMENT IN FINANCIAL SERVICES

OSS has reshaped the way financial services organizations develop software. But because OSS is free, the financial services industry has had difficulty reconciling traditional, cost-based notions such as ROI with the value of OSS to the organization.

OSS presents other issues as well. It does not align with traditional measurements of cost, such as operational expenditure (OpEx) or capital expenditure (CapEx). More importantly, hidden costs – and risks – are present in OpEx costs, where a large percentage of security vulnerability-related expenditures occur. Decoding the true costs, risks, and value of OSS to the organization is critical.

Among the OpEx and CapEx questions facing financial services IT executives and enterprise architects are the following:

1. **Who uses OSS software in the organization, and what will happen if something goes wrong?** What is the potential impact if security vulnerabilities are found in a file, but no one knows who is using the code?
2. A file with a security issue is found – but **what is it?** Does it affect the front, middle, or back office? Will it affect how business is done? Is it customer facing, such as in a mobile banking system?

THE PATH OF OSS IN FINANCIAL SERVICES

OSS comes in to an organization through one of two pathways: at the bottom of the stack, as code sanctioned by the organization (the architecture review board decides to use Linux or Maven); or through the back-channel, top of the stack, as developers search the web for code to perform specific tasks (such as for mobile banking systems). This approach leaves development managers in the dark, unaware of how much OSS is being used (and where), unable to manage its use, and unable to determine its true value to the organization.

While vulnerability detection is necessary, it's not enough. An organization must also know what's in its code. The increasing adoption of open source software, coupled with insufficient management of OSS code, poses a major threat to enterprise security – and is a potential source of operational risk.

For years, financial services organizations have viewed license compliance as the greatest barrier to OSS deployment. By building catalogues of approved software and training developers in the proper use of OSS, much compliance risk has been avoided. Yet significant operational risk still exists as a new generation of developers, accustomed to using any code that comes to hand, brings unmonitored OSS into the organization. This practice is significant not only for license compliance reasons; it also represents extreme risk to the security of an organization's code. When code comes in from unmonitored sources, it greatly complicates the process of identifying where in an organization's software operations it is being used. This has significance at all stages of the software supply chain.

DETECTING SECURITY AND OTHER VULNERABILITIES IN OSS

In 2013 alone, security breaches affected Drupal.org, Adobe, MongoHQ, Target, LivingSocial, Evernote, Facebook, The New York Times, the National Security Agency and others.[1] Financial security organizations including Bank of America, JP Morgan Chase and the Federal Reserve sustained breaches. From denial-of-service attacks to third-party software hacks to SQL injections, malware, and outright theft of data, global organizations and their customers were broadly affected.

Security vulnerability detection is both a passive and active operation. Passive detection of security vulnerabilities relies on a range code scanning tools such as Black Duck, which identifies OSS code, from snippets to components, at both the binary and source code level. Black Duck seeks out code and matches it against a proven database of a million projects, matching code by version, license, and security profile. Security information, updated daily by Black Duck, is drawn from the National Vulnerability Database. It is also drawn from projects such as the OWASP Top 10, which identifies critical web application security flaws; the CWE SANS Top 25 Most Dangerous Software Errors, a list of the most widespread, critical errors that can lead to serious vulnerabilities in software; and through other sources.

While vulnerability detection is necessary, it's not enough. An organization must also know what's in its code. The increasing adoption of open source software, coupled with insufficient management of OSS code, poses a major threat to enterprise security – and is a potential source of operational risk.

MANAGING OPEN SOURCE SOFTWARE TO REDUCE OPEX – "FIND AND FIX"

Managing the selection, adoption, and use of OSS to reduce operational exposure is the missing link in many financial firms' approach to OpEx risk and security management, yet it's a relatively painless fix. The key is to achieve transparency into OSS usage by creating a "find and fix" culture, based on a documented open source strategy, backed by supporting tools and processes.

[1] http://www.crn.com/slide-shows/security/240165003/top-10-security-breaches-of-2013.htm/pgno/0/10 and others

Defining an open source software strategy begins with an articulation of the business objectives for using open source. For most financial services organizations, there are three drivers for using open source: cost avoidance, time-to-market for critical new initiatives such as mobile banking services, and developer satisfaction (including hiring, retention, and productivity).

While most firms struggle to document a strategy for using open source software, creating one can establish stakeholder consensus for proper use of OSS. In addition to a defined business rationale for OSS use, OSS strategy must include the following:

- Policies to detect existing open source in all software
- Agreement among stakeholders where open source will be used in the future
- Policies for evaluating, approving, using, and releasing open source code
- Development objectives for OSS use, including a method for calculating value and savings
- Agreement on methodologies and tools to be used

Processes also must define the way OSS is managed and monitored on a day-to-day basis. To be accepted by developers, processes and policies must be made part of the organization's standard development and release process.

Enterprise-wide adoption and use of OSS is especially important in geographically dispersed development organizations. Automated tools such as Black Duck simplify the implementation of OSS policies and processes. They also streamline OSS component acquisition and approval, component updates, vulnerability checking, software releases, and regulatory compliance.

Ideally, an OSS management tool should include:

- Approval workflows informed by OSS use policy
- Security vulnerability alerts
- License obligation management
- Version analysis and control
- Reporting (where and how OSS is used)
- Authentication and access control
- Support for creation of an approved OSS component catalogue
- Search capability to speed discovery of approved OSS components

Such a system can bridge the gap between compliance and security stakeholders and those who are striving to break productivity barriers by providing a tool that is useful for both groups.

The ability to search, select, analyze, and audit open source code empowers financial services firms to make better informed choices about which OSS to deploy. A configurable approval workflow accelerates the approval process. Access to a catalog of approved components saves time and eliminates duplicate requests and redundant effort. The ability to manage security vulnerabilities and cryptographic code ensures selection of the most secure open source components. And the ability to index code increases developer productivity and ensures ongoing compliance.

INTEGRATION WITH EXISTING PROCESSES AND TOOLS

To be useful and accepted by developers, an OSS management tool must support integration for at least two reasons: to avoid disruption of development operations, and to support use of a variety of development tools.

The need to integrate is especially important when sharing code and information with third parties. Firms need to protect their IP and to protect against security vulnerabilities that can arise with the introduction of third-party software.

Integration is crucial for firms with large, geographically dispersed development organizations; it is unwise to implement an OSS management tool that changes the way developers work. Ideally a management tool will be integrated in the background, allowing development operations to continue. This is one of the biggest barriers to manual systems for tracking OSS use. No developer wants to sit and update a spreadsheet of OSS components manually, and no compliance or security officer wants to rely on that spreadsheet. The goal of any system should be to integrate and automate OSS management in the development backend, without adding complexity.

Integration is also necessary with leading source code management tools, application lifecycle management tools, issue tracking systems, build systems, and a full range of software configuration management systems and version control systems. These tools and systems also provide the ability to create custom integrations, which help OSS management to become accepted into existing development processes. OSS management tools also should include integration with leading repository management systems.

WHAT HAPPENS IF SOMETHING GOES WRONG?

Without implementing processes, policies and tools to discover and manage OSS in the enterprise, organizations face incalculable risk and operational exposure. Consider the recent Heartbleed bug, which resulted from an error introduced into the open source Open SSL cryptographic software library. While few banks use Open SSL in, for example, mobile sign-ons, many websites and firms were affected. One source estimated the eventual cost of Heartbleed to be in excess of $500 million.

While an extreme example, Heartbleed illustrates the significant operational risk of not knowing what software is running, where it is, and who is using it.

A more likely scenario for most firms is the challenge of maintaining outdated or multiple versions of OSS components. (Ironically, many organizations were protected from Heartbleed because they used on old version of Open SSL). Having an OSS use strategy also provides important regulatory compliance capability.

COMPLIANCE ISSUES

SOFTWARE LICENSES

More than 2,200 licenses govern the use of the nearly one million OSS projects currently available. The Open Source Initiative (OSI), a non-profit that maintains the open source definition, also maintains a list of approximately 70 OSI-approved OSS licenses.

Licenses reflect a developer's intent for the use of a project. Many developers want others to use their software without restriction. These developers use permissive licenses such as BSD or Apache. Some prefer to prescribe how those who use their software may modify and/or redistribute the original code. This group may choose restrictive or copyleft licenses such as GPLv3.

Focusing on licenses as the sole source of operational risk misses other areas of concern identified by Accenture in its report, *Accelerating the Benefits of Open Source Software*, which include managing project versions, support, and integration.

REGULATORY ISSUES

Banks are subject to a host of regulations that directly or indirectly affect software used or created by the institution. These regulations include Basel II and III, Sarbanes-Oxley, and PCI-DSS. Open source is often an integral part of the applications that interact with critical financial data. So a lack of visibility into what the code is doing and how it works can represent a control oversight and create regulatory exposure. In addition, the way developers integrate open source with proprietary code can affect intellectual property ownership. As with license compliance, firms need a comprehensive strategy for the sourcing and use of OSS, along with a robust OSS management system.

QUANTIFYING OPERATIONAL RISK

Operational risk can be broken into three categories: direct risk (can it bring down the business?) manageable risk, (can we replace the software? how long will it take to replace? is it business critical?), and legal risk, discussed above.

Direct risk should be weighted by criticality (will it stop operations?), ability to build a replacement component in-house, time to replacement, and impact of removal of software, if necessary. Organizations must determine what value to assign to each level of risk.

VALUE

A value calculation for OSS requires an objective, quantifiable methodology for determining value which can be used by the CIO, the architecture review board, and development management to agree to a value 'number'. Once the method for calculating value is determined, firms can increase use of OSS to drive value throughout the software development lifecycle. For more information, visit the Black Duck website, www.blackducksoftware.com/open-source-value-assessment, to obtain your own Open Source Value Assessment.

NEXT STEPS: FREE OPEN SOURCE RISK PROFILE

Black Duck's Open Source Risk Profile provides a quick and thorough analysis of the security, operational, and legal risk associated with the OSS code used in your enterprise. To learn how your organization can benefit from a free, in-depth profile of all your open source components, along with a detailed action plan, visit www.blackducksoftware.com/osrp and sign up for your Open Source Risk Profile today.