sonatype | **Blog**

Products ▾   Success Stories   Solutions ▾   Resources ▾

About ▾   Blog

Using Nexus 3 as Your Repository – Part 3: Docker Images

March 06, 2017 By   **Rafael Eyng**

This is the third and last part of a series of posts on Nexus 3 and how to use it as repository for several technologies. (**Part 1**   .   **Part 2** .)

NOTE: We're constantly improving the Nexus Repository. This blog post may no longer be accurate. For the latest instructions, please click here to visit guides.sonatype.com.

Installation

Check out the first part of this series to see how we installed and ran Nexus 3 using a single docker command. Just do that and the installation is done.

AUTHOR POSTS

TOPIC POSTS

Using Nexus 3 as Your Repository – Part 3: Docker Images
🖉 Rafael Eyng

Using Nexus 3 as Your Repository – Part 2: npm Packages
🖉 Rafael Eyng

Using Nexus 3 as Your Repository – Part 1: Maven Artifacts
🖉 Rafael Eyng

Configuring Nexus as a Docker repo

What we will do:
– create a private (hosted) repository for our own images
– create a proxy repository pointing to Docker Hub
– create a group repository to provide all the above repos under a single URL

I suggest you to create a new blob store for each new repo you want to create. That way, the data for every repo will be in a different folder in /nexus-data (inside the Docker container). But this is not mandatory for it to work.

By default, the Docker client communicates with the repo using HTTPS. In my use case I had to configure it with HTTP, because we didn't have the certificate nor the knowledge on how to obtain it.

Important to notice: the Docker repo requires 2 different ports. We are going to use 8082 for pull from the proxy repo and 8083 for pull and push to the private repo.

I had some problems with slightly older versions of Docker, so I strongly suggesting you to start with the version that I've tested with, that is 1.12.3.

private repo

A repository for Docker images that your team creates.

Create a new Docker (hosted) repository and configure it like:

Deployment policy:

Controls if deployments of and updates to artifacts are allowed

Allow redeploy

proxy repo

A repository that proxies everything you download from the official registry, Docker Hub. Next time you download the same dependency, it will be cached in your Nexus.

Create a new Docker (proxy) repository and configure it like:

Repositories / Select Recipe / Create Repository: docker (proxy)

Name:
A unique identifier for this repository
docker-hub

Online: ✓ If checked, the repository accepts incoming requests

Repository Connectors

Connectors allow Docker clients to connect directly to hosted registries, but are not always required. Consult our documentation for which connector is appropriate for your use case.

HTTP:
Create an HTTP connector at specified port. Normally used if the server is behind a secure proxy.

HTTPS:
Create an HTTPS connector at specified port. Normally used if the server is configured for https.

Docker Registry API Support

Enable Docker V1 API:

Allow clients to use the V1 API to interact with this Repository.

Proxy

Remote storage:
Location of the remote repository being proxied
https://registry-1.docker.io

Use the Nexus truststore:
Use certificates stored in the Nexus truststore to connect to external systems   ⚙ View certificate

Docker Index:

○ Use proxy registry (specified above)

● Use Docker Hub

○ Custom index

Location of Docker index
https://index.docker.io/

Use the Nexus truststore:

☐ Use certificates stored in the Nexus truststore to connect to external systems   ⚙ View certificate

Blocked:

☐ Block outbound connections on the repository

Auto blocking enabled:

☑ Auto-block outbound connections on the repository if remote peer is detected as unreachable/unresponsive

Maximum component age:

How long (in minutes) to cache artifacts before rechecking the remote repository. Release repositories should use -1.

| 1440 | ⇅ |

Maximum metadata age:

How long (in minutes) to cache metadata before rechecking the remote repository.

| 1440 | ⇅ |

Storage

Blob store:

Blob store used to store asset contents

| docker-hub | ▾ |

Strict Content Type Validation:

☑ Validate that all content uploaded to this repository is of a MIME type appropriate for the repository format

Negative Cache

Not found cache enabled:

☑ Cache responses for content not present in the proxied repository

Not found cache TTL:

How long to cache the fact that a file was not found in the repository (in minutes)

| 1440 | ⇅ |

HTTP

☐ **Authentication**

☐ **HTTP request settings**

group repo

This will group all the above repos and provide you a single URL to configure your clients to download from to.

Create a new Docker (group) repository and configure it like:

🗄 **Repositories** / 🗄 Select Recipe / 🗄 Create Repository: docker (group)

Name:	A unique identifier for this repository
	docker-group
Online:	☑ If checked, the repository accepts incoming requests

Repository Connectors

Connectors allow Docker clients to connect directly to hosted registries, but are not always required. Consult our documentation for which connector is appropriate for your use case.

HTTP:

Create an HTTP connector at specified port. Normally used if the server is behind a secure proxy.

| ☑ 8082 | ⇅ |

HTTPS:

Create an HTTPS connector at specified port. Normally used if the server is configured for https.

| ☐ | ⇅ |

Docker Registry API Support

Enable Docker V1 API:

☐ Allow clients to use the V1 API to interact with this Repository.

Storage

Blob store:

Blob store used to store asset contents

docker-group

Strict Content Type Validation:

☑ Validate that all content uploaded to this repository is of a MIME type appropriate for the repository format

Group

Member repositories:

Select and order the repositories that are part of this group

Available   Members

Filter   ⊗   docker-hub

               docker-private

You can create as many repos as you need and group them all in the group repo.

This step is actually optional to use Nexus 3 as a Docker repository, because we can stick to pulling and pushing to the proxy and hosted repositories as will be discussed later.

Configuring your clients and projects to use your Nexus repos

To interact with your repo, the first thing is to configure the Docker daemon in your machine to accept working with HTTP instead of HTTPS.

How exactly to do this config depends on your operating system, so you should check **dockerd** documentation. On RHEL I did it putting this content in `/etc/docker/daemon.json`:

```
{
  "insecure-registries": [
    "your-repo:8082",
    "your-repo:8083"
  ],
  "disable-legacy-registry": true
}
```

You have to restart the daemon after setting this (`sudo systemctl restart docker`).

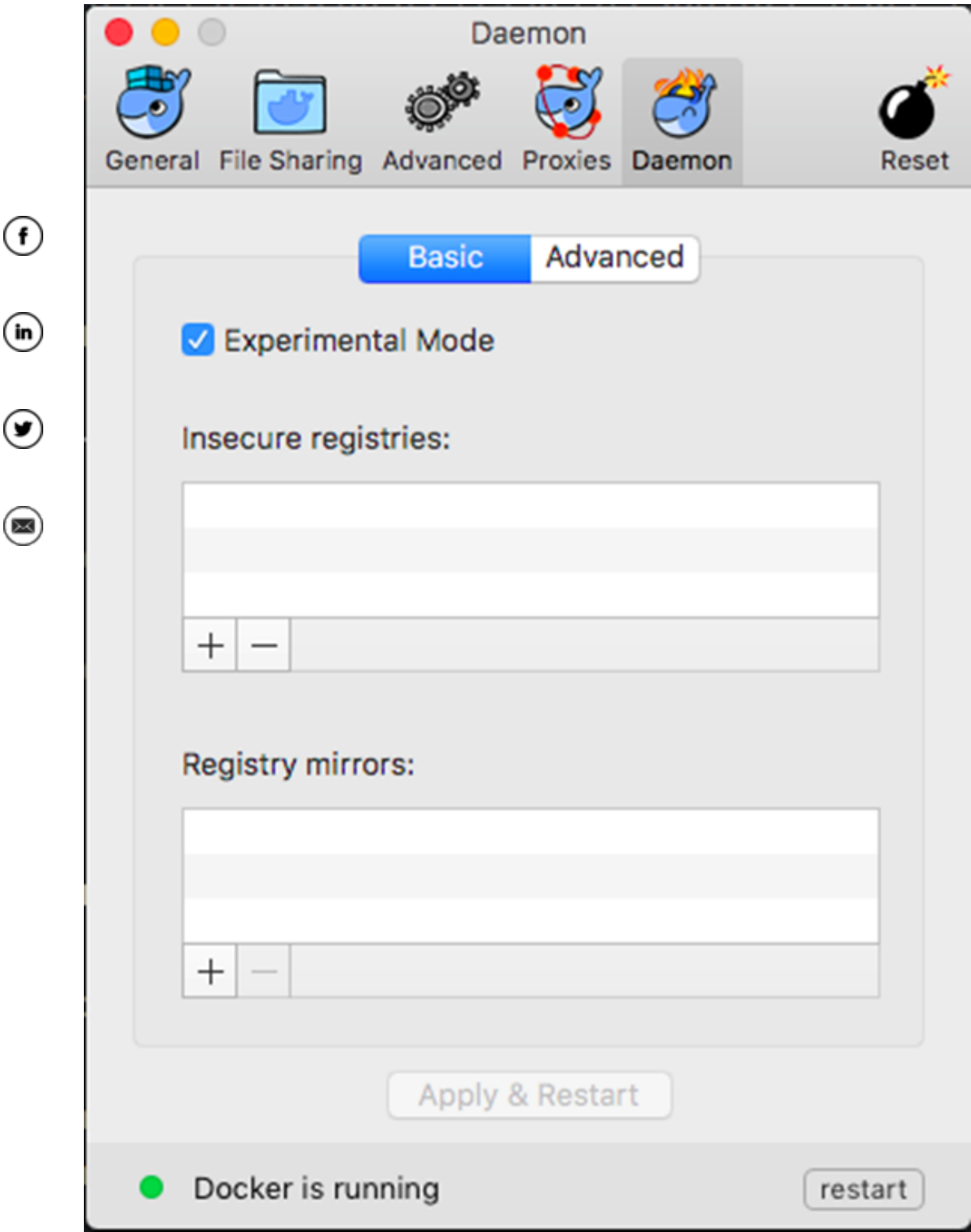On Windows or Mac you should config your deamon in a box like this:

Now we have to authenticate your machine to the repo with:

```
docker login -u admin -p admin123 your-repo:8082
docker login -u admin -p admin123 your-repo:8083
```

This will create an entry in   `~/.docker/config.json`:

```
{
        "auths": {
                "your-repo:8082": {
                        "auth": "YWRtaW46YWRtaW4xMjM="
                },
                "your-repo:8083": {
                        "auth": "YWRtaW46YWRtaW4xMjM="
                }
        }
}
```

To pull images from your repo, use (notice port **808** being used):

```
docker pull your-repo:8082/httpd:2.4-alpine
```

To push your own images to your repo, you have to tag the image with a tag that points to the repo. This is

Subscribe to the Blog ✕

Get the latest in DevSecOps and Open Source Governance delivered to your inbox each week!

Subscribe now

strange to me, since I was trying to think about Docker tags the same way I do about Git tags, but they seem be somewhat different (notice port **8083** being used):

```
docker tag your-own-image:1 your-repo:8083/your-own-image:1
docker push your-repo:8083/your-own-image:1
```

To pull your own images from the repo, you can use:

```
docker tag your-own-image:1 your-repo:8082/your-own-image:1
# or
docker tag your-own-image:1 your-repo:8083/your-own-image:1
```

Both ports will work. I suspect that is because using port 8083 will connect directly to the hosted repo, whilst using port 8082 will connect to the group repo, which contains the hosted repo. I suggest you to stick to port **8083** to avoid duplicate images in your machines. If you chose to stick with port 8083 to pull your own images, you probably could skip creating the group repo, if you prefer.

Tags: **Docker** , **Nexus Repository**   , **Product**

Written by   **Rafael Eyng**

---

aNeutrino aNeutrino   3/10/2018, 4:18:06 PM

Hi,
I have noticed that images are not working on this blog post.
can you fix them, please?

thx
aNeutrino

↩ Reply to *aNeutrino aNeutrino*

Sven Frinlan   3/13/2018, 8:05:26 PM

I'm seeing the same problem as aNeutrino and it effectively

ruins this whole post

↩ Reply to *Sven Frinlan*

Jeff Black   3/22/2018, 5:21:25 AM

Same problem still exists with the missing images. The links to Part 1 and 2 are also broken.

Jeff

↩ Reply to *Jeff Black*

Mister Anon   4/7/2018, 4:03:20 AM

I really hate the fact that this website makes web browser's title bar blink saying "YOU HAVE 2 NEW MESSAGES" when it's really just a damn marketing bot.

↩ Reply to *Mister Anon*

bhaskar sareddy   12/28/2018, 3:54:43 PM

Having a problem to login to the server.

$ docker login -u admin -p admin123 Repo URL:8082
WARNING! Using --password via the CLI is insecure. Use --password-stdin.
Error response from daemon: login attempt to http://Repo URL:8082/v2/ failed with status: 503 Service Unavailable

↩ Reply to *bhaskar sareddy*

co baebae   1/31/2019, 8:33:57 PM

I am having the same problem.
How did you solve it?

↩ Reply to *co baebae*

vpu hoff   5/13/2019, 10:40:16 PM

echo 'admin123'>pass.txt

```
cat pass.txt | docker login -u admin --password-stdin
RepoURL:8082
```

↩ Reply to *vpu hoff*

Sean Creighton   1/23/2019, 7:24:25 AM

Same, cannot login

net/http: request canceled while waiting for connection
(Client.Timeout exceeded while awaiting headers)

↩ Reply to *Sean Creighton*

Ivan Travnichev   4/4/2020, 5:08:09 PM

Have you managed to deal with it?

↩ Reply to *Ivan Travnichev*

Ricardo Chicangana   5/23/2019, 3:46:45 AM

Hi, thank you so much, all perfect

↩ Reply to *Ricardo Chicangana*

SIMON MILLER   6/20/2019, 8:19:18 PM

this: "I had some problems with slightly older versions of
Docker, so I strongly suggesting you to start with the version
that I've tested with, that is 1.12.3." was the big decider for me.
I was losing my mind trying to get it to work on an older
version of Docker. Thank you, Rafael

↩ Reply to *SIMON MILLER*

Richard Cross   10/22/2019, 5:52:13 PM

Is "docker search" supported for hosted repositories?

↩ Reply to *Richard Cross*

Mohamadreza Chalak   1/15/2020, 4:27:45 PM

In Docker 17.12, operations against registries supporting only the legacy v1 protocol are no longer supported.

The "disable-legacy-registry" configuration option has been removed and, when used, will produce an error on daemon startup.

based on https://docs.docker.com/engine/reference/commandline/dockerd/

↰ Reply to *Mohamadreza Chalak*

Volodymyr Burian   2/14/2020, 8:53:41 AM

Have a problem with log in getting 503
login attempt to http://(ip):8082/v2/ failed with status: 503 Service Unavailable

↰ Reply to *Volodymyr Burian*

Janis Liepins   3/17/2020, 8:57:24 PM

Hi,

Did you manage to find solution for this?
Having the same error

↰ Reply to *Janis Liepins*

Alejandro Otero   4/24/2020, 6:46:59 PM

Hi,

I had the same problem (now I have a different one) so I might be able to help.

- Are your running in Docker?
- If yes, make sure you add the connector port to yo[u]
Docker file, otherwise the host machine won't allow
to connect.
- Make sure that the user you are using has the cor[r]
permissions to the Docker registry.

After checking all of this now I encountered a differ[e]

error:

```
net/http: request canceled (Client.Timeout exceeded
while awaiting headers)
```

Hope to be of help!

↩ Reply to *Alejandro Otero*

---

jadran mestrovic   3/4/2020, 3:37:13 AM

Docker architecture enable that Docker runs both on Windows and macOS.

↩ Reply to *jadran mestrovic*

---

navin prasad kumar   4/15/2020, 1:47:56 AM

In my case, I have created three repos in nexus. After that, I've searched for /etc/docker/daemon.json in CentOS but it is not there. So I've created a file daemon.json and appended the values. But what I should I enter instead of your-repo in daemon.json?

↩ Reply to *navin prasad kumar*

First Name   *

Last Name   *

Email   *

Comment   *

protected by **reCAPTCHA**
Privacy · Terms

SUBMIT COMMENT

Products

Nexus Firewall

Nexus Lifecycle

Nexus Lifecycle
Foundation

Nexus Auditor

Nexus
Repository Pro

Nexus
Repository OSS

Nexus
Integrations

Pricing

Security Research

Nexus
Intelligence

CVE Insights

OSS Index

Nexus
Vulnerability
Scanner

Solutions

Developers

AppSec

DevSecOps

Legal &
Compliance

Government

Financial
Services

Manufacturing

Technology

Healthcare

Resources

Success Stories

White Papers

Webinars

Partners

Exchange

Training

Videos

Documentation

Community

Support

About

About Sonatype

Events

Management

Investors

Press Releases

Media

Coverage

Careers

Contact Us

Twitter

LinkedIn

Facebook

Instagram

YouTube

GitHub

Subscribe to the Blog ✕

Get the latest in DevSecOps and
Open Source Governance
delivered to your inbox each
week!

sonatype

Subscribe now