

Ensuring Superior Software Quality

Coverity Prevent is the leading automated approach for ensuring the highest-quality, most reliable software at the earliest phase of the development lifecycle. The most accurate static code analysis solution available today, Prevent automatically scans C/C++, Java and C# code bases with no changes to your code or build system. Because it produces a complete understanding of your build environment and source code, Prevent is the tool of choice for developers who need flexible, deep, and accurate source code analysis.

Hundreds of development organizations worldwide use Prevent to automatically analyze large, complex code bases and root out the critical, must-fix defects that lead to system failures, runtime exceptions, security vulnerabilities, and performance degradation. Many belong to industries with necessarily stringent requirements for code security and reliability, such as medical device manufacturing, aerospace and defense technology, networking infrastructure, storage systems, automotive production, and financial services. These companies understand the severe consequences of a software failure and the vital importance of their code quality, and they trust Coverity Prevent to help them achieve high software integrity.

The sophisticated technology that powers Prevent ensures the most accurate and comprehensive static analysis currently possible, with the lowest average false positive rate of any static analysis tool on the market. Prevent is the only product to provide the Software DNA Map analysis system, which generates a unified representation of your build—not just calls to the compiler—giving developers a complete, high-fidelity understanding of the whole code base.

Benefits of Coverity Prevent:

- Automatically find critical defects that can cause data corruption and application failures
- Improve development team efficiency and speed time to market for critical applications
- Improve software integrity and end-user satisfaction

Prevent is flexible and designed to fit with your existing processes. It can be integrated with Source Code Management and bug tracking systems and its customizable workflow provides a collaborative platform for remediating defects. Coverity Prevent helps development teams increase time to market for critical applications and reduce costs by finding critical defects at the earliest stage in the development cycle. And when combined with dynamic analysis tools such as Coverity Thread Analyzer, creates a powerful software quality solution.

How Coverity Prevent Works: 3 Steps to Software Integrity

Unlike other source code analysis tools that focus on programming style and syntax-based checks, Prevent uses sophisticated, in-depth analysis techniques to uncover the critical, must-fix defects that matter most to developers. The only provider of Path Simulation (through dataflow analysis) plus Boolean Satisfiability (SAT) solving, Coverity achieves 100% coverage of code and execution paths to deliver the deepest and most accurate static analysis available—with false positive rates for some users as low as 5%.

Eliminate critical defects and improve software integrity.

Prevent's superior technology gives it these key capabilities:

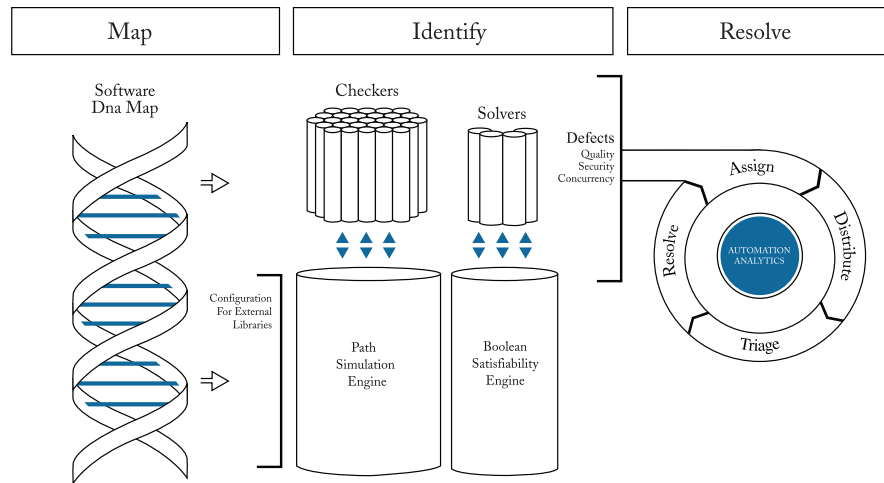
Highly Accurate Analysis - Prevent delivers the most accurate, reliable results available in static analysis and maintains the lowest false positive rate on the market—an average of less than 15%.

100% Path Coverage - Prevent analyzes 100% of the paths through your source code, ensuring that all possible execution branches are followed while also avoiding impossible paths.

Scalability & Integration - Prevent scans millions of lines of code in a matter of hours and easily integrates into your regular build process with no disruption to your development process.

Flexible Workflow - Prevent lets developers analyze code on their desktops or on the central build system, and the Defect Manager helps teams understand defects in minutes, assign ownership, and then resolve them via a customizable workflow.

Breadth of Checkers - Prevent's checkers are designed to find critical, must-fix quality and concurrency defects that can cause system crashes, memory leaks, memory corruption, and unpredictable behavior as well as security vulnerabilities such as buffer overflows and command injection attacks.



Coverity's three-step process provides a comprehensive methodology for proactively finding and eliminating critical software defects.

1. Map the Software DNA

Only Coverity Prevent features the advanced technology necessary to automatically achieve a complete view of every operation in your code base. Coverity's Software DNA Map analysis system provides the very deepest insight into your build system, giving developers a precise, unified representation of all relevant source code data used to generate executables. This high-fidelity representation of your code base is the first step toward ensuring superior code quality.

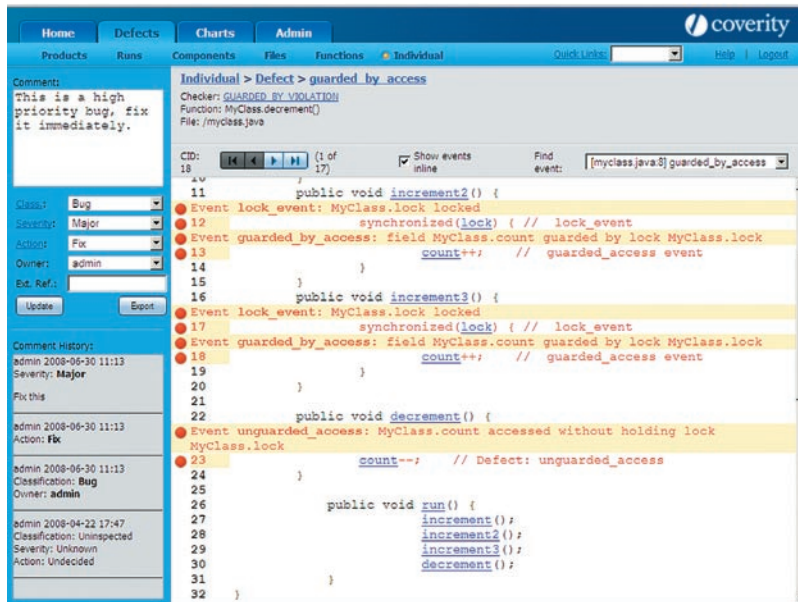
2. Identify Critical Defects

Leveraging the comprehensive data produced by the Software DNA Map analysis system, Prevent uses a breadth of sophisticated techniques to detect critical quality, security, and concurrency defects at the earliest stage in the development cycle.

- **Path Simulation and Interprocedural Analysis** - Path simulation traces the control flow through each function in your code base, enabling Prevent to simulate 100% of all values and data paths and detect a wide range of defects pertaining to resource allocation, pointer manipulation, buffer and string usage, and tainted data handling. Coverity's interprocedural analysis reviews complex call chains at any depth across files and modules by summarizing the effects of each function. Interprocedural analysis includes resolution of virtual function calls, ensuring deep coverage of object oriented code. A statistical analysis engine tracks behavior patterns throughout your entire code base, which enables it to infer correct behavior based on previously observed behavior. Additionally, false path pruning solves each branch condition to determine if it will be true, false, or unknown on the current path—this helps Prevent to efficiently remove false positives and achieve an average FP rate of about 15%.
- **Boolean Satisfiability (SAT)** - Long used by electronic design automation (EDA) companies for digital circuits, only Coverity uses this technique in C/C++ static analysis tools. Coverity's groundbreaking technology first generates a bit-accurate representation of your system from the Software DNA Map. Every relevant operation is then translated into a Boolean expression consisting of Boolean variables (i.e. variables having possible values of true and false) and Boolean operators (e.g. AND, NOT, OR), which are translated into formulas that are analyzed by SAT solvers. The SAT engine significantly improves testing accuracy by eliminating false paths involving complex combinations of conditions and by detecting critical arithmetic problems such as integer overflows.

3. Resolve Defects

It's not enough to simply identify defects. Developers must use static analysis data to resolve defects so they can deliver higher quality code. Prevent's intuitive Defect Manager interface makes it easy to quickly understand defects, establish ownership, and then resolve them via a customizable workflow. Analysis results are displayed in the Defect Manager, which enables automatic defect assignment and email notification whenever a new defect is found. Full path information for each defect is displayed, allowing developers to quickly track the root cause of errors, and critical attributes are in-lined directly in the source code to show what logic the analysis engine used to detect the error. Accurate cross-references provide access to the declarations and uses of all variables and functions, making error diagnosis even easier. Actionable reports provide up-to-date information about the status of current and historical defects, conveying instant visibility into the quality of the code.



Coverity Prevent's flexible interface provides a collaborative platform that development teams can use to triage defects, assign ownership and track defect resolution efforts.

Features: Prevent is equipped with advanced features that give it unmatched accuracy and depth of analysis and enable easy integration with your current development process.

Software DNA Map Analysis System: Coverity's Software DNA Map analysis system observes and records the build system and generates a comprehensive representation of the code base suitable for deep, precise static analysis.

Automatic Defect Detection: Prevent automatically scans your complete code base and identifies must-fix defects including memory corruption, failures and run-time exceptions, security vulnerabilities, concurrency defects, performance degradation, and much more.

Defect Manager: The comprehensive workflow platform makes it easy to view, triage, and resolve defects collaboratively via a customized workflow that mirrors your existing development process. The web-based interface allows developers to remediate defects anywhere, at any time.

Local & Central Analysis: Prevent allows developers to analyze code on central build systems or locally on their desktops before and after checking it in to the build system. Then triage defects after the nightly build and sync your results with the central server.

Boolean Satisfiability (SAT): This groundbreaking analysis engine ensures the most accurate defect detection and the lowest false positive rate available by translating C/C++ code into questions based on Boolean values, and then applying SAT solvers to determine if paths are feasible at runtime. Only Prevent offers the added precision of this breakthrough technique.

Concurrency Checkers: Prevent features concurrency checkers for today's complex multithreaded applications, allowing developers to eliminate deadlocks and race conditions that can cause data corruption or system failures.

Statistical & Interprocedural Analysis: To find the critical, crash-causing defects that matter most to developers, Prevent uses statistical analysis to infer correct behavior based on the behavioral patterns it tracks throughout your code base and performs deep interprocedural (whole-program) analysis across function, file, and module boundaries.

Lowest False Positive Rate: Prevent's sophisticated analysis engines intelligently avoid false positives by carefully considering the semantics and idioms specific to each language and error type giving Prevent an average FP rate of about 15%, with some users reporting FP rates as low as 5%.

Email Alerts: Automatic email notifications let you know when Prevent detects new defects in the most recent run for a certain component and also provide key intelligence about those bugs.

LDAP Integration: Prevent fits easily within the existing authentication procedures of organizations that require centralized administration of users.

Build Integration: Prevent integrates seamlessly into your regular build system without any system modifications or disruption to your development process, and with little or no additional hardware.

Actionable Reporting: Graphical reports provide instant visibility into code quality and up-to-date information about current and historical defects, giving managers all the information necessary to assess progress towards meeting quality goals.

Supported Environments

Coverity Prevent for C/C++

Supported Platforms	Supported Compilers	Supported IDEs	System Requirements
<ul style="list-style-type: none"> Windows XP, 2003 Linux Solaris HPUX Mac OS X FreeBSD NetBSD 	<ul style="list-style-type: none"> gcc, g++ Microsoft Visual C++ Sun C/C++ ARM Wind River TI Code Composer Green Hills Freescale/Nokia CodeWarrior IAR HI-TECH PICC Intel C/C++ HPUX aCC Marvell MSA Cosmic Compiler for Freescale Renesas H8S, SH, M16C, M32C, M32R Support for other ANSI C/C++ compatible compilers available upon request 	<ul style="list-style-type: none"> Eclipse 3.2 and later Eclipse-based IDEs including: <ul style="list-style-type: none"> ARM IBM Rational Mentor Graphics Nokia QNX Telelogic Wind River 	<ul style="list-style-type: none"> 1 GHz CPU 1 GB of RAM minimum, 2 GB recommended 1 GB of free hard disk space

Coverity Prevent for Java

Supported Platforms	Supported IDEs	System Requirements
<ul style="list-style-type: none"> Windows XP, 2003 Linux Solaris Mac OS X 	<ul style="list-style-type: none"> Eclipse 3.2 and 3.3 Eclipse-based IDEs such as IBM Rational Application Developer 7.0 and Borland JBuilder 2007 	<ul style="list-style-type: none"> JDK 1.4 or later 1 GHz CPU (x86 or SPARC) 1 GB of RAM minimum, 2 GB recommended 1 GB of free hard disk space

Coverity Prevent for C#

Supported Platforms	Supported Compilers	System Requirements
<ul style="list-style-type: none"> Windows XP, 2003 	<ul style="list-style-type: none"> Microsoft Visual Studio 2003, 2005, 2008 	<ul style="list-style-type: none"> 1 GHz CPU 1.5 GB of RAM minimum, 2 GB recommended 2 GB of free hard disk space

Free Trial

Request a free Coverity trial and see first hand how to rapidly detect and remediate serious defects and vulnerabilities. No changes to your code are necessary. There are no limitations on code size, and you will receive a complimentary report detailing actionable analysis results. Register for the on-site evaluation at www.coverity.com or call us at (800) 873-8193.

About Coverity

Coverity (www.coverity.com), the leader in improving software quality and security, is a privately held company headquartered in San Francisco. Coverity's ground breaking technology enables developers to control complexity in the development process by automatically finding and helping to repair critical software defects and security vulnerabilities throughout the application life cycle. Hundreds of companies worldwide including ARM, Phillips, RIM, Rockwell-Collins, Samsung, and UBS rely on Coverity to help them ensure the delivery of superior software.

Coverity Inc. Headquarters

185 Berry Street, Suite 2400
 San Francisco, CA 94107 USA
 U.S. Sales: (800) 873-8193
 International Sales +1 (415) 321-5237
www.coverity.com
scan.coverity.com