
Reduce Your Costs: Eliminate Critical Security Vulnerabilities with Development Testing

The Stakes Are Rising

Security breaches in software and mobile devices are making headline news and costing companies millions in lost revenue and damage to brand equity. As more people conduct increasingly sophisticated and sensitive transactions, the stakes around software security are rising. Plus, the software and platforms are becoming increasingly complex with multiple components coming from third-party suppliers. Companies often have little visibility into the security or quality of the third-party code which can introduce multiple points of failure and blame. Traditional approaches to security are no longer sufficient. For too many organizations, security is left to an isolated security audit team with limited resources to conduct at the end of the software development lifecycle. And the later the issues are raised in the lifecycle, the more expensive and time consuming they are to address.

Compounding this issue is the fact that security audit and development teams have different goals. Security audit teams are focused on risk—meeting audit and compliance requirements by ensuring security vulnerabilities are identified and remediated prior to release. Development teams, on the other hand, are driven by speed and innovation—delivering new products to market, faster, at the least possible cost. All too often, the security audit team will perform an audit at the end of the development lifecycle with a tool designed specifically for them, meaning it is designed to find every possible issue, and produces a large number of false positive results. Then, a PDF report containing the long list of potential security vulnerabilities—without context or guidance of where they exist in the code and how to fix them—makes its way to the developers desk as they are racing to get the product out the door on-schedule. The information is neither actionable, nor presented in the developer's workflow. Developers do not like working outside of their standard workflow so often potential issues raised by security auditors are ignored. To properly address security risks and vulnerabilities without jeopardizing speed or cost, companies must bring security into the development process.

Organizations cannot bring security into development by giving developers a security auditing tool. The developer will simply ignore the results in time because of the high false positive rate. The tool also will not work well within the developer's workflow and often requires too much security expertise. From a developer's perspective, a defect is a defect. They simply want to be pointed to the defect so they can quickly address it. Developers must be able to address defects that can lead to security vulnerabilities in the same way they manage quality defects. This means adapting security to the way the developers work, not the other way around.

Development Testing: The Perfect Complement to Security Auditing

Testing for security defects during development is the perfect complement to the testing conducted by security auditors at the end of the lifecycle. By testing the code during development, organizations can eliminate the majority of defects that can lead to security vulnerabilities and allow the security auditing team to focus on the edge cases and apply their resources more effectively. This combination of development testing and security audit testing can help organizations save time, money and resources.

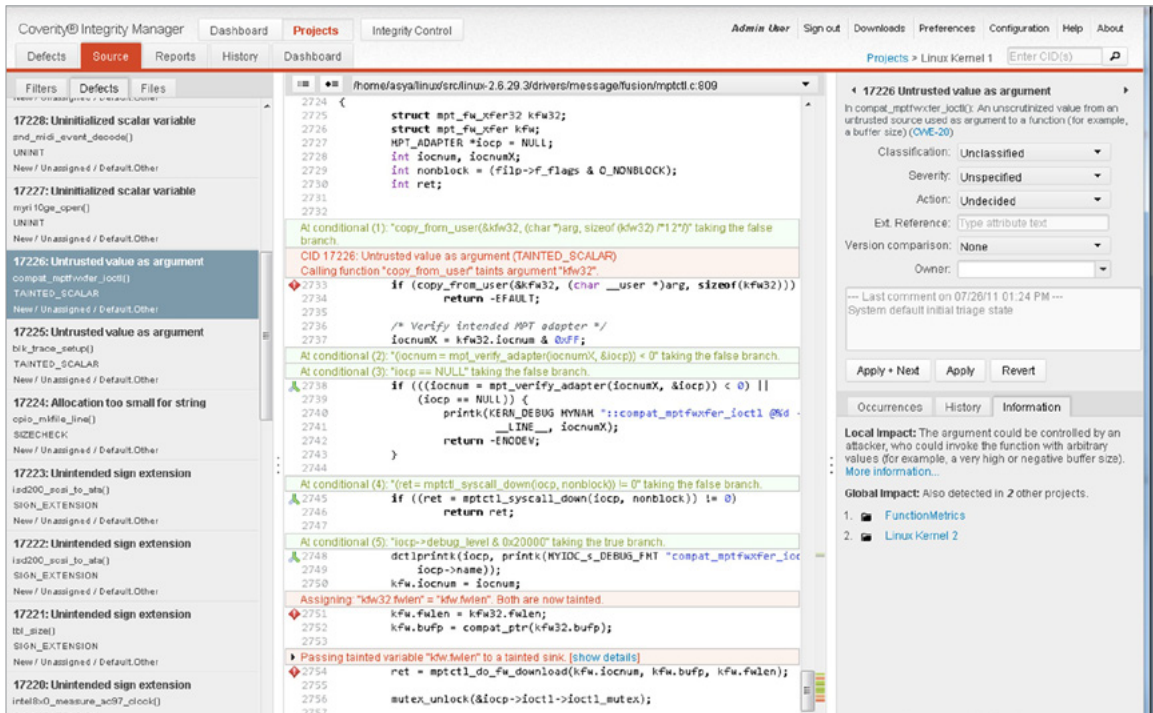
To properly address security during development, organizations need an automated approach for identifying defects. Existing testing methods and manual processes are no longer sufficient to address the problems given the increasing size and complexity of codebases. Automated code testing via static analysis provides developers with a means to automatically examine all of the code paths in minutes in many cases. This means more defects found overall, including ones on rarely executed paths that can be hard to reproduce and happen only infrequently. This also helps improve the security of the code as fewer defects in the product mean fewer potential places for an attacker to discover and exploit. And finding and fixing defects early reduces overall costs. The top three defects found with static analysis that can lead to security vulnerabilities include:

Defect	Description	Potential Impact
Buffer Overflow	A buffer overflow occurs when a program or process tries to store more data in a buffer, temporary data storage area than it was intended to hold. Since buffers are created to contain a finite amount of data, the extra information can overflow into adjacent buffers, corrupting or overwriting the valid data held in them.	Denial of service Arbitrary code execution Disclosure of sensitive information
Integer Overflow	Integer overflow is the result of trying to place into computer memory a whole number that is too large for the integer data type in a given system.	Denial of service Arbitrary code execution Disclosure of sensitive information
Format String Vulnerability	<p>Format string vulnerability denial of service attacks are characterized by utilizing multiple instances of the %s format specifier to read data off of the stack until the program attempts to read data from an illegal address, which will cause the program to crash.</p> <p>Format string vulnerability reading attacks typically utilize the %x format specifier to print sections of memory to which one does not typically have access.</p> <p>Format string vulnerability writing attacks utilize the %d, %u or %x format specifiers to overwrite the Instruction Pointer and force execution of user-supplied shell code.</p>	Allows an attacker to install a “back-door” to a system gaining access to sensitive information, files and programs on that machine

Arming Your Developers

The Coverity development testing platform is used by 3 of the top 5 security software companies in the world to solve their security issues. With over 1,100 customers and five billion lines of code currently under management, Coverity is the market leader for development testing. We've served as the quality and security gate to the shipment of over 11 billion products in the market. Coverity is a recognized expert in the area of security for C/C++ code. We've been working closely with CERT to evolve their secure coding standards and we've been deeply involved in the development of the C Secure Coding Rules which is a set of rules intended to be automatically enforced by analysis tools. Coverity's industry-leading analysis engines find critical defects articulated in the C Secure Coding Rules including buffer overflow, integer overflow, and format string errors.

Coverity provides accurate, actionable information about security and quality risks, prioritized together in a single interface which is part of the developer's existing workflow. Developers can view information about the severity of defects, the likely impact and where they exist in the code. This helps developers prioritize what defects to fix first, and provides them with guidance on how to fix the problems so they don't have to become security experts. They can quickly spot potential security issues such as buffer overflows and API error handling and prioritize those along with quality defects for the fastest time to resolution. And developers can find the defect once and then fix it everywhere it occurs across the code base. Developers can find and fix defects as the code is being written and as part of their standard workflow which is critical to speed of resolution and developer adoption.

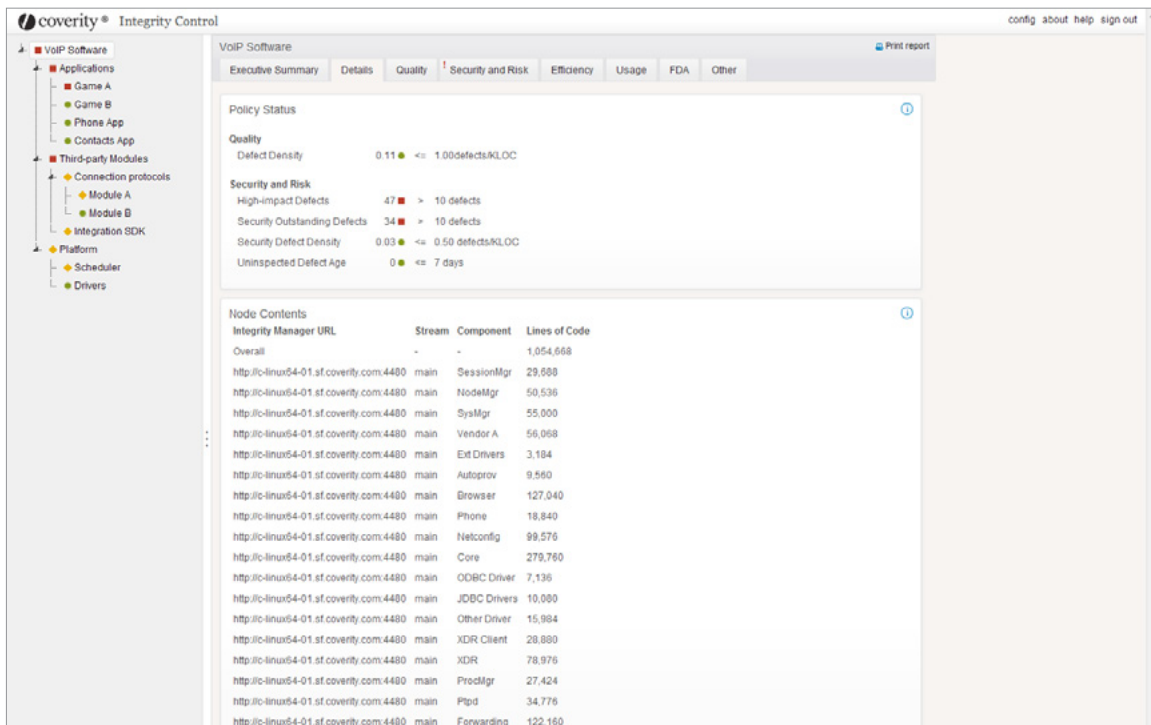


Test for Quality Defects and Security Vulnerabilities in Development

Improved Visibility and Control Across the Software Supply Chain

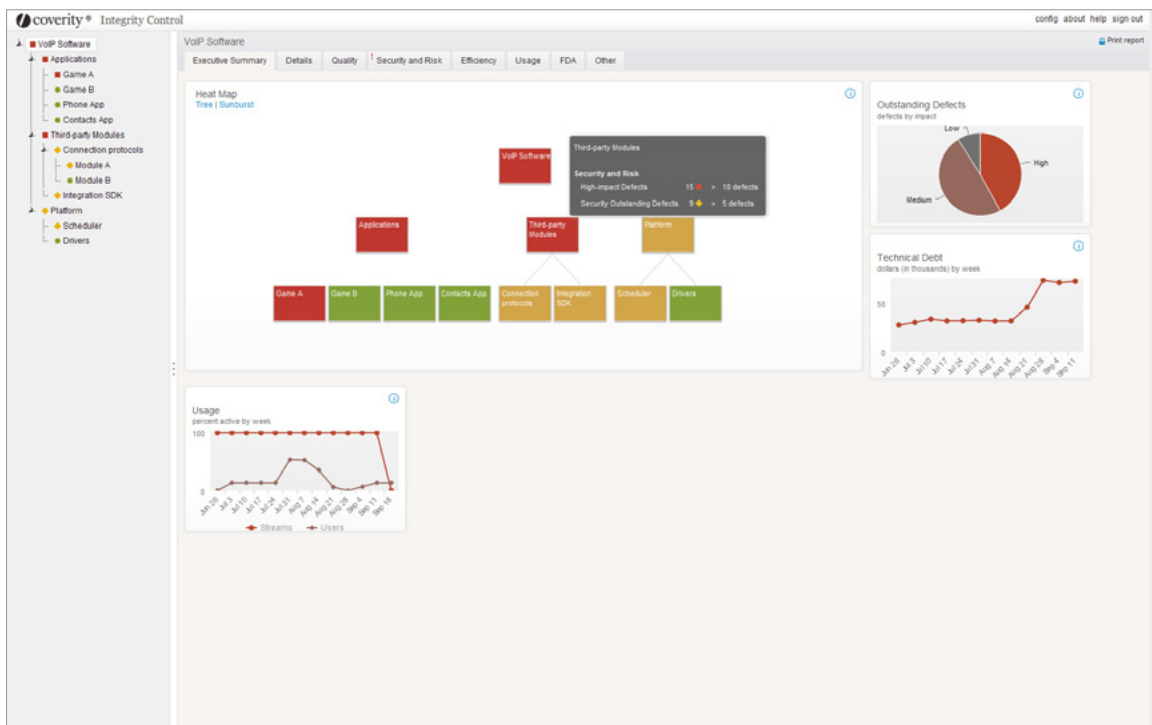
With the increasing complexity of software products, companies commonly rely on a myriad of software suppliers, from internal teams that share and re-use code to third-party commercial software suppliers and outsourcing development partners. Companies are increasingly being held accountable by their customers for the quality and security of the complete product. Yet according to a recent industry report, third-party code typically isn't tested with the same level of rigor as internally developed code. That means a defect could be lurking in the third-party code that could cause significant security breach or quality issue.

This increased risk has resulted in the demand for better visibility into all of the software components that make up a product. Coverity® Integrity Control enables managers to establish and enforce consistent measures for quality and security across the organization and across the supply chain. Organizations could set a policy for zero uninspected defects prior to launch since any one of those defects could contain a security vulnerability. Policies could also be established for zero security defects such as buffer overflow, integer overflow and format string errors.



Coverity Integrity Control Enables Consistent Enforcement of Quality and Security Standards

Once the policies have been established and tested against, organizations can quickly visualize the quality and security risk in their projects. Managers can quickly see which areas of their code are out of compliance with the established policies and drill down into detailed reports to pinpoint the issue. This information is critical for making decisions about whether the product is ready to be released and whether there is a quality or security issue with a third-party code provider that needs to be resolved prior to release.



Coverity Integrity Control Provides Visibility into Project Risk

Summary

Security must be addressed early in the software development lifecycle to minimize project—and business—risk and cost. We believe that the only effective way to do this is to bring security to the developer, not the other way around, in the same way they manage quality today. This means presenting developers with actionable information, in their existing workflow, so they can fix vulnerabilities in the same way as quality defects, as the code is written. While the security audit team should perform its audit throughout the development project, waiting until the end of the development cycle to identify vulnerabilities and pass them back to developers who are under time-to-market pressure, is ineffective. In addition, with the increasingly complex software stack and reliance on third-party suppliers, companies need better visibility into the security, quality and safety of their code. Coverity provides developers with actionable information that enables them to manage security and quality defects early in the lifecycle, in their existing workflow, without requiring them to become security experts. Coverity Integrity Control provides companies with the visibility they need to feel confident about the third party code they are shipping as part of their product—and under their brand—to reduce the real business risk of becoming the latest front page news headline related to software failure.

To find out more about Coverity can help your company address its security, quality and safety requirements contact your local sales representative or visit us at www.coverity.com.

About Coverity

Coverity, Inc. (www.coverity.com), the development testing leader, is the trusted standard for companies that need to protect their brands and bottom lines from software failures. More than 1,100 Coverity customers use Coverity's development testing suite of products to automatically test source code for software defects that could lead to product crashes, unexpected behavior, security breaches, or catastrophic failure.

For More Information:
sales@coverity.com

Coverity Inc. Headquarters
185 Berry Street, Suite 6500
San Francisco, CA 94107 USA

U.S. Sales: (800) 873-8193
International Sales: +1 (415) 321-5237
www.coverity.com

Coverity and the Coverity logo are trademarks or registered trademarks of Coverity, Inc. in the U.S. and other countries. All other company and product names are the property of their respective owners. © 2008-2011 Coverity, Inc. All rights reserved.

