

This document provides guidelines for deploying the Nexus Pro repository manager in your organization. This first section provides general deployment guidelines. The second section provides further details on each of the items that will impact the repository architecture.

General Guidelines

The following are general guidelines one should follow when designing a repository architecture.

1. Have at least one repository manager in each location that has a CI system, a release manager or more than a handful of developers.
2. Locate the hosted repository for each project near the producers of artifacts.
3. Consider the underlying networking infrastructure when deciding how to configure your proxies.
4. Give more weight to the location of CI servers because on average these are producing many orders of magnitude more artifacts than release managers.
5. The relative importance of controlling which components are available in each repository will impact the repository architecture. A centralized architecture will provide greater control over components but may impact performance.
6. Implement a highly available solution when you need to prevent development delays.
7. Utilize either physical or virtual machines as both work well with Nexus. However, we do not recommend using NFS.

Items that Impact the Repository Architecture

This section details specific items that you should consider when designing the repository architecture.

The Types and Number of Users at Each Location

This is one of the primary items that determine where you should locate repositories. We strongly recommend locating a repository manager in each office that has a CI system, release manager or more than a handful of developers.

Artifact Production

Generally, only release managers will produce release artifacts, and CI servers will produce snapshots. The geographic dispersion of producers is important because it dictates whether you have a federated model of hosted repositories or a star pattern.

Each project will have a master hosted repository where artifacts are staged and deployed. Other repositories will be configured as read-only proxies of this master. The master repository for each project is typically co-located with the producers for that project.

If all of your producers are in a single location, then you can designate a single master repository for your organization and set up proxy repository at other locations. This is a star type of pattern.

If the producers for different projects are located in multiple places, then each location will have a master hosted repository for the locally produced artifacts. The locations that need to consume, but not produce, artifacts from each project will typically have a proxy repository. This is a federated pattern, but another way to look at this is as a set of overlapping star patterns.

Networking Infrastructure

The underlying data networking infrastructure will impact the repository architecture. You need to be sure that the underlying pipes can handle the expected data traffic both between repositories and between repositories and users.

If you have a star networking topology that requires all data traffic to go through a central location, then you might want to create an intermediate proxy repository in the central location. Remote offices will not proxy directly with each other, but rather through the intermediate proxy.

On the other hand, if you have fast network connections between all your offices, then you can set up direct proxies between locations as needed without worrying about the underlying infrastructure.

Internet Connectivity

If a location has a direct connection to the Internet, we recommend configuring the local repository manager to access the Central Repository directly as this provides the best performance, subject to your need to manage components (see the next section). On the other hand, if a location accesses the Internet over an enterprise WAN, then you will probably want to proxy Central from within the enterprise network.

Component Management vs. Performance

The relative importance of component management versus performance will impact the architecture.

The fastest performance can be achieved by locating hosted repositories closest to producers, and locally proxying all other repositories, including Central. Each location with an adequate Internet connection would proxy Central directly.

However, if you need to control which open source components are available in each repository, then you'll want to create a proxy repository for Central that contains only vetted components. Each office will proxy this vetted repository. The Nexus Pro Procurement feature can help you manage the process of white and/or black listing components.

If your goal is to ensure that developers do not utilize problematic components, then we recommend Sonatype Insight. Insight works throughout the development process and integrates with all of your key tools. It provides the information developers need, when and where they need it, to choose the best

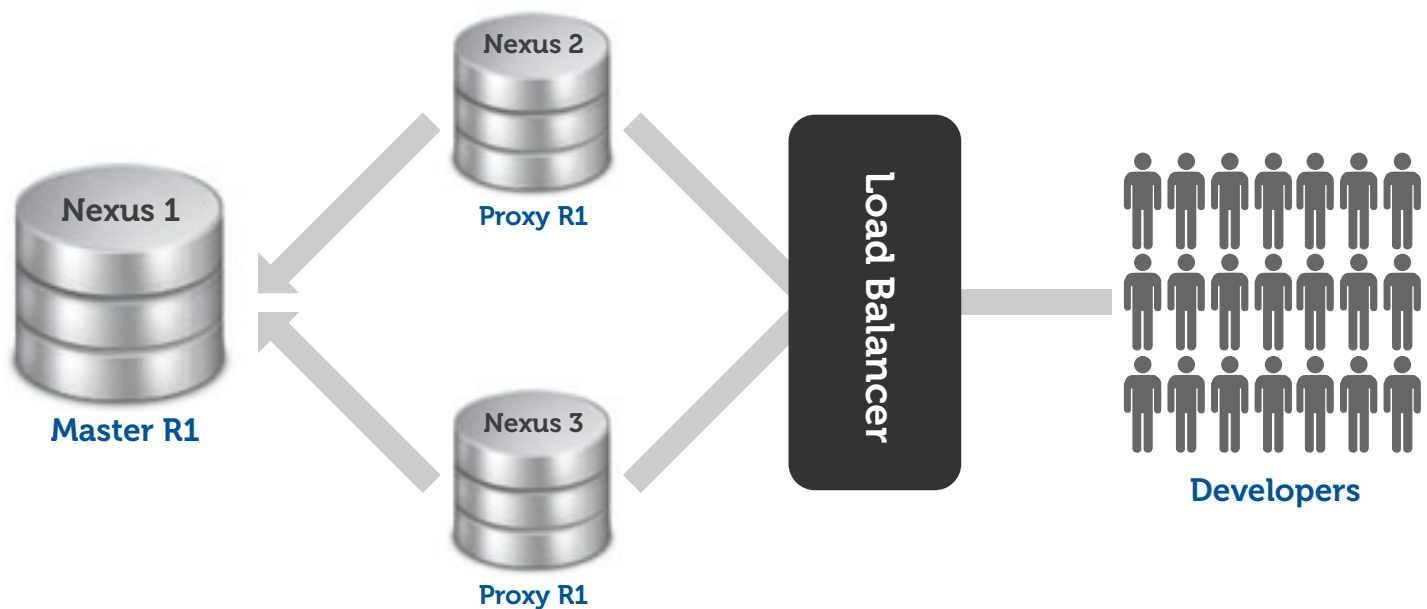
components. Insight also provides the controls you need to ensure that problematic components do not make it into production applications.

High Availability

When your development teams are working on mission critical projects that cannot suffer a delay, you'll want to design the architecture to ensure high availability. The architectures described in this section work best with the improved proxy capability available in Nexus Pro 2.0 and later releases.

High Availability for Reading Artifacts

The following configuration will ensure that components are always available for consumption using Active/Active "reads" with load balancing:

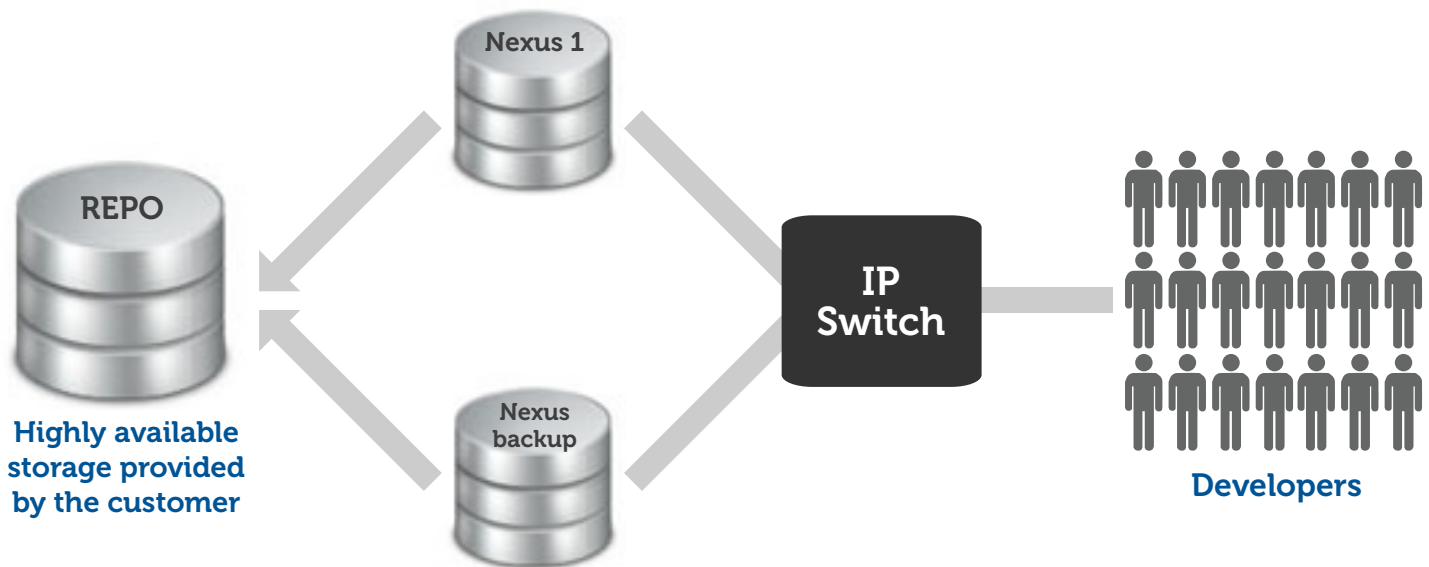


- You'll need 2 or more Nexus repository managers and a load balancer. Most users deploy 3 such servers.
- The Nexus master contains the hosted repositories.
- The other two Nexus repository managers are configured as proxy repositories of the master.
- The load balancer sits in front of the three repository managers and balances the read load between them.

This configuration provides a very robust solution for ensuring that artifacts are always available for reading. This is the most important function to protect as the read load typically outweighs the write load by many orders of magnitude (thousands of developers are typically reading, only a very small set are writing).

High Availability for Creating Artifacts

The following configuration will increase the availability of repository "writes" by using redundant, stand-by servers backed by a highly available file system.



- You'll need to configure 2 (or more) Nexus servers identically, sharing the same file system. Only 1 Nexus can be active at a time
- The file system should be highly available using an off the shelf solution
- When Nexus 1 fails, Nexus 1 backup must be activated through a configuration change
- An IP switch (or similar device) enables clients to continue using the same Name/IP address for the Nexus server

Virtual or Physical Machine

Nexus supports both physical and virtual machines equally well. Nexus doesn't require a lot of CPU or RAM to do its job effectively. At Sonatype, we've moved all of our managed forges over to virtual machines with the following specifications:

- 2 CPUs
- 3GB RAM
- 400GB disk (this is completely dependent on your repository contents)
- RHEL 5.6 x64 (this is the Contegix Preferred OS and they manage the systems for us)
- Java 1.6 x64 with 1GB Heap
- The virtual disk is located on a SAN connected with iSCSI over 1GBE
- For I/O performance, we recommend a redundant solution that maximizes disk spindles, while maintaining fault tolerance. We use RAID 50 in our SAN. A RAID 50 combines the straight block-level striping of RAID 0 with the distributed parity of RAID 5. It is a RAID 0 array striped across RAID 5 elements. It requires at least 6 drives.

These systems serve requests on the order of 1,400-2,500 requests per minute. Above that, the system typically needs to scale up in terms of network and IO optimization. Increasing the number of CPUs and amount of RAM can typically help as well.

We do not recommend using NFS to mount a virtual disk or the working folder as many customers have had trouble with locking and corrupted indexes. iSCSI is working very well for us and for many of our customers.