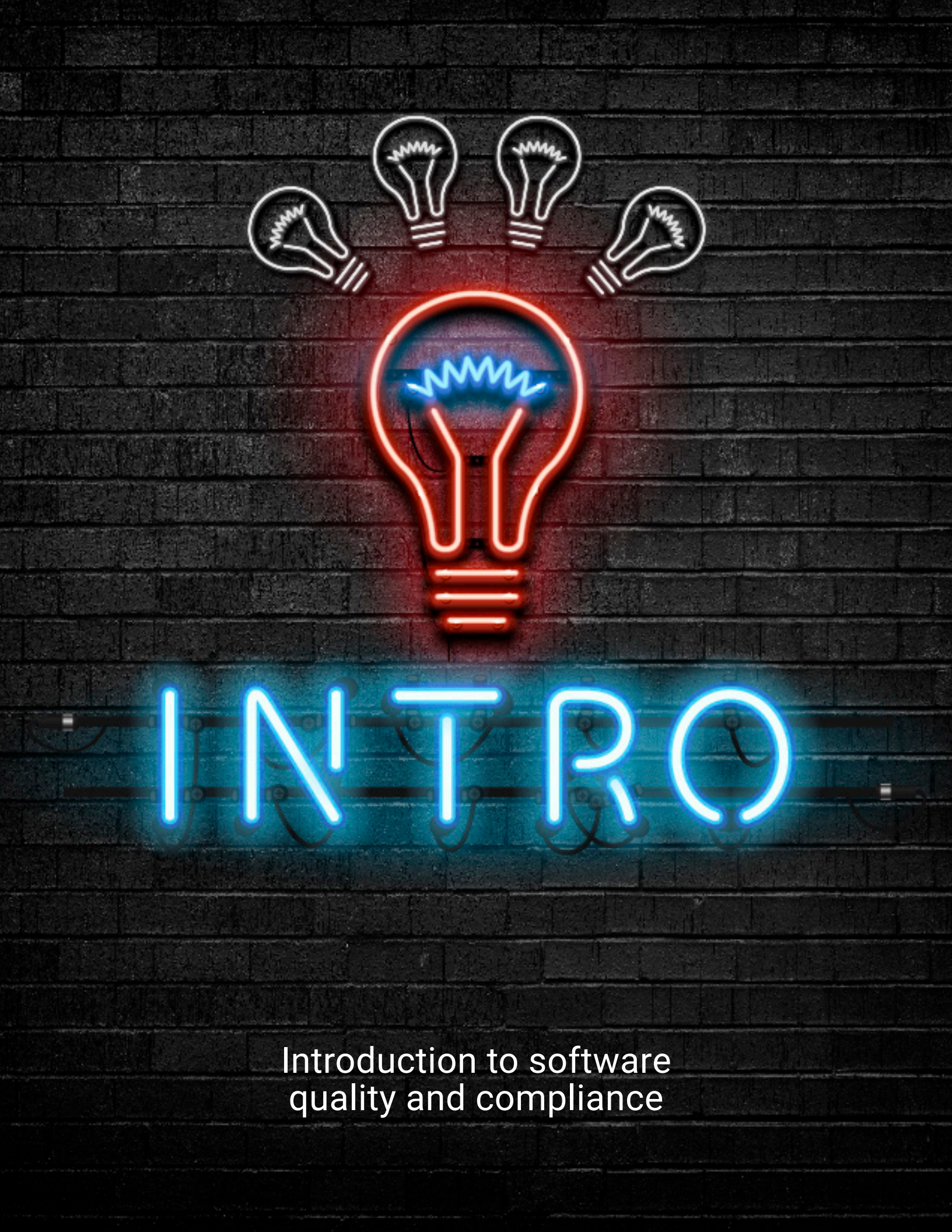
BLACKDUCK®

4 Software Compliance Gotchas to Avoid

How to Bypass Code Issues, Keep Regulators Away, and Stay Out of the News

Table of contents



Introduction to software
quality and compliance

Software quality and compliance are closely related

Many people think software quality and software compliance are completely separate topics. It's true that these two software attributes are distinct. "Software quality" refers to how well an application works: Does it do what it's supposed to do? Does it meet the market's needs? Can the development team easily maintain and upgrade it? "Software compliance," by contrast, refers to how well an application meets the requirements outlined in certain standards established by government and industry groups. One goal of these standards is to ensure that software isn't vulnerable to exploits that could lead to security or safety issues, including data breaches, bodily injuries, and even accidental deaths.

In reality, software quality and compliance are closely related: Well-written, high-quality code is less likely to contain the bugs, security vulnerabilities, and structural issues named in software standards. In short, writing quality code goes a long way toward writing compliant code. But the quest for software compliance doesn't end there.

Modern development teams face significant challenges as they make their way through the complex software standards landscape. This eBook explores four common gotchas your development teams may encounter on their journey to achieve compliance with the standards required in your industry, as well as recommendations for overcoming them.

Who *really* cares about software compliance?

Organizations rely more and more on software to handle sensitive data, automate business processes, and even protect people's safety. As a result, the consequences of software flaws have grown exponentially. For this reason, auditors and government agencies have drafted coding standards to help ensure mission-critical applications don't contain key software problems.

Given the potential repercussions of a single software bug or security vulnerability, these industry software standards are strictly enforced. Violations can result in steep fines, extensive legal fees, and significant damage to business reputation and credibility. In extreme cases, corporate executives can even face criminal prosecution.

Because of the wide-ranging effects of noncompliance, the demand for proof of compliance comes from many internal and external stakeholders. Auditors and global market regulators want to ensure that software doesn't contain bugs or vulnerabilities that might harm customers or the public or expose their data. Executives want to avoid financial penalties and criminal charges. And heads of legal, PR, and other outward-facing functions want to keep their organization from making headlines for all the wrong reasons: public data breaches, safety problems, and malfunctioning products.

In the end, though the demand for proof of compliance comes through many channels, only one group can demonstrate that the code they create indeed complies with software standards: the development team. A development team that doesn't prioritize software quality and compliance puts the entire organization at risk.

‘Checkbox’ standards compliance won’t prevent massive web app data breaches

While organizations have enjoyed the benefits of digital transformation via web applications, their growing dependence on these apps, coupled with high-profile security breaches, has made the task of securing web apps top priority. Most organizations still focus primarily—if not entirely—on protecting the perimeter, detecting breaches, and addressing them after the fact. Instead, they should focus on the layer where most breaches occur—the application layer—to prevent breaches altogether. Unfortunately, when it comes to web applications, just checking the compliance box is not enough to protect your data from breaches.

Web applications represent a common attack surface for hackers. As we've seen, exploitable software vulnerabilities in web applications have led to some of the highest-profile hacks in the last decade. For example, Equifax, Target, and Yahoo all suffered massive data breaches due to vulnerable web applications that handled sensitive information—resulting in steep fines and profound damage to their business reputations.

Many of these vulnerabilities are tracked by the communities that maintain web app security standards such as OWASP Top 10 and CWE/SANS Top 25. These standards help organizations produce secure web apps by providing strict guidance for preventing common and dangerous software security weaknesses such as code injection (e.g., SQL injection, command injection), cross-site scripting, and missing encryption of sensitive data.

Clearly, software standards can help organizations reduce the vulnerabilities in their software, but compliance alone isn't enough. A software standard can demand that you encrypt all data at rest, for example, but it can't detect whether you've implemented a cryptographic function correctly. In addition, software standards often lag behind new technologies (e.g., cloud, blockchain), opening up other opportunities for hackers.

Therefore, instead of focusing on perimeter security or checking boxes on software standards, organizations can better address web app security by building secure software that is difficult to hack.

How to avoid it

The outcome of having tens of millions of customers' personally identifiable information (PII) exposed is a PR, financial, and legal nightmare. So the consequences of a data breach extend far beyond the software development life cycle (SDLC). But to prevent data breaches, the first objective is to find and fix software defects early in the SDLC while keeping disruptions in the development pipeline to a minimum.

Obviously, no security tool will catch every bug. Fortunately, multiple technologies look at code in different ways to find exploitable software vulnerabilities, including static application security testing (SAST), dynamic application security testing (DAST), interactive application security testing (IAST), and software composition analysis (SCA). The challenge is to synthesize all these tests into a single comprehensive analysis so managers and developers can visualize all issues at a glance.

What you need: A software security solution that brings together results from different tools in one view. Having a single comprehensive view into the status of your applications will enable you to manage many types of risk across your entire web application portfolio.

Documenting compliance with data privacy regulations is difficult

Organizations that deal with PII in areas governed by data privacy regulations have to create and maintain a great deal of documentation. For example, they must assemble reports manually (including screenshots of security settings), fill out questionnaires, update policies and procedures, and complete self-assessments and/or undergo lengthy, expensive third-party audits—annually in some cases. They must update all this documentation with regular security scans to make sure PII is still secure as it flows through the organization or sits in storage.

Government and industry data privacy regulations make it public policy to protect customer data adequately—with substantial financial penalties and possible criminal charges for extreme violations. Many data privacy standards extend well beyond the realm of application security to provide guidelines for a complete sensitive-data protection program. Considering the increasing rate and consequences of hacks via exploitable software vulnerabilities, organizations are starting to recognize that application security must be part of their data protection strategy.

How to avoid it

When it comes to government and industry data privacy standards, getting the official stamp of compliance is essential to your go-to-market strategy. So it's critical to demonstrate compliance through well-documented reporting. Auditors, top executives, and concerned customers all demand proof of compliance, so being able to generate an electronic paper trail is vital.

What you need: Tools with extensive reporting functionality, or a tool or platform that can combine data from other tools and generate the reports you need. Customizable reports will help you meet proof-of-compliance requirements for complex industry standards such as AUTOSAR and DO-178C.

For organizations that operate in highly regulated markets (e.g., automotive, aviation, financial services) with complex standards, software

Data privacy regulations

Organizations that handle certain types of data must comply with data privacy regulations and standards, including these:

- PCI DSS for credit card data
- GDPR for data belonging to residents of the European Union
- HIPAA for health data

These guidelines name specific software security weaknesses to avoid, such as buffer overflow and insecure cryptographic storage.

They also require that organizations implement processes such as educating developers on secure coding practices and not using sensitive data in testing environments.

To demonstrate compliance with these regulations and standards, an organization must provide extensive reports showing, for example, security features they've implemented, software settings they've selected, and policies that delineate the proactive steps they've taken to improve data security.

Modern software compliance standards
in regulated markets are complex

development teams need solutions that make testing, enforcing, and demonstrating software quality and software compliance easier.

On any development team, skill sets vary as to secure coding practices and writing functional code. Development leaders struggle to create consistent, repeatable processes that enable developers with different strengths to find and fix security weaknesses quickly.

In addition, the raw volume of applications in modern organizations can be overwhelming. Going through them manually to achieve software compliance is neither scalable nor realistic. Whatever the compliance strategy, it needs to scale to hundreds of projects and millions of lines of code (LOC).

But before implementing a strategy to achieve compliance, organizations must understand the unique requirements demanded by the software standards that apply to them.

Code quality standards in embedded software

As embedded code replaces electronic, analog, and mechanical systems, auditors continue to enforce the strict code quality standards they've drafted to keep the users of embedded applications out of harm's way.

These standards are meant to reduce the likelihood of a critical failure in embedded applications. They include, for example, restrictions around null statements, library functions, and memory allocations. Organizations can't expect to go to market if their embedded applications don't comply with the relevant industry standards. Even worse, organizations using noncompliant embedded software in shipped devices may be forced to recall all devices with that software.

You can find embedded software quality and security standards in these publications:

Automotive

- AUTOSAR focuses on standardizing automotive software interfaces, making it easier to scale, transfer, and reuse functions across platforms.
- MISRA C and C++ are aimed at making automotive C and C++ code safer, more secure, and more reliable.
- ISO 26262 covers the entire development process, with the goal of improving the safety of all automotive electrical and electronic systems.

Software using C/C++

- CERT C/C++ and ISO 17961 both outline secure coding rules for C/C++ applicable to any industry or application.

Aviation/airlines

- DO-178C provides recommendations for avionics software development.

How to avoid it

In embedded applications, software issues can manifest themselves physically—potentially causing harm to consumers.

What you need: Tools that can help your developers write clean software and find bugs in embedded applications early in the SDLC, before products ship.

Considering the scale and complexity of modern application portfolios, the ability to track and manage specific issues named in regulations will keep your team focused on the most important items.

What you need: Tools that automatically assign severity—based on compliance—to issues and triage them accordingly. The ability to sort and filter issues by severity, standard, and other criteria is also valuable.

Software compliance can
slow down development

Treating compliance and testing as a separate step from development is a surefire way to slow down development.

Software compliance requires software security and quality testing. A key challenge to producing compliant applications is to conduct this testing without disrupting existing development processes.

The developer's primary job is to produce functional software by a certain deadline. But as stakeholders pile on feature requests and competitive timeframes shrink, development teams have become overburdened, understaffed, and time-crunched. And while DevOps has accelerated application delivery by supporting continuous development with quick release cycles, it has also changed how development teams should approach software compliance.

Faced with continuing pressure to do more with less, developers resist new processes that add friction and complexity to their day-to-day routine. In addition, anything that disrupts their development workflows, causing them to push or miss their project deadlines, puts the organization at a competitive disadvantage.

So even though testing is essential for compliance, many developers have already become disenchanted by their experiences using testing tools that can't match the pace of modern software development. Treating compliance and testing as a separate step from development—basically, asking developers to "bolt compliance on" late in the workflow to conform with standards—is a surefire way to slow down development.

How to avoid it

To integrate software compliance into the SDLC without slowing down development, managers must give developers the flexibility to test for compliance in a way that works best for them. Again, developers' No. 1 objective is to produce functional applications on time. If compliance strategies work against this goal, developers are unlikely to embrace them. But if developers can code in their integrated development environment (IDE) and perform compliance processes at the same time, they can catch errors earlier, when it's easier and less costly to correct them—reducing the number of issues in long revision loops. Other teams may prefer to add compliance as a gate in the CI/CD pipeline to align with their automation strategy. Either way, managers should strive to implement compliance organically into the SDLC and developers' day-to-day processes and avoid changing existing workflows too much.

What you need: Tools that fit into existing pipelines and don't force developers to add even more work to their workflow. Tools that offer a range of integrations, plugins, and APIs can help you inject security into existing processes rather than adding a new process.

Bringing it all together

Whether you're trying to move beyond mere "checkbox" software compliance and protect web applications under constant assault from multiple attack vectors, create and maintain customer data privacy documentation per government requirements, conform to complex modern software compliance standards in regulated markets, integrate application security into the development process, or some combination of the above, the world of software quality and compliance is tricky. Gotchas lurk in the least likely places and sometimes hide in plain sight, waiting to ensnare the less vigilant.

No single software testing solution has all the answers. But a fully thought-out, comprehensive application security program that combines technology and best practices is the strongest strategy for keeping the regulators out of your offices and your organization out of the headlines. It will pre-empt these gotchas by bypassing coding issues and finding as many bugs and vulnerabilities as possible, as early as possible.

But application security must not come at the expense of development efficiency. It must be a seamless part of the workflow, not a separate process, so that developers see security as an organic part of making high-quality software. Finally, documentation must be an integrated part of the solution, not a manual process that takes weeks or even months to complete. An automated testing and reporting methodology must be available to help document compliance with government and industry security standards and data privacy regulations.

Not sure where to start?

Black Duck has all the tools and services you need to achieve not just software compliance and software quality but true software security. We can help you avoid data breaches and compliance gotchas by implementing security at every step of your software development life cycle, from developer to deployment. At Black Duck, we help organizations build secure, high-quality software faster.

Let Black Duck help you build secure, high-quality software faster.

READY TO LEARN MORE?

About Black Duck

Black Duck® offers the most comprehensive, powerful, and trusted portfolio of application security solutions in the industry. We have an unmatched track record of helping organizations around the world secure their software quickly, integrate security efficiently in their development environments, and safely innovate with new technologies. As the recognized leaders, experts, and innovators in software security, Black Duck has everything you need to build trust in your software. Learn more at www.blackduck.com.