CloudBees®

eBook

Stop paying the compliance tax: A guide to continuously compliant software

You've streamlined your software development process for faster feature delivery. You've adopted continuous integration and continuous delivery (CI/CD) processes to make it more user responsive. Perhaps you've even reached a state of enterprise progressive delivery. But, you're still not done.

There's another pressing demand on your software team, and it's dragging down your productivity and ability to bring innovation to market. Your next big digital transformation challenge to tackle? Software compliance.

Software compliance is a financial and logistical burden that delivers no direct customer value but can incur large legal and regulatory fees, as well as lost developer time and talent retention issues if done incorrectly. In this sense, it's like a tax. Every unnecessary dollar you spend satisfying it through manual processes prevents you from burning down your production backlog, retiring technical debt, and bringing more value to customers.

This eBook will explore what we mean by the compliance tax, how companies can assess their tax, and how to mitigate it across the software development lifecycle (SDLC) with a holistic, automated approach.

What's covered under software compliance?

To be truly compliant with regulatory and internal frameworks, the scope of software compliance must cover the entire digital estate. This includes five main areas:

- Source code
- Compiled application binaries
- Production environments (both cloud and on-premises)
- Users' identities and access management infrastructure
- Data your applications use

Organizations must also consider all three typical compliance deliverables: assessment, attestation, and artifacts. Assessment covers the initial evaluation of the software to determine whether it complies with the appropriate regulations and controls. A senior executive, typically from the legal or compliance teams, must then attest to that compliance. These first two attributes are closely connected. Attesting to compliance carries legal and potentially financial liability, meaning that a company must be sure the initial assessment is accurate. Lastly, compliance regulations require proof in the form of credible artifacts; the paper trail showing when and how the assessment took place and who carried it out.

This process requires ample dedicated time from team members, all of whom could be focused on driving value elsewhere. The sum of resources spent, financial and otherwise, combined with the opportunity cost, could be equated to a compliance tax all organizations in highly regulated industries pay.

“

Organizations must also consider all three typical compliance deliverables:
- **assessment**
- **attestation**
- **artifacts**

What is the compliance tax?

Proving compliance involves **assessing** the current state of the software delivery life cycle, **asserting** that the process and digital assets are in compliance with both internal and regulatory controls, and then providing the **evidence** that supports that assertion. Simple enough on the face of it, but in reality, this is a DevOps anti-pattern. Releases have to be delayed, valuable resources have to be shifted to no-value workloads like log mining, and developers have to be taken off innovation to attend yet another meeting on the latest compliance requirements or to recreate builds from months in the past—thus the genesis of the compliance tax. Let's take a closer look at those three elements of compliance and how they contribute to the tax.

Assessing compliance

Assessing the state of the software delivery process for compliance is more than running a few static or composition analysis scans early in the cycle. Those only look at code at that point in time. "Shifting left" ignores most of the rest of the digital estate, such as runtime environments, binaries, identities, and data, but it also ignores the very real prospect of something changing post-scan.

Add to that a highly complex tool chain, disconnected processes or departments, and multiple data pools to analyze, and it's no wonder that assessing the estate is a time-consuming and labor-intensive, albeit necessary, step. Someone has to wade through those logs to find out what happened or why a decision was made and that is usually at the cost of spending time on innovation.

Asserting compliance

There are two main stakeholders who must assert compliance, one of whom can be at risk personally and financially for that assertion. First, the application owner or head of software engineering has to assert that the software has passed agreed standards and they have chosen to remediate the right issues in order to release the software. Second, the Chief Information Security Officer (CISO) must assert compliance to the board of directors as well as to multiple external regulatory bodies. If that assertion is ever to be challenged and found incomplete, the CISO can be fired, and in some global jurisdictions can be held legally and financially responsible for that problem. Both stakeholders have to trust the assessments and the data those assessments generate.

How do they do that? For the engineering side, it involves time spent in meetings understanding the findings of all the security scans, knowing how the regulatory controls are translated into checks and rules at the software level, deciding which issues need to be fixed, and then more meetings to determine if those issues have indeed been addressed. Developers are often left to their own devices to know what scan results mean or attending more training sessions from the security team to truly know what secure and compliant means. For the CISO, it also means multiple meetings, delayed releases, and extra costs for employees to source and analyze the data (from a process that is not entirely visible to them).

Evidence of compliance

As discussed above, sourcing and providing the data for compliance or an audit is a highly inefficient process and, by definition, suspect. Is the data correct? Does this include all the data? What data is not included? Does it include who approved an exception and the reasons why that decision was made? Does the data reflect the current state of things? Which controls are the least effective? How does a control failure affect the applications delivered or the critical business service that relies on that application? If a demand letter or a surprise audit request arrived today, what effort would it take to collect and provide that evidence and what impact would it have on overall operations and release schedules?

Getting answers to these questions requires more bodies, more time, more wasted effort—all driving down time available for innovation or retiring technical debt.

How to calculate the compliance tax

Organizations often underestimate the compliance tax. They might acknowledge the labor cost involved—for instance, if 50 developers, each paid an average of $52.95 per hour, spend three hours each week completing compliance reports, then you might assume a weekly cost of 50 x 3 x $52.95, or $7,924.50. This translates to $95,094 per quarter.

The real cost of compliance is far higher. It combines two figures: An employee's fully burdened cost and the opportunity cost of not having their time spent on work that delivers value to customers (and the bottom line).

Calculating the fully burdened cost

The fully burdened cost of an employee takes into account their annual pre-tax labor cost, their direct related employment costs, and their actual hours worked. We can describe it with the following equation:

(Annual pre-tax labor cost + direct related labor cost) / actual working hours

Here's how we calculate those three metrics:

- **Annual pre-tax labor cost:** This is a factor of the employee's pre-tax hourly wage and their available working hours. The available working hours might be 40 hours a week multiplied by 52 weeks (2,080 hours). At an hourly rate of $52.95, that would equate to $110,136.

- **Direct related labor cost:** Calculate this figure by adding together total costs, including benefits, equipment, insurance, meals, payroll taxes, property taxes, supplies, training, and utilities. Divide these costs by the number of employees involved in compliance to reach an average per-employee labor burden cost. As an example, let's assume these costs total $12,000 per employee.

- **Actual working hours:** Few employees work all of their available hours. They will take allowed vacation time and possibly sick days. Calculate the hours from these potential absentee days and subtract them from the available working hours to arrive at the employee's actual working hours. Let's say an employee is allowed three weeks of vacation time and the average sick time is two days. That's 17 days x 8 hours, or 136 hours. Subtract that from the available hours (2,080) to get the actual working hours (1,944).

These example figures give us the following fully burdened cost per employee:
(110,136 + 12,000) / 1,944 = $62.83

Factoring in the opportunity cost

The fully burdened cost per employee gets us some way toward the compliance tax—but again, it doesn't tell the whole story. Developers identifying, fixing, and proving compliance issues are not working on other projects. This takes time away from innovation-focused work that delivers customer value. It also pushes back planned release times for software that could save your business money or boost revenues.

This opportunity cost also stops companies from paying down technical debt. Every system has architectural compromises introduced for expediency or cost reasons or simply due to a lack of expertise. Over time, this technical debt constrains the introduction of new features and makes it more difficult to improve software quality. Good development teams devote some time to eradicating this debt by fixing underlying architectural issues.

We can assess the opportunity cost by looking at the value that companies expect employees to generate. This value is naturally higher than their burdened cost.

Depending on the company and the type of employee, the expectation could range from 1.5x the fully burdened cost up to 3x or more. The higher the expected value, the more value the company loses when the employee focuses their time on software compliance. In this way, opportunity cost is a multiplier for the compliance tax.

Assessing the compliance tax: A real-world example

Estimating the compliance tax involves more than just assessing developer time. It also includes paying auditors, risk managers, risk stewards, security, and compliance teams. CloudBees recently explored the real-world cost of the compliance tax at a global banking customer that devotes 100 employees full time to proving software compliance. This is how it was calculated:

- Fully burdened cost of labor: $16.8 million. The average salary was $140,000, but the fully burdened cost added 20% to that figure, bringing it to $168,000 per employee. That equated to a total labor cost of $16.8 million for 100 workers.

- Opportunity cost: $33.6 million. The bank expected each employee to deliver twice their fully burdened cost in value. This amounted to $336,000 per worker for a total of $33.6 million.

- Compliance tax: $50.4 million. The compliance tax comprised the fully burdened labor cost ($16.8 million) added to the opportunity cost ($33.6 million), bringing us to a total compliance tax of $50.4 million.

Approaches to mitigate the compliance tax

Shift-left compliance

The last few years have seen a focus on "shift-left" when it comes to security and compliance efforts, where developers tackle problems earlier (and, ideally, proactively) in the software development process. The idea is that mistakes are cheaper to fix during these stages. It's a common theme in DevOps initiatives.

Shift-left compliance is certainly useful, but development teams shouldn't take a restricted approach to it. The early stages of the development process are not the only areas where compliance needs proving. It's just as important to demonstrate compliance during production, where environments and conditions often change.

The people solution

This focus on compliance at all stages in the software development lifecycle forces some companies to devote more people to assessing compliance. They must write compliance policies and map them to the software's source code. Then, compliance staff and developers must deal with them, compounding the compliance tax.

Burdening developers with compliance work is the biggest problem with the people solution. These workers are in short supply, and constraints will only increase. The Bureau of Labor Statistics predicts a 22% increase in vacancies for developers and software quality assurance analysts between 2020 and 2030, from 1.8 million to 2.3 million.

Companies facing a shortage of developers cannot afford to sacrifice their morale. Forcing them to respond to ad hoc compliance requests by filling out reports affects their job satisfaction, increasing the likelihood of attrition. Throwing more and more people at the compliance problem also carries other issues.

It's inefficient, as in there's no guarantee that compliance staff and developers will identify all of the control risks in a piece of software. And with compliance regulations changing frequently, it's hard for these teams to remain up to date. It's also a recurring issue. Taking a team offline to check compliance provides a point-in-time snapshot. Software constantly changes, and there's no guarantee of continued compliance in the future.

The point tool solution

Automating software compliance somewhat solves these problems. In theory, automation enables companies to track software compliance more actively. In practice, they build these solutions from individual tools that don't work well together. Typically, these solutions include:

- Source code analysis, using static and dynamic application security testing tools.
- Open Policy Agent, which codifies policies in declarative code.
- Cloud Security Posture Management, which automates risk identification and remediation in cloud infrastructures.

These tools are siloed. Although they might catch some issues, they don't provide a joined-up view of the entire digital estate. Neither do they map the compliance issues they do find to real-world business risk.

This leaves developers dealing with a potential flood of security alerts, none of which have the necessary context. On paper, everything looks urgent, but your team must make defensible decisions to do one thing over another. Red lights end up flashing everywhere, driving up total remediation times.

The holistic solution: Shift security everywhere

The optimal solution applies these automated solutions across the entire digital estate in a more joined-up process. This expands the focus beyond the early stages of software development to secure the entire development pipeline. It's a concept that CloudBees calls shifting security everywhere.

After baking security into the design of the software, this method secures the development pipeline itself to reinforce security and compliance principles throughout the entire SDLC, all the way to production. This inclusion of security into traditional DevOps processes creates a more robust DevSecOps methodology that automates security and enables teams to better assess and prove their software compliance. It also covers all parts of the digital estate, ensuring thorough compliance coverage.

Shifting security everywhere offers some key advantages that reduce the compliance tax:

- **Automated discovery:** The solution examines the entire digital estate to surface compliance issues to the right team members as soon as they arise.
- **Continuous compliance:** Because the solution is automated, it continuously assesses compliance risk across the digital estate. This makes time spent on remediation more predictable and eliminates emergency ad hoc rushes.
- **Identification and prioritization:** An integrated, automated tool set highlights the most pressing issues for developers to fix by mapping the compliance issues to business risk. This involves evaluating each issue in context using an end-to-end view of the digital estate to assess the assets affected, the critical business services supported, the software's stage of development, and the regulatory controls that are triggered. Prioritizing compliance risks reduces the total time that developers must allocate to fixing them and also reduces the burden on other less technical compliance staff.

Solve fragmented software compliance with visibility, control, and context

A secure and compliant software delivery supply chain is the hallmark of digital transformation excellence. Reduce your compliance tax by introducing solutions that automate the assessment and proof of compliance, giving those ultimately responsible for attesting to compliance the confidence they need.

These solutions should go beyond data compliance to cover the other areas of the digital estate: the software source code and binaries, user identities, and production environments. They should automate this process not just at the beginning of the software development process but at every stage of the CI/CD pipeline, through to production and beyond.

Getting software compliance right will improve morale among developers. It will also free them to burn down the development backlog and retire technical debt, eliminating the hidden opportunity cost of poorly managed, manual software compliance approaches. Most of all, it will enable your company to maximize the delivery of software that delights customers and satisfies business goals in the future.

Learn more about CloudBees' rich suite of capabilities to streamline your software compliance operations.

CloudBees®

CloudBees, Inc., 4 North 2nd Street, Suite 1270 San José, CA 95113 United States
www.cloudbees.com • info@cloudbees.com