# 使用Convoy-NFS构建共享卷 - RancherLabs的博客 - 博客频道 - CSDN.NET

http://blog.csdn.net/rancherlabs/article/details/52774702

**使用Convoy-NFS构建共享卷**

**Introduction**

如果你用过[**Docker**](http://lib.csdn.net/base/docker)你就会知道，共享卷和跨主机的数据访问是个非常棘手的问题。虽然Docker的生态系统在逐渐走向成熟，但对大多数人来说，在不同环境中实现持久化存储还是很麻烦的。幸运的是，Rancher一直在研究这件事，并且想出了一个能独特的、能解决大部分这个问题的方案。用共享存储运行[**数据库**](http://lib.csdn.net/base/mysql)的方法仍没有被广泛推荐，但对于许多其他的情况，跨主机共享卷倒是一个很好的做法。

这个指南中的大部分内容是受Rancher的一个线上Meetup的启发。另外，如果你想自己从头开始搭建Convoy-NFS，这个[网页](https://github.com/rancher/rancher/wiki/Setup:-convoy-nfs)上有一些有关NFS配置的信息，你也许能作为参考。

**Rancher Convoy**

如果你以前没有听说过Rancher的[Convoy项目](https://github.com/rancher/convoy)，这里可以先简单介绍一下。Rancher希望可以通过Convoy项目让持久化容量存储变得简单方便。Convoy是一个非常棒的磁盘插件，因为它为用户提供了多种不同的选择。例如EBS卷和S3的支持，与VFS/NFS一起，为用户提供了一些用于配置共享存储的厉害并且灵活的选择。

**Dockerized-NFS**

这里有一个小秘诀，教你怎样启动一个能和Convoy-NFS服务连接起来的Docker化的NFS服务器。Docker-NFS基本上是一个穷人的EFS。如果你想运行它，你必须有足够的信心，相信服务器不会被毁，或相信你的数据无足轻重，即使丢失了也无所谓。你可以在这里找到更多我过去使用的Docker NFS服务器的信息。

给点进阶些的提议，我会建议你看看这个叫做弹性文件存储或简称EFS的东西，它是 NFS的AWS实现。这个解决方案更简单粗暴，它是一个随时可用于生产的NFS服务器，你可以把它视为Convoy-NFS的后端。构建EFS是很简单的，不过它超出了这篇文章的讨论范围。你可以在这个链接中看看[如何构建和配置EFS](https://aws.amazon.com/blogs/aws/amazon-elastic-file-system-production-ready-in-three-regions/)。

警告：因为EFS是AWS的一个较新的服务，它只能在几个位置上使用。（不过更多的可使用位置也很快会出现）。

下图就是对于Docker-NFS服务器来说，你的docker-compose.yml 可能的样子：

docker-nfs:

image: cpuguy83/nfs-server

privileged: true

volumes:

- /exports

command:

- /exports

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 1
* 2
* 3
* 4
* 5
* 6
* 7

在采用这个容器化的NFS服务器的方法时你可能会碰到的一个疑难杂症是，你的主机要么没有安装NFS内核模块，要么没有打开伴随的服务。

在Ubuntu上很容易安装内核模块，SSH到主机上， 运行NFS Server容器以及以下命令：

sudo apt-get install nfs-kernel-server

* 1
* 1

在CoreOS上，模块是安装了的，只是没有打开。要启用NFS，你需要SSH到对应主机上，运行NFS Server容器，并运行下面的命令：

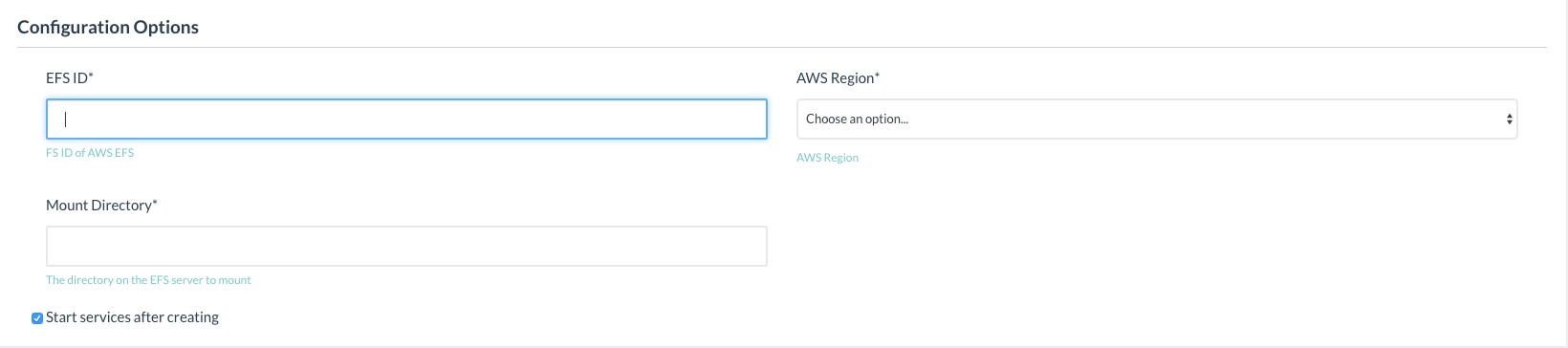
sudo systemctl start rpc-mountd

* 1
* 1

**配置EFS**

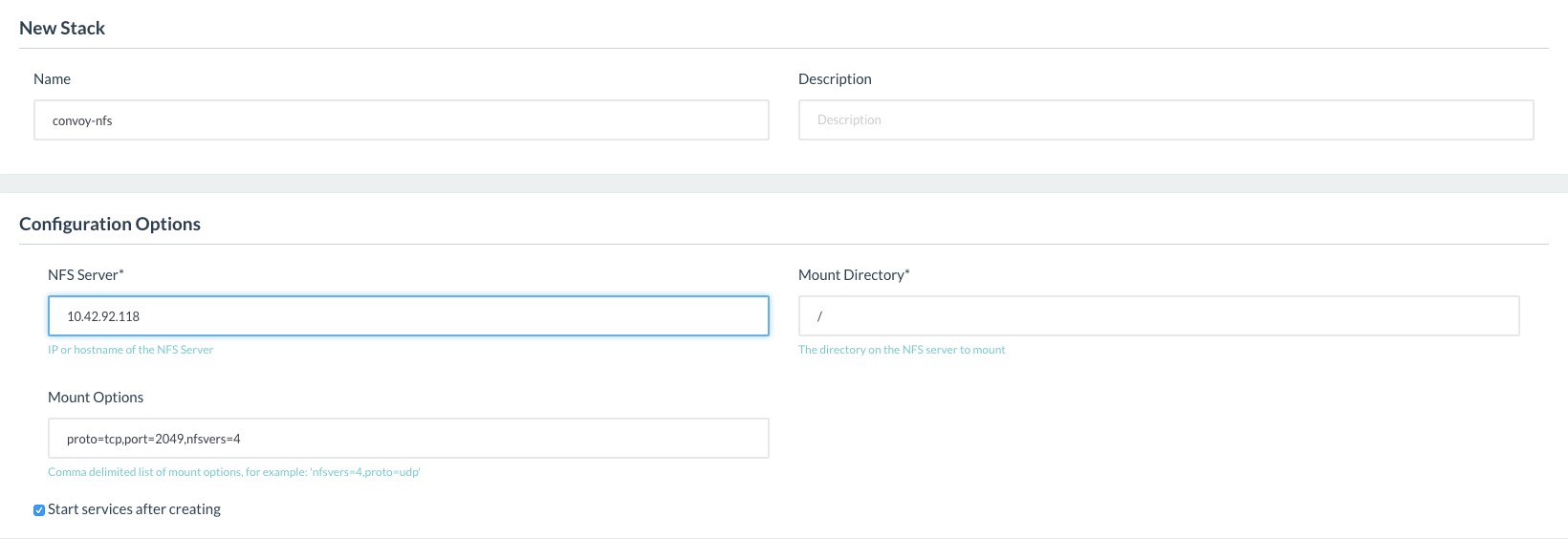
配置EFS是很容易的。用上面提到过的[链接](https://aws.amazon.com/blogs/aws/amazon-elastic-file-system-production-ready-in-three-regions/)，你可以了解到创建一个能与Convoy连接的EFS卷的全部步骤。创建EFS卷时，你可以注意一下EFS share的IP地址，或者如果你在创建卷之后展开到配置中，那就注意一下AWS为share提供的DNS名。

在Rancher的社区应用服务目录里有一个新的目录入口，有了它，你可以直接通过Convoy使用EFS卷，这能简化某些配置。使用应用服务目录入口仍需要你先创建EFS share，但配置Convoy来与其连接的过程会简单很多。只需要从AWS里复制出EFS ID，选择其中你已创建共享的区域，然后指定要将EFS share安装到本地的位置。“/efs”就是一个很好的在最开始[**测试**](http://lib.csdn.net/base/softwaretest)东西的例子：



**Convoy-NFS**

目前，Rancher还提供另外一种名叫“Convoy-NFS”的应用服务目录项，来将容器连接到NFS Server上：



安装程序是非常简单的，但也有几件事情要注意。首先，堆栈必须命名为“Convoy-NFS”，这是插件的名称。其次，NFS Server应该和在其上设置NFS Server的主机名相匹配。如果你创建的是Docker-NFS容器，记得使用容器的IP（为了写这篇文章，我用了一个测试环境，因而只用了Rancher容器内部IP来部署NFS Server）。当你用EFS创建你的NFS share时，使用已配置的DNS的名称。

最后一点要注意的是安装选项和安装点。这里的端口需要与将NFS Server 和(2049 for docker-nfs)配置在一起的端口相匹配，并且确保你打开了nfsver=4。另外，如果你要使用nfsvers=4选项，记得一定为MountDirectory用“/”。不使用nfsvers=4选项的话就用“/exports”。

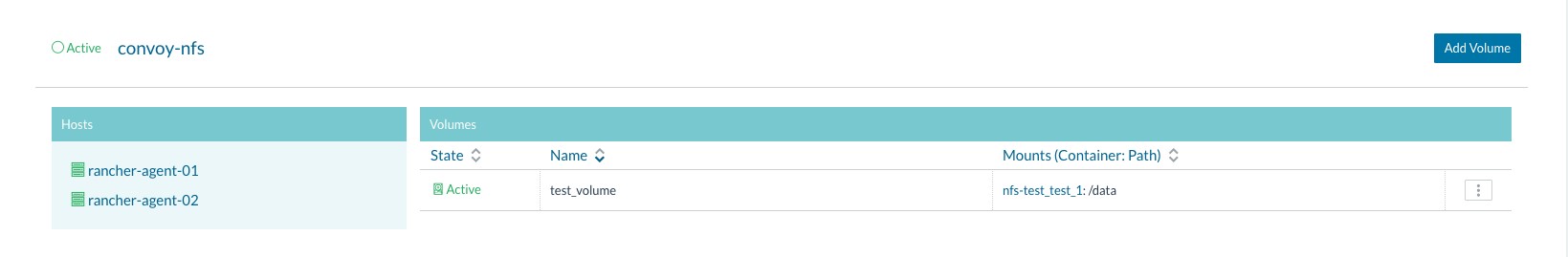
最终一步的配置和下图看上去差不多：

proto=tcp,port=2049,nfsvers=4

* 1
* 1

你可以添加其他选项来调整共享，但这些都是最低限度的构建所必需的组件。

给Rancher几分钟时间提供Convoy-NFS容器。一切完成后你就可以创建和攻击NFS卷了。最快的检查是否一切正常的方法，是点击Infrastructure -> Storage Pools选项。如果你在视图中看见了主机，那你就可以开始准备创建和分享卷了。



现在你可以开始从Storage Pools视图中手动创建一个卷，或干脆创建一个使用Convoy-NFS driver和卷名的服务。

我将创建一个测试容器，它可跨越两个主机共享相同的“test\_volume”以及数据，如下图所示（我在本地使用了rancher-compose来堆栈，如果你觉得使用GUI更容易的话敬请使用GUI）。

test:

image: ubuntu

volume\_driver: convoy-nfs

tty: true

volumes:

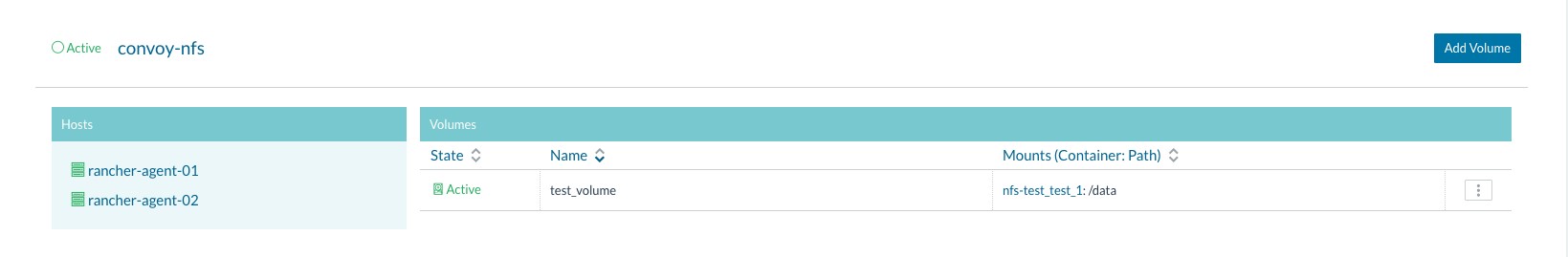
- test\_volume:/data

command:

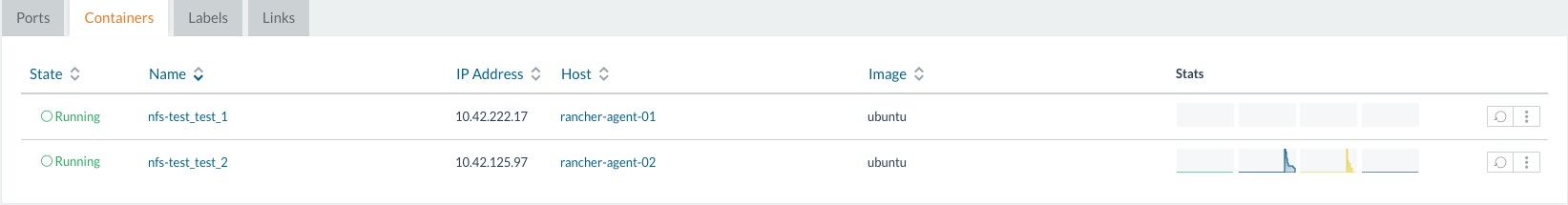
- bash

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8

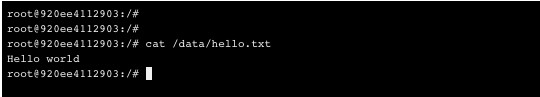
一个新的卷会在Storage Pools页面上弹出来：



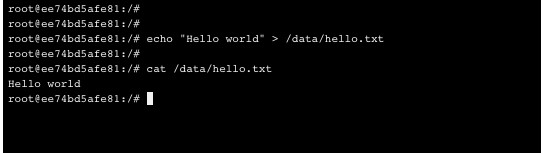
如果你想验证是否一切运行正常，你可以将容器的数量扩展到两个或更多个。然后exec到一个容器中，在其中创建一个文件，看你能否从另一个容器中读取这个文件，而且最好是在一个不同的主机上。如果这些都没问题，你应该已经成功了：



在第一个容器中，我们可以编写出文件：



在第二个容器中，我们可以重新读取它，检测一下共享存储是否运行正常：



**结论**

随着Convoy不断的发展和壮大，我们会看到更多的部署的选择。现在，NFS是稳定的，能正常工作的。如果你已经在使用AWS了，EFS会让一切都变得更加容易。我非常期待看到Convoy和Rancher中的共享卷会有什么下一步的发展。



点[这里](http://www.bagevent.com/event/251408)报名参加11月6日成都「Docker & CI/CD技术沙龙」。

# Rancher | Convoy-NFS 使用Convoy-NFS构建共享卷｜cnrancher.com

http://www.cnrancher.com/setting-shared-volumes-convoy-nfs/

[Shirley Huang](http://www.cnrancher.com/author/huangxu/) on 10月 13, 2016

如果你用过Docker你就会知道，共享卷和跨主机的数据访问是个非常棘手的问题。虽然Docker的生态系统在逐渐走向成熟，但对大多数人来说，在不同环境中实现持久化存储还是很麻烦的。不过，Rancher一直在研究这件事，并且想出了一个能独特的、能解决大部分这个问题的方案。用共享存储运行数据库的方法仍没有被广泛推荐，但对于许多其他的情况，跨主机共享卷倒是一个很好的做法。

这个指南中的大部分内容是受Rancher的一个线上Meetup的启发。另外，如果你想自己从头开始搭建Convoy-NFS，这个网页上有一些有关NFS配置的信息，你也许能作为参考。

**Rancher Convoy**

如果你以前没有听说过Rancher的Convoy项目，这里可以先简单介绍一下。Rancher希望可以通过Convoy项目让持久化容量存储变得简单方便。Convoy是一个非常棒的磁盘插件，因为它为用户提供了多种不同的选择。例如EBS卷和S3的支持，与VFS/NFS一起，为用户提供了一些用于配置共享存储的厉害并且灵活的选择。

**Dockerized-NFS**

这里有一个小秘诀，教你怎样启动一个能和Convoy-NFS服务连接起来的Docker化的NFS服务器。Docker-NFS基本上是一个穷人的EFS。如果你想运行它，你必须有足够的信心，相信服务器不会被毁，或相信你的数据无足轻重，即使丢失了也无所谓。你可以在这里找到更多我过去使用的Docker NFS服务器的信息。

给点进阶些的提议，我会建议你看看这个叫做弹性文件存储或简称EFS的东西，它是 NFS的AWS实现。这个解决方案更简单粗暴，它是一个随时可用于生产的NFS服务器，你可以把它视为Convoy-NFS的后端。构建EFS是很简单的，不过它超出了这篇文章的讨论范围。你可以在这个链接中看看如何构建和配置EFS。

警告：因为EFS是AWS的一个较新的服务，它只能在几个位置上使用。（不过更多的可使用位置也很快会出现）。

下图就是对于Docker-NFS服务器来说，你的docker-compose.yml 可能的样子：

docker-nfs:

image: cpuguy83/nfs-server

privileged: true

volumes:

- /exports

command:

- /exports

在采用这个容器化的NFS服务器的方法时你可能会碰到的一个疑难杂症是，你的主机要么没有安装NFS内核模块，要么没有打开伴随的服务。

在Ubuntu上很容易安装内核模块，SSH到主机上， 运行NFS Server容器以及以下命令：

sudo apt-get install nfs-kernel-server

在CoreOS上，模块是安装了的，只是没有打开。要启用NFS，你需要SSH到对应主机上，运行NFS Server容器，并运行下面的命令：

sudo systemctl start rpc-mountd

**配置EFS**

配置EFS是很容易的。用上面提到过的链接，你可以了解到创建一个能与Convoy连接的EFS卷的全部步骤。创建EFS卷时，你可以注意一下EFS share的IP地址，或者如果你在创建卷之后展开到配置中，那就注意一下AWS为share提供的DNS名。

在Rancher的社区应用服务目录里有一个新的目录入口，有了它，你可以直接通过Convoy使用EFS卷，这能简化某些配置。使用应用服务目录入口仍需要你先创建EFS share，但配置Convoy来与其连接的过程会简单很多。只需要从AWS里复制出EFS ID，选择其中你已创建共享的区域，然后指定要将EFS share安装到本地的位置。“/efs”就是一个很好的在最开始测试东西的例子：

[](http://www.cnrancher.com/wp-content/uploads/2016/10/%E5%9B%BE4.jpg)

**Convoy-NFS**

目前，Rancher还提供另外一种名叫“Convoy-NFS”的应用服务目录项，来将容器连接到NFS Server上：

[](http://www.cnrancher.com/wp-content/uploads/2016/10/%E5%9B%BE5.jpg)

安装程序是非常简单的，但也有几件事情要注意。首先，堆栈必须命名为“Convoy-NFS”，这是插件的名称。其次，NFS Server应该和在其上设置NFS Server的主机名相匹配。如果你创建的是Docker-NFS容器，记得使用容器的IP（为了写这篇文章，我用了一个测试环境，因而只用了Rancher容器内部IP来部署NFS Server）。当你用EFS创建你的NFS share时，使用已配置的DNS的名称。

最后一点要注意的是安装选项和安装点。这里的端口需要与将NFS Server 和(2049 for docker-nfs)配置在一起的端口相匹配，并且确保你打开了nfsver=4。另外，如果你要使用nfsvers=4选项，记得一定为MountDirectory用“/”。不使用nfsvers=4选项的话就用“/exports”。

最终一步的配置和下图看上去差不多：

proto=tcp,port=2049,nfsvers=4

你可以添加其他选项来调整共享，但这些都是最低限度的构建所必需的组件。

给Rancher几分钟时间提供Convoy-NFS容器。一切完成后你就可以创建和攻击NFS卷了。最快的检查是否一切正常的方法，是点击Infrastructure -> Storage Pools选项。如果你在视图中看见了主机，那你就可以开始准备创建和分享卷了。

[Convoy-NFS构建共享卷-图3](http://www.cnrancher.com/wp-content/uploads/2016/10/%E5%9B%BE7.jpg)

现在你可以开始从Storage Pools视图中手动创建一个卷，或干脆创建一个使用Convoy-NFS driver和卷名的服务。

我将创建一个测试容器，它可跨越两个主机共享相同的“test\_volume”以及数据，如下图所示（我在本地使用了rancher-compose来堆栈，如果你觉得使用GUI更容易的话敬请使用GUI）。

test:

image: ubuntu

volume\_driver: convoy-nfs

tty: true

volumes:

- test\_volume:/data

command:

- bash

一个新的卷会在Storage Pools页面上弹出来：

[Convoy-NFS构建共享卷-图3](http://www.cnrancher.com/wp-content/uploads/2016/10/%E5%9B%BE7.jpg)

如果你想验证是否一切运行正常，你可以将容器的数量扩展到两个或更多个。然后exec到一个容器中，在其中创建一个文件，看你能否从另一个容器中读取这个文件，而且最好是在一个不同的主机上。如果这些都没问题，你应该已经成功了：

[Convoy-NFS构建共享卷-图4](http://www.cnrancher.com/wp-content/uploads/2016/10/%E5%9B%BE10.jpg)

在第一个容器中，我们可以编写出文件：

[Convoy-NFS构建共享卷-图5](http://www.cnrancher.com/wp-content/uploads/2016/10/%E5%9B%BE11.jpg)

在第二个容器中，我们可以重新读取它，检测一下共享存储是否运行正常：

[](http://www.cnrancher.com/wp-content/uploads/2016/10/%E5%9B%BE12.jpg)

**结论**

随着Convoy不断的发展和壮大，我们会看到更多的部署的选择。现在，NFS是稳定的，能正常工作的。如果你已经在使用AWS了，EFS会让一切都变得更加容易。我非常期待看到Convoy和Rancher中的共享卷会有什么下一步的发展。