



How to restore Nexus Repository usage

This guide is designed to help you when you have encountered the usage limits of Nexus Repository Community Edition. It includes information on three main topics:

- ▶ Short-term strategies for getting usage under the limits as quickly as possible
- ▶ More drastic options for keeping usage low in the face of organic growth in your development organization
- ▶ Guidance on determining whether the Community or Pro edition is most appropriate for your organization

Table of Contents

Usage Limits in Community Edition	3
Symptoms of Exceeding the Usage Maximums	3
Monitoring Usage	4
Typical Reasons for Exceeding Community Edition Usage Limits	4
Temporary Request Rate Spike	4
Runaway Reader	4
Runaway Publisher	5
Insufficient Cleanup Policies	5
Organic Growth in Usage over Time	6
Summary of Immediate Actions	7
More Drastic Strategies for Reducing Use	7
Ongoing Monitoring of Usage Levels	8
Choosing Nexus Repository Community or Pro Edition	8

Usage Limits in Community Edition

Nexus Repository is used by over 100,000 organizations around the world to manage software binaries in the development process. It accelerates development by serving as:

- ▶ A single ingestion route for open source and vendor components into the SDLC
- ▶ A component cache for fast, repeatable builds
- ▶ A secure, scalable system of record for internally produced components and applications as they move from development to production
- ▶ A scalable delivery mechanism for applications to downstream consumers and customers

USAGE MAXIMUMS

The Community Edition is a good choice for small teams, or organizations that are building a high performance SDLC. Community Edition has built-in limits that restrict usage to:

- ▶ **200,000 requests** in any 24-hour period. A request is an HTTP request made by a browser, command-line tool, or other automation to one of Nexus Repository's format-specific REST endpoints, commonly to store, retrieve, or search for a binary.
- ▶ **100,000 components** being managed. A component is one or more software binaries and supporting files that share an identifying coordinate (group, name, and version).

For the small percentage of larger or high-use teams that exceed these limits, Nexus Repository Pro is required to ensure that mission-critical systems have the support and enterprise features they need. This aligns our free Community Edition tier with teams and smaller organizations, the Pro tier with larger high-usage enterprises, and ensures a sustainable development model for small and large organizations alike.

Symptoms of Exceeding the Usage Maximums

When Community Edition usage crosses the usage threshold of either the Request in 24 Hours or Total Components Managed, any further write requests are blocked. Any HTTP requests to publish a new component (whether publishing one, or requesting one not currently cached) will be rejected with an HTTP error. Once both usage metrics drop below the limits, writes can be performed once more.

To avoid surprises, Community Edition offers a 45-day grace period before any blocking will occur: the grace period begins the first time that usage is over the limits. After the grace period expires, blocking only occurs when usage is above the limits.

Symptoms of this blocking may include build failures from developer machines or CI/CD agents, as well as errors when attempting to manually publish components into Nexus Repository.

Any components previously stored in Nexus Repository will **continue to be retrievable**. REST endpoints for configuring Nexus Repository, searching or retrieving of components will still function normally.

Monitoring Usage

Administrators of Community Edition deployments should monitor usage on a regular basis to understand how usage limits may affect their organization. [Information on usage monitoring capabilities](#) is available online.

Whenever usage is near or over limits, Nexus Repository adds prominent warnings to its usage dashboard and other parts of its administrative user interface.

Typical Reasons for Exceeding Community Edition Usage Limits

There are a number of common reasons for exceeding Community Edition's usage limits. Your strategy for handling the excess usage and preventing blocking will depend on that reason.

When usage climbs above the limits for the first time (regardless of the reason), the 45-day grace period begins. No blocking will occur during the grace period, regardless of usage levels. Once the grace period elapses, blocking will take place whenever usage is above the Community Edition limits.

Temporary Request Rate Spike

If your Nexus Repository instance frequently operates near the daily request rate limits of Community Edition, it's possible that the limits have been tripped by an unexpected peak of activity from developers, third-party users or automated build systems.

In this situation, the only action that is required is to wait until Requests in the Last 24 hours falls below the limit again. (Repository's 24-hour request totals are recalculated hourly.)

Runaway Reader

Most traffic to busy Nexus Repository instances comes from automated systems rather than directly from human users. A misconfigured or malfunctioning external script or CI/CD job could generate a sustained load that exceeds Community Edition's limits. In this case, Nexus Repository will not become fully usable until the culprit is identified and stopped.

Telling the difference between intentional traffic and wasteful is beyond what Nexus Repository can do—what seems excessive in one organization might be standard practice in another. However, if you suspect a runaway reader, reviewing [Nexus Repository's request logs](#) can reveal patterns of repetitive or low-value activity that should be stopped.

If you find and stop a runaway reader, continue to review Nexus Repository's usage dashboard to ensure that request levels are falling back within limits.

Runaway Publisher

Similar to a runaway reader, a runaway publisher creates a lot of components inside Nexus Repository. This may be a misconfigured nightly build that has started publishing continuously, or it may be an unusual use of Nexus Repository to store many tiny files without bundling them into an archive (e.g. JSON extracts, or snapshots of a large web site uploaded as separate files instead of as a .zip file).

Because binary repository use cases are diverse, Repository can't tell essential components from low-value components on its own. However, a review of request logs can reveal repetitive patterns of excessive PUT or POST from a specific user and/or a pattern of filenames.

If you find and stop a runaway publisher, you must still remove the files it created from Nexus Repository in order to restore usage. There are several methods:

1. If the runaway publisher has been active for some time, Cleanup Policies (see below) may be effective at removing unused files without having to target them by name.
2. If you know the paths of low-value files from request logs, you can delete them using the Browse view in the administrative user interface.
3. You can also remove files by finding them in the REST Search API, and then removing them one at a time from the REST Component API.

Insufficient Cleanup Policies

Nexus Repository comes equipped with cleanup policies that can prevent the buildup of low-value components that waste storage space. If these are not set up, Nexus Repository can reach usage limits as a byproduct of normal build activity over time.

If Nexus Repository is hitting its Total Components limit, check to see that you have adequate cleanup policies configured. If you do not, implement a cleanup policy that removes unused files.

1. [Perform a full backup](#) of Nexus Repository, including the embedded or external database and all the binaries that it manages. Since you may need to restore usage quickly, this step is important to ensure that you don't lose any components that turn out to be essential later.
2. Implement a [cleanup policy](#) that targets one of the highly used hosted or proxy repositories inside Nexus Repository. Start with a conservative time horizon (e.g. 2 years).
3. Cleanup policies are carried out by an automated task that runs every hour.
4. Monitor your usage dashboard to see if the component count is dropping. The specific cleanup actions will be listed in a) the audit log, and b) the task-specific logs created by the cleanup task.
5. If the component count is not dropping, consider the following actions:
 - Adjust the cleanup policy to a less conservative time horizon
 - Apply the cleanup policy to more proxy and/or hosted repositories

Organic Growth in Usage over Time

If you cannot identify that the limits were reached because of temporary request spikes, runaway readers, runaway writers, or insufficient cleanup policies, it may be that your organization has outgrown the Community Edition limits.

In general, increased usage over time is normal. Apart from simply adding more developers to an existing organization, a number of software development best practices tend to increase usage of a binary repository:

MORE USE CASES

- ▶ Using the binary repository directly to support a production deployment system
- ▶ Distribution of components or applications to third parties outside the development team

SHORTER FEEDBACK CYCLES FOR RAPID INNOVATION

- ▶ Building more frequently for faster feedback cycles from tests or customers
- ▶ Cascading builds to rapidly detect bugs caused by changes to core components that are widely used by other applications
- ▶ Publishing/releasing more frequently to deliver new features and fixes more quickly to downstream consumers

APPLICATION COMPLEXITY

- ▶ Multi-language applications that draw components from multiple software ecosystems (e.g. maven and npm), and/or bundling for multiple deployment ecosystems (e.g. .war files, RPMs, containers)
- ▶ Decomposition of monolithic applications into smaller services
- ▶ Improved sharing of components internally across teams for better reuse

APPLICATION SECURITY

- ▶ Increased focus on eliminating CVEs from in open source components can cause dependencies to be upgraded more regularly, especially if an automated dependency management system is used
- ▶ External repository scanners that regularly download available components for scanning purposes
- ▶ Reducing 'shadow downloads' that bypass the single ingestion point for outside components, to reduce the risk of non-repeatable builds and supply chain malware

Summary of Immediate Actions

- ▶ Check Nexus Repository's usage dashboard to see which limit has been exceeded, **Requests per 24 Hours** or **Total Components**.
- ▶ **Requests per 24 Hours** exceeds 200,000
 - ▶ Wait to see if a high request rate subsides, indicating a **temporary request spike** that will resolve on its own.
 - ▶ Review request logs to look for frequent, repetitive fetches that could indicate a **runaway reader**.
- ▶ **Total Components** exceeds 100,000
 - ▶ Check request logs to see if a **runaway publisher** is writing many low-value files to Nexus Repository.
 - ▶ Check for **insufficient cleanup policies** that may have allowed an accumulation of unused components.
- ▶ Consider more drastic strategies that **reverse organic usage increases**

More Drastic Strategies for Reducing Use

Because many positive SDLC practices cause higher use of binary repositories, the following drastic strategies with caution. Sonatype recommends using these strategies on a temporary basis, if at all, as they can hamper productivity or reduce application security, or increase the total cost of ownership.

- ▶ Disable **anonymous access**. This doesn't directly reduce usage, but it can make it easier to identify excessive read load by forcing all readers to authenticate. It can have the indirect consequence of blocking any wasteful usage from automation that doesn't have an active maintainer.
- ▶ Reduce the frequency of any external process that **scans the entire repository**, such as third party security solutions that download all binaries.
- ▶ Reduce the **frequency of regular builds**. Consider reducing the frequency of nightly builds to alternating nights.
- ▶ Review the necessity of build jobs that have a **build-on-commit strategy**.
- ▶ **Partition your Nexus Repository** into two or more instances. This approach should be balanced against the increased cost of ownership of maintaining multiple instances with different configuration.

Nexus Repository's Community Edition limits come with a built-in 45-day grace period to give your organization time to choose a course of action before blocking begins. Sonatype recommends exploring your usage and determining a usage strategy during the grace period to avoid these drastic measures.

Ongoing Monitoring of Usage Levels

Once usage has been restored below limits and your pipelines are running smoothly, you should continuously monitor Nexus Repository usage levels to prevent unexpected blocking when the Community Edition limits are exceeded.

Choosing Nexus Repository Community or Pro Edition

If you have eliminated low-value usage and usage levels are still near the Community Edition limits, it may be appropriate for your organization to adopt Nexus Repository Pro.

Upgrading to Nexus Repository Pro removes the Community Edition limits, and is as easy as uploading a license file to your existing instance. Until you're ready to make use of more advanced deployment options, no other changes to your environment are necessary.

In addition to expanded usage, Nexus Repository Pro has a number of features designed to help busy devops teams support growing, high-performance development organizations.

- **Enterprise Scale:** Nexus Repository Pro has the enterprise features to support mission-critical operations at scale. Capabilities such as high availability and resilient failover ensure continuous delivery and minimize both production and developer tooling downtime.
- **DevOps Flexibility:** Advanced cleanup options, repository import/export, and content replication facilitate distributed development and improve collaboration across teams.
- **Security and Control:** With advanced authentication options like SAML/SSO and auth tokens, Sonatype Nexus Repository Pro improves security and simplifies user management.
- **Open Source Supply Chain Security:** Nexus Repository Pro's advanced repository health check provides deeper insights into the security and quality of repositories, identifying vulnerabilities and quality issues early to mitigate security risks and avoid rework.
- **Complete Ecosystem Coverage:** Nexus Repository Pro supports all major package formats, build tools, and CI servers, offering a universal binary repository. This saves time and reduces complexity, allowing developers to focus on core tasks.
- **Enterprise Support from Sonatype:** Nexus Repository Pro customers benefit from world-class support and a dedicated customer success team. This ensures quick resolution of issues, expert assistance, faster adoption, and optimized tool usage, maximizing return on investment and helping achieve business goals.

These capabilities are designed to make it easy to support a growing development organization with a binary repository that expands to any scale.

For more information about Nexus Repository Pro, or which edition is best for your organization, [please contact Sonatype](#).



Sonatype is the software supply chain management company. We empower developers and security professionals with intelligent tools to innovate more securely at scale. Our platform addresses every element of an organization's entire software development life cycle, including third-party open source code, first-party source code, infrastructure as code, and containerized code. Sonatype identifies critical security vulnerabilities and code quality issues and reports results directly to developers when they can most effectively fix them. This helps organizations develop consistently high-quality, secure software which fully meets their business needs and those of their end-customers and partners. More than 2,000 organizations, including 70% of the Fortune 100, and 15 million software developers already rely on our tools and guidance to help them deliver and maintain exceptional and secure software. For more information, please visit [Sonatype.com](https://www.sonatype.com), or connect with us on [Facebook](#), [Twitter](#), or [LinkedIn](#).

Headquarters

8161 Maple Lawn Blvd,
Suite 250
Fulton, MD 20759
USA • 1.877.866.2836

European Office

168 Shoreditch High
St, 5th Fl
London E1 6JE
United Kingdom

APAC Office

60 Martin Place,
Level 1
Sydney 2000, NSW
Australia

Sonatype Inc.

www.sonatype.com
Copyright 2023
All Rights Reserved.