ATLASSIAN

Scaling up, out, and around with Bitbucket

A guide to scaling Bitbucket

When it comes to scale, the most demanding load many Bitbucket instances encounter is often due to managing Git hosting operations (simultaneous user-initiated commands, like git clone, fetch, and push.)

This is because when you run a Git command that has to communicate with a remote repository, your Git client opens one or more connections to your Bitbucket instance depending on whether you are using HTTP or SSH. When each of these connection reaches the backend server (after authentication and other processing) the connection executes a Git process and streams its standard input, output, and error output back to the client.

These Git processes on the server are CPU and memory intensive, especially when they generate packfiles to transfer repository contents over the network. By comparison, most other kinds of operations you can perform against a Bitbucket instance, like browsing and interacting with pull requests, are generally much lighter and faster. The resource consumption of a single git clone may not seem so bad, but when you have hundreds or thousands of users doing these operations concurrently, the CPU and memory usage add up quickly.

Continuous integration systems like Bamboo and Jenkins are also famous (or, perhaps, infamous) for making a lot of Git hosting operations against a Bitbucket instance, to clone and fetch from (and sometimes push to) repositories for builds and tests. Load from build agents tends to come in bursts: builds configured with many parallel or cascading stages can generate massive “storms” of Git hosting operations all at once. Many continuous integration systems can also spin up large numbers of elastic build agents in a cloud environment like Amazon EC2. This can provide vast amounts of CPU resources in a short time to get through the build queue, but can overwhelm a Bitbucket instance that’s not provisioned with enough resources to handle the peaks. So how can you scale Bitbucket Data Center to combat all of these challenges?

We’ve outlined a number of scenarios you might be experiencing and recommendations on different tactics you can use to scale.

IMPORTANT:
Before jumping into any of the specific use cases, the first thing to consider when it comes to scale is your hardware. Take the time to examine your infrastructure to identify ways to increase capacity or resources by provisioning things like CPU, RAM, disk capacity or IOPS.

To help you determine the right infrastructure to accommodate the size of your environment and meet your organization’s needs, we’ve compiled infrastructure recommendations for Bitbucket Data Center. Use these recommendations as guidelines when it comes time to look for ways to scale up.

Increase in concurrent users

Are you experiencing degraded performance at peak times? Is a new office opening or have you seen a recent growth in the number of users due to other organizational changes? The number of concurrent users accessing your system can grow for a number of reasons, and this growth can quickly negatively impact your instance. In order to swiftly respond to changes in concurrent users, you can take advantage of horizontal scaling.

Horizontal scaling

Scale your Bitbucket Data Center cluster horizontally by adding machines to form a cluster of many nodes, behind a load balancer. The load balancer included in this setup will route incoming traffic to the nodes in a cluster according to the load they are experiencing, allowing for more concurrent users.

Degraded performance for distributed teams

Finding an enterprise organization without distributed teams is near impossible today. As the size of your distributed team(s) grows or the number of offices increases, the physical distance between these users and the primary instance can have a negative impact on performance. To solve for this, Bitbucket has a couple of tools.

Smart Mirroring

Smart mirroring allows you to set up live mirror nodes, which are copies of repositories in remote locations. The mirrors automatically keep all repositories hosted on them in sync with the primary Bitbucket Data Center instance. Users in those remote locations may clone and fetch repositories from the mirror and get identical content, only faster.

Content delivery network (CDN)

Setting up a CDN is another way to help you scale globally. As end users access your site, static assets will be cached on the edge server closest to them - allowing content to be delivered from the location closest to them, rather than waiting for it to return from the primary location.

Multiple instances

We hear from many customers that they own too many instances. Sometimes they'd like to keep these instances separate due to different business units using them or for other use cases. But in other use cases, they are interested in consolidating instances to improve manageability. Learn how Bitbucket Data Center helps simplify this complex task.

Data Center migration tool

Data Center Migration is a tool for admins that can be used to consolidate multiple Bitbucket instances, move from a Bitbucket Server to a Bitbucket Data Center instance, or selectively export/import projects and repositories from one Bitbucket Data Center instance to another. With this tool, Git data can be imported or exported into Bitbucket Data Center from another Bitbucket Server or Data Center deployment, along with pull requests, comments and attachment history.

Increase in API usage

CI, apps and scripts for custom use cases request or update data in Bitbucket Data Center. As your tool chain and team grows, the number of Git requests from REST API will increase and subsequently place significant load on your Bitbucket Data Center instance.

Traffic shaping

To further optimize scale and performance, you can tailor how you segment traffic across your application nodes using traffic shaping. Traffic shaping allows you to categorize and prioritize particular types of traffic and redirect that traffic to a specific node in your cluster. In this case, you could direct external REST API traffic to a dedicated node or a set of nodes.

Adaptive throttling

Use adaptive throttling to help your instance adapt to the stress it is under due to Git hosting operations. It allows Bitbucket to examine the total physical memory on the machine and defines the number of Git operations that can be executed by monitoring the system resources. Compared to fixed throttling, this strategy gives you the best of both worlds. When your machine has capacity to spare, Bitbucket allows as many concurrent Git hosting operations that you can throw at it. But if it starts to strain under the load, Bitbucket dynamically detects this and reduces the limit, protecting the overall responsiveness of the whole system.

Rate limiting

It isn't just Git hosting operations that can impact your instance. Automated integrations and scripts can also affect or take down your instance. With rate limiting, you can now control how many HTTP requests (e.g. REST API requests) automations and scripts can make, and how often they can make them - improving performance and team productivity (and hopefully for admins, more sleep too).

Webhooks instead of polling for changes

To avoid a suboptimal experience that you might encounter when polling for new code changes in Bitbucket on a fixed time period, we recommend using webhooks. You can select the event and a webhook will be sent to the specified target system. This means not only will the target system be notified quickly but you also won't place additional load on your Bitbucket Data Center instance from polling requests.

Managing or cloning large repositories

Repositories can easily become quite large, usually due to accumulating a long history or a large number of binary assets. And it can be even worse if the repo contains old, deprecated binary artifacts. Trying to clone these large repositories can become incredibly challenging. There are a few tactics to handle this.

Adding additional storage

As the repository data in your Bitbucket Data Center grows it may become too large to hold in a single storage location. When this happens, you'll need to add additional storage space. You can increase the storage space allocated to your shared home directory or add additional data stores. Increasing the storage space allocated to your shared home directory is simplest option. If this is not an option for you (for example, because of a company policy limiting the size of disks or partitions), you should add a data store.

Shallow cloning

You can only pull down the latest n commits of the repo's history with Git's shallow clone. With improvements to shallow clone in recent years, you can even push and pull to repositories from a shallow clone today.

Git filter branch

For the huge repositories that have lots of binary cruft committed by mistake, or old assets that aren't needed anymore, a great solution is to use git filter-branch. The command lets you walk through the entire history of the project filtering out, modifying, and skipping files according to predefined patterns. It can be very powerful once you've identified where your repo is heavy. But there are helper scripts available to help with this.

TIP:
If you're planning to carry out a cleanup action using *git filter-branch*, make sure to alert your team, plan a short freeze while the operation is carried out, and then notify everyone that they should clone the repository again.

Git LFS

The best solution of all is Git LFS, which improves how large files are handled. Git LFS is an extension that replaces files with tiny text pointers that are stored on a remote server instead of in their repository. As you can imagine, this dramatically reduces the time it takes to clone a repo.

Storage optimizations

- **Hardware and environment configuration:** Invest in a high performance file server such as a SAN, NAS, or RAID server and use a high-speed LAN such as 10 GB Ethernet or Fibre Channel within your cluster. The file server should run on a dedicated machine and you should avoid multi-tenanting your Bitbucket Data Center file server with other services on the same physical machine.
- **Adjust your SCM cache plugin:** Where possible, enable the SCM Cache plugin (already bundled in Bitbucket Data Center) with as much disk or SSD space on your cluster nodes as you can. An effective SCM Cache can greatly reduce load on your shared file server.

Large number of CI builds, distributed builds or build spikes

As CI/CD workflows become more sophisticated, frequent (many times a day), and critical, a simple change in a file or repository can trigger hundreds of builds, test dependent libraries, or services. These large numbers of CI builds can lead to large numbers of clone/fetch requests to Bitbucket, then which degrades the performance of an instance. Another common thing we see is when large, distributed development teams with a large number of CI builds experience surges of long running clone and fetch requests. This slows down their ability to run builds in a timely manner or even make use of their mirrors due to the traffic jam their CI tool has created. Here are some of the strategies to deal with large number of builds including build spikes.

Smart mirror farms

Cluster mirrors into “farms” grouped together behind a load balancer to increase your teams’ CI/CD capacity, reduce the time your distributed teams spend waiting, and scale as the size of your distributed teams grow.

Shallow cloning

Pull down the latest n commits of the repo’s history with Git’s shallow clone for faster builds rather than an entire history. Things have gotten even faster and easier with improvements to shallow clone in recent years. Today, you can even push and pull to repositories from a shallow clone.

Adaptive throttling

You can also use adaptive throttling, a tool we mentioned earlier, to combat this. With adaptive throttling, Bitbucket examines the total physical memory on the machine and determines a maximum ticket number that the machine can safely support given an estimate of how much memory a hosting operation consumes, how much memory Bitbucket needs, and how much search needs

Have questions?

Learn more at atlassian.com/software/bitbucket/enterprise/data-center

ATLASSIAN