OPEN SOURCE HYGIENE:

CRITICAL FOR APPLICATION SECURITY

With the use of open source software increasing throughout the enterprise, and with recent high-profile open source security vulnerabilities raising alarms, an essential component of an effective application security strategy is **Open Source Hygiene**.

Just as physical hygiene involves neutralizing sources of infection, Open Source Hygiene entails keeping software stacks and application portfolios free of known-exploitable versions of open source code. Implementing Open Source Hygiene requires ongoing code maintenance, informed open source technology selection, and vigilance by users and integrators of open source code.

This white paper examines the real-world need for Open Source Hygiene and the best practices and technologies for implementing it.

OPEN SOURCE – PAST, PRESENT AND FUTURE

Many in the open source community acknowledge that better Open Source Hygiene – particularly as it relates to application security – is necessary today. They said so loudly and clearly in the 2015 Future of Open Source Survey.

The comprehensive survey showed that eight out of 10 companies are running part or all of their operations on open source software, double what it was five years earlier. With a mere three percent of companies reporting no use at all, open source software is a long way from being considered "gimmicky" as was the case in the first annual Future of Open Source Survey in 2007.

hy·giene

/ˈhīˌjēn/

conditions or practices conducive to maintaining health and preventing disease, especially through cleanliness

synonyms: cleanliness, sanitation, sterility, purity, disinfection

Still, despite the confidence companies have in open source software and their reliance on it, efforts to assure the cleanliness of open source code have not been robust to date. For example, more than half of companies that use open source software admitted they have no open source governance policies or procedures in place. Two out of three of companies admitted they don't monitor their open source code for security vulnerabilities.

Perhaps most surprising is that many companies admit they have no idea what open source software they're using, where open source code is located within their code base, and whether their open source software presents known security vulnerabilities.

UNEARTH VULNERABILITY INFORMATION WITHOUT SLOWING DEVELOPMENT

The presence of vulnerabilities in all types of software is inevitable, and open source is no exception. Detection and remediation of vulnerabilities in open source, such as in the case of high-profile vulnerabilities like Heartbleed and Freak, is increasingly seen as

2015 *the future of* OPEN SOURCE

78% OF COMPANIES RUN PART OR ALL OF THEIR OPERATIONS ON OSS

55% HAVE NO OSS GOVERNANCE POLICIES OR PROCEDURES IN PLACE

67% DON'T MONITOR THEIR OPEN SOURCE CODE FOR SECURITY VULNERABILITIES a security imperative and a key part of a strong application security strategy.

The good news is that while companies are developing an end-to-end open source security strategy, innovative tools are available today to give them a tactical head start. These tools can catalog all open source in software portfolios – from whole platforms such as Linux, Android and Hadoop, to individual code components, all the way down to the level of code snippets cut and pasted into internally-developed application code.

These scanning tools, when employed on demand or when integrated into software development workflows, provide critical information for companies to act upon, without slowing development or delaying time to market.

It's critical to develop robust processes for determining the following:

- Exactly what open source software resides in or is deployed along with an application
- Where this open source software is located in build trees and system architectures
- Whether the code exhibits any known security vulnerabilities
- How to create an accurate open source risk profile

Because most companies do not have open source security policies in place to guide them in the early phases of the application development process (setting requirements and software selection) the next best place to begin is in the application build stages when all of the open source assets required for an application are in place.

Once application code development has progressed to the build stage, companies can use the above-mentioned tools to perform thorough scans and inspection of the software. This process allows for the development of an application bill of materials (BOM) that is used to identify and map all the open source software components integrated into application code.

With such detailed information in hand (and online) about your organization's open source assets, Open Source Hygiene then entails cross-referencing each component against multiple open source vulnerability databases –the NIST National Vulnerability Database (NVD), the Open Source Vulnerability Database (OSVDB), and commercial databases such as VulnDB from Risk-Based Security. Automation then can streamline the creation of reports that call out deprecated versions (old or out-of-date components) along with providing recommendations for remediation.

The value of the information gathered from such scans is significant. At once the application's open source assets become transparent, any known vulnerabilities are identified, and an open source security risk profile emerges. Also, with automation, the hygiene of all open source code in use at your company can be constantly monitored from that point forward, allowing for alerts to be delivered immediately as new security threats emerge.

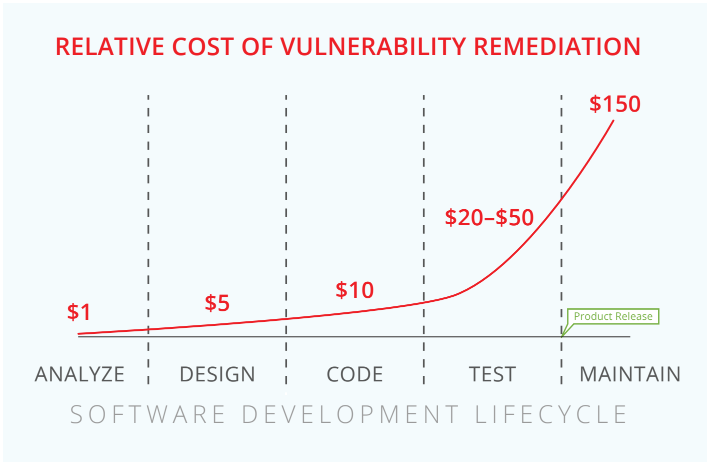
NOT JUST *WHAT* BUT ALSO *WHERE* AND *WHEN*

Knowing the exact location of open source code vulnerabilities in software portfolios is very important for enabling companies to assess security risk, decide which vulnerabilities must be fixed immediately, and set timetables for fixing lower-risk vulnerabilities. In cases where risk is deemed minimal (e.g., the code is unreachable by an external attacker), companies may conclude that the cost/time required to address a vulnerability is too steep and the offending code will be left in place.

Open source vulnerabilities identified in Internet-accessible applications usually gain higher priority for remediation because they're more susceptible to exploitation. Similarly, open source vulnerabilities in applications that are part of commercial products distributed to external customers are also strong candidates for prioritized remediation. By contrast, vulnerabilities found in internal-facing applications pose lesser risks and merit lower remediation priority.

This risk-based triage approach is very useful as companies calculate the associated costs and time required for vulnerability remediation and adjust for revenue forgone while remediation takes place.

Discovering and making fixes early in the development lifecycle can be 100 times less costly than during later development phases. Still, the most effective Open Source Hygiene strategies are end-to-end. They are implemented much earlier – ideally in the application requirements phase – and the process extends throughout the test and release phases.



OPEN SOURCE HYGIENE: THE BLACK DUCK WAY

While it's theoretically possible to practice Open Source Hygiene with spreadsheets and other manual techniques, it's next to impossible to practice comprehensive and timely open source vulnerability analysis without the use of automation. Black Duck is known for its high-performance scanning tools and the Black Duck® KnowledgeBase™, the industry's most comprehensive compendium of open source project profiles. The Black Duck Hub scans code bases to identify all the open source code in use, including exact versions of open source project code being deployed.

KNOWLEDGE IS POWER: *KNOW YOUR CODE*

We live in an era of pervasive and ever-increasing use of open source software in both development and deployment settings. Our age is also one of necessary focus on open source security. Vital open source security processes that assure open source transparency and hygiene will continue to proliferate and improve as companies demand to *know their code* inside and out.

CONTACT

To learn more, please contact: sales@blackducksoftware.com or 1.781.891.5100
Additional information is available at: www.blackducksoftware.com

BLACKDUCK