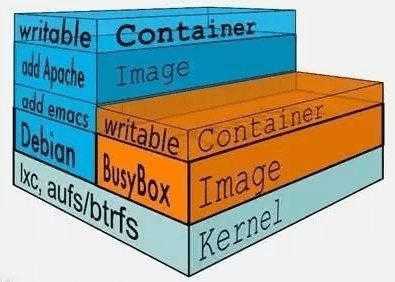
Docker联合文件系统Union File System，它是实现**[Docker镜像](http://www.dockerinfo.net/dockerimages" \t "_blank)**的技术基础，是一种轻量级的高性能分层文件系统，支持将文件系统中的修改进行提交和层层叠加，这个特性使得镜像可以通过分层实现和继承。同时支持将不同目录挂载到同一个虚拟文件系统下。

在Docker镜像分为基础镜像和父镜像，没有父镜像的镜像被称为基础镜像。用户是基于基础镜像来制作各种不同的应用镜像。这些应用镜像共享同一个基础镜像层，提高了存储效率。

当用户通过升级程序到新版本，改变了一个Docker镜像时，一个新的镜像层会被创建。因此，用户不用替换整个原镜像或者完全重新建立新镜像，只需要添加新层即可。在用户分发镜像的时，也只需要分发被改动的新层内容(增量部分)。这让Docker的镜像管理变得十分轻松级和快速。

在[**Docker**](http://www.dockerinfo.net/)中使用AUFS(Another Union File System或Advanced Multilayered Unification File System)就是一种联合文件系统。AUFS不仅可以对每一个目录设定只读(Readonly)、读写(Readwrite)和写(Witeout-able)权限，同时AUFS也可以支持分层的机制，例如，可以对只读权限部分逻辑上进行增量地修改而不影响只读部分。



当Docker在利用镜像启动一个容器时，Docker镜像将分配文件系统，并且挂载一个新的可读写的层给容器，容器将会在这个文件系统中被创建，并且这个可读写的层被添加到镜像中。Docker目前支持的联合文件系统种类包括AUFS、Btrfs、VFS和DeviceMapper等。

**AUFS (AnotherUnionFS)**

Docker的Container机制和使用是建立在LXC基础之上的，然而LXC本身存在很多问题，例如难以移动、标准化、模板化、重建、复制等。但这些操作又是Container实现快速大规模部署和更新所必备的。

Docker 正是利用AUFS分层技术来实现对Container的快速更新和大规模部署，并且在Docker中引入了Storage Driver技术，实现对外置存储的良好支持。Docker目前 支持AUFS、 VFS、 DeviceMapper、 对BTRFS以及ZFS引入和支持提供了技术规划。

AUFS是一种 Union FS, 简单来说就是“支持将不同目录挂载到同一个虚拟文件系统下的文件系统”, AUFS支持为每一个成员目录设定只读(Rreadonly)、读写(Readwrite)和写(Whiteout-able)权限。Union FS 可以将一个Readonly的Branch和一个Writeable的Branch联合在一起挂载在同一个文件系统下，Live

CD正是基于此可以允许在 OS image 不变的基础上允许用户在其上进行一些写操作。Docker在AUFS上构建的Container Image也正是如此。

接下来我们从linux启动为例介绍docker在AUFS特性的运用。前面我们介绍容器演进和技术基础介绍，典型的Linux启动到运行需要两个FileSystem，BootFS 和RootFS。

BootFS 主要包含BootLoader 和Kernel, BootLoader主要是引导加载Kernel, 当Boot成功后，Kernel被加载到内存中BootFS就被Umount了。

RootFS包含的就是典型 Linux 系统中的 /dev、/proc、/bin 等标准目录和文件。

不同的linux发行版，BootFS基本是一致的, RootFS会有差别，因此不同的发行版可以共享BootFS。

Linux在启动后，首先将RootFS 置为 Readonly，进行一系列检查后将其切换为Readwrite供用户使用。在Docker中，也是利用该技术，然后利用Union Mount在Readonly的RootFS文件系统之上挂载Readwrite文件系统。并且向上叠加, 使得一组Readonly和一个Readwrite的结构构成一个Container的运行目录、每一个被称作一个文件系统Layer。

AUFS的特性, 使得每一个对Readonly层文件/目录的修改都只会存在于上层的Writeable层中。这样由于不存在竞争、而且多个Container可以共享Readonly文件系统层。在Docker中，将Readonly的层称作“image” 镜像。对于Container整体而言，整个RootFS变得是read-write的，但事实上所有的修改都写入最上层的writeable层中，image不保存用户状态，可以用于模板、重建和复制。

在Docker中，上层的Image依赖下层的Image，因此**[Docker](http://www.dockerinfo.net/" \t "_blank)**中把下层的Image称作父Image，没有父Image的Image称作Base Image。因此，想要从一个Image启动一个Container，Docker会先逐次加载其父Image直到Base Image，用户的进程运行在Writeable的文件系统层中。所有父Image中的数据信息以及ID、**[网络](http://www.dockerinfo.net/docker/docker%E7%BD%91%E7%BB%9C" \t "_blank)**和LXC管理的资源限制、具体container的配置等，构成一个Docker概念上的Container。

最后我们总结一下Docker优势，采用AUFS作为Docker的Container的文件系统，能够提供的优势只要有以下几点。

多个Container可以共享父Image存储，节省存储空间；快速部署 – 如果要部署多个Container，Base Image可以避免多次拷贝，实现快速部署。因为多个Container共享Image，提高多个Container中的进程命中缓存内容的几率。相比于Copy-on-write类型的FS，Base Image也是可以挂载为可Writeable的，可以通过更新Base Image而一次性更新其之上的Container。