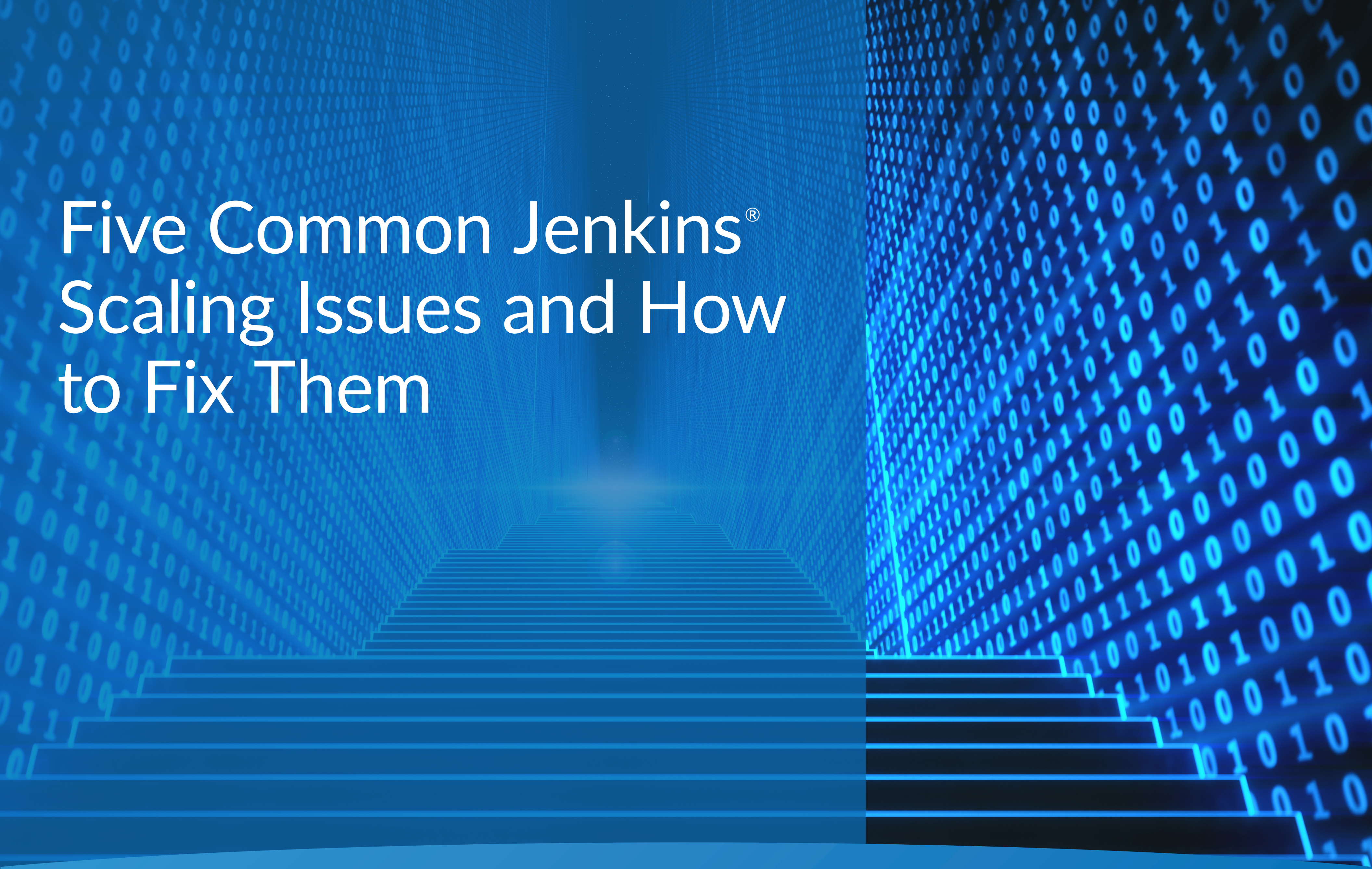
Five Common Jenkins® Scaling Issues and How to Fix Them



CloudBees®

Jenkins is arguably one of the most popular development tools on the planet, with some reports estimating that over 70% of all continuous integration pipelines run on Jenkins. It's great at helping small, agile development teams integrate code multiple times a day.

However, as teams, environments, projects and market pressures increase, it's easy for some of the maintenance that comes with building and using Jenkins pipelines to grow burdensome. For instance, you might lose sight of how many teams are on each instance, how many controllers are in use or who has made changes to code and scripts.

These challenges often arise as an organization reaches the enterprise level and looks to scale its Jenkins usage across the company, without affecting the security or stability of its software development lifecycle. In working with many of the Fortune 100 enterprises, CloudBees has identified the five common reasons enterprises struggle to scale Jenkins and how to fix those issues.

Five Common Jenkins® Scaling Issues and How to Fix Them

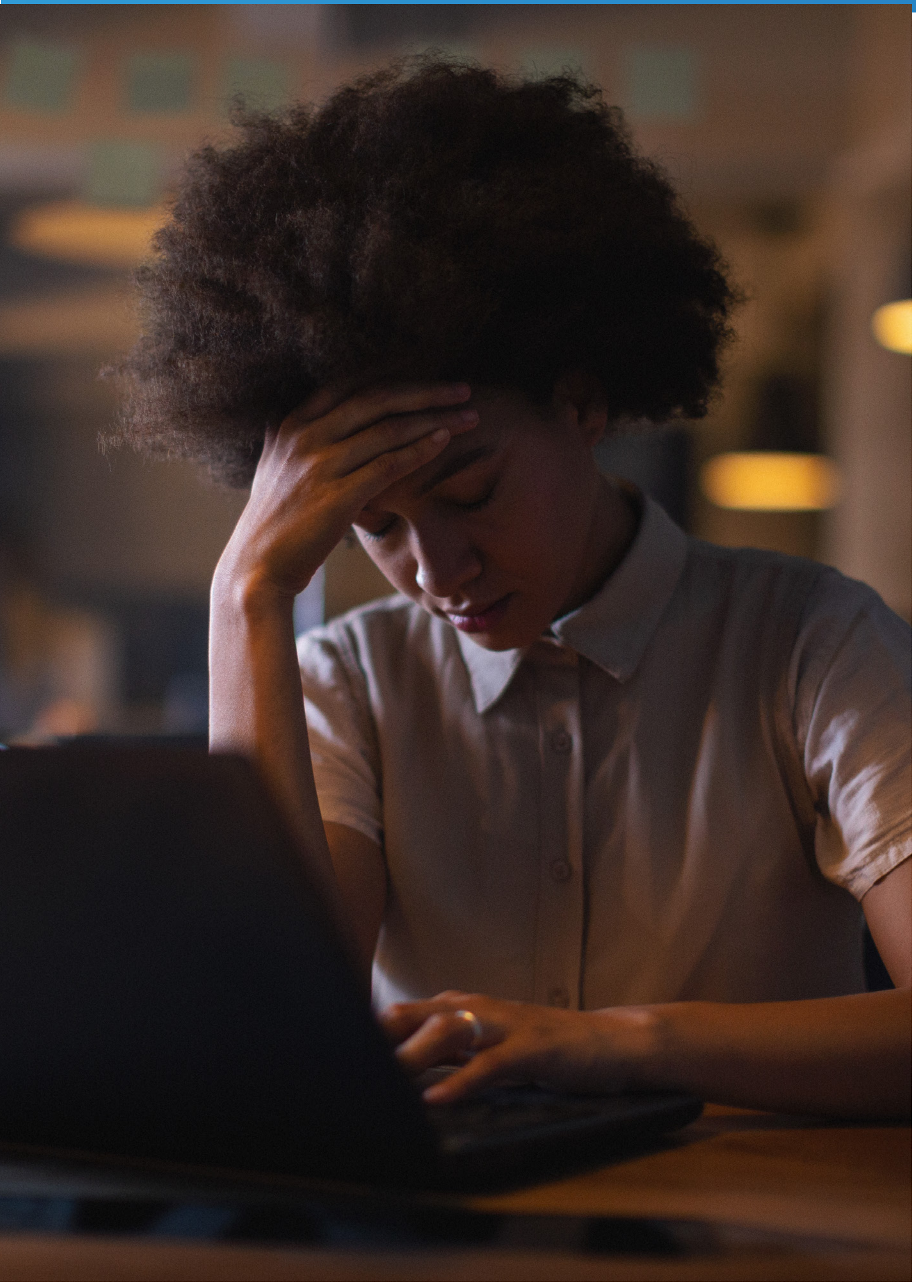
1. Your Jenkins instance is fragile, so you avoid upgrades and changes.
2. You have lost track or control over which plugins are being used or should be used.
3. You experience multiple outages and restarts that take 20 minutes or longer.
4. Your teams are on their own "Islands of Jenkins."
5. You have concerns over security and credential management is out of control.

This eBook further explains those Jenkins scaling challenges and how to extend your existing investment into an enterprise-grade Jenkins environment that makes software development repeatable, auditable and scalable.

1. Your Jenkins instance is fragile, so you avoid upgrades and changes.

A fragile Jenkins instance is largely due to the competing needs of an overused and poorly architected controller. It's common for Jenkins usage and benefits to spread through an organization and have user after user ask to be onboarded to the same Jenkins instance. The end result is a Jenkins instance in use by multiple teams with conflicting plugin needs and maintenance schedules.

To achieve the ideal balance between freedom, security and manageability CloudBees recommends a controller per team model with no more than 75-100 jobs each. This team-based approach boosts the performance and speed of your build jobs and reduces cross-team friction when managing plugin versions and agents' needs. Bonus – a controller per team approach limits the blast radius when something goes wrong.



2. You have lost track or control over which plugins are being used or should be used.

Every team has its own standalone flavor of Jenkins and bespoke build processes. This includes their preferred plugins. This can make keeping track of plugin inventory, a best practice to ensure system stability and optimal performance, difficult for Jenkins administrators. Having fewer plugins also typically means less upgrade risk and less verification effort needed during an upgrade.

Can you easily name answers to the following questions? If not, you need to analyze your plugin usage ASAP.

- Can you list the names of your installed plugins across all teams?
- Do you know how often plugins are being used?
- Where is each plugin located?
- What projects are currently using each specific plugin?

3. You experience multiple outages and restarts that take 20 minutes or longer.

This is another sign of a poorly managed Jenkins instance. Startup time is proportional to the size of your instance (number of jobs, users, installed plugins). We generally observe Jenkins handling up to 100 simultaneous builds, with some Jenkins regularly running many multiples of this number. However, poorly written or complicated pipeline code can significantly affect the performance and scalability of Jenkins.

The solution for this is implementing a distributed build architecture by breaking up your single instance into multiple client/managed controllers for each of your teams. Spreading the Jenkins love improves performance and prevents restarts.

4. Your teams are on their own “Islands of Jenkins.”

You are following best practices. Each team within your organization has its own Jenkins controller. That keeps the Jenkinstein issues at bay, but it also presents new challenges. All those separate servers create the sense that each team lives on its own island.

On these “islands,” engineers are left to maintain different Jenkins instances and configurations, wasting valuable time and resources, and creating lots of hidden costs for the company. What about governance? Some teams might ensure that security scans are built into every segment of their CI pipelines while some teams may not. But from an administrative point of view, there is no oversight, no control on what these individual servers are doing. Collaboration also suffers. If a team discovers a new way of doing things that are more efficient, they have no way of sharing this information because the teams are operating in silos.

5. You have concerns over security and credential management is out of control.

One of the beauties of Jenkins is it's so easy for anyone to set up a server and start integrating code. One of the major drawbacks of Jenkins is that it is so easy for anyone to set up a server and start integrating code. Users are frequently onboarded without a clear definition of roles or team association. Without these important characteristics defined, governance measures such as RBAC are essentially pointless.

When it comes time for an audit, lack of governance can become a nightmare. With so many instances of Jenkins in place, generating a report on every single action taken, by user and role, as well as all edits to every model/object definition becomes its own project. That is neither efficient nor practical in the long run. You spend more time preparing for an audit than you do delivering software!

Enterprises Grow from Jenkins® to CloudBees

While Jenkins® is the leading automation platform for continuous integration (CI), enterprises have unique needs. They must be able to scale CI across a multitude of teams without increasing the administrative burden. Enterprises must ensure security and compliance without inhibiting productivity. At the same time, a growing number of them are embracing a hybrid cloud strategy. Supporting a vast range of infrastructure types and optimizing software delivery across multiple tools and teams presents complex, real challenges for enterprises.

Run on Jenkins, CloudBees CI extends Jenkins with enterprise functionality that embeds best practices, rapid onboarding, security and compliance.

Read about how Autodesk saw a 10x productivity increase with CloudBees CI.

About

CloudBees is the industry's leading DevOps technology platform delivering the world's first end-to-end continuous software delivery management system. CloudBees enables developers to focus on what they do best: Build stuff that matters — while providing peace of mind to management with powerful risk mitigation, compliance and governance tools.

Used by many of the Fortune 100, CloudBees is helping thousands of companies harness the power of continuous everything and gets them on the fastest path from great idea, to great software, to amazing customer experiences, to being a business that changes lives.

Visit CloudBees at www.cloudbees.com.

CloudBees, Inc.
4 North Second Street | Suite 1270
San Jose, CA 95113
United States
www.cloudbees.com
info@cloudbees.com