Docker面试问题与解答（典藏版）

忠实的码农
还是那个来自火星的Java高级架构师

如今，公司不再一次性发布大量功能，而是试图通过一系列发布列车将小功能传输给客户。这具有许多优点，例如来自客户的快速反馈，更好的软件质量等，这反过来导致高的客户满意度。为实现这一目标，公司必须：

- 增加部署频率
- 降低新版本的故障率
- 修复之间缩短了提前期
- 新版本崩溃时平均恢复时间更快

DevOps满足所有这些要求，有助于实现无缝的软件交付。

DevOps有哪些优势?

技术优势：

- 持续的软件交付
- 修复不太复杂的问题
- 更快地解决问题

商业利益：

- 更快速地传递功能
- 更稳定的操作环境
- 有更多时间可以增加价值（而不是修复/维护）

CI（持续集成）服务器的功能是什么?

CI服务器功能是不断地集成所有正在进行的更改并由不同的开发人员提交到存储库，并检查编译错误。它需要每天多次构建代码，最好是在每次提交之后，以便它可以检测在问题发生时是哪个提交Bug了。

什么是虚拟化?

虚拟化允许您在相同的硬件上运行两个完全不同的操作系统。每个客户操作系统都经历了引导，加载内核等所有过程。您可以拥有非常严格的安全性，例如，客户操作系统无法完全访问主机操作系统或其他客户端并搞砸了。

可以基于虚拟化方法如何模仿客户操作系统的硬件并模拟客户操作环境来对虚拟化方法进行分类。主要有三种类型的虚拟化：

- 仿真
- 半虚拟化
- 基于容器的虚拟化

Docker与虚拟机有何不同?

Docker不是虚拟化方法。它依赖于实际实现基于容器的虚拟化或操作系统级虚拟化的其他工具。为此，Docker最初使用LXC驱动程序，然后移动到libcontainer现在重命名为runc。Docker主要专注于在应用程序容器内自动部署应用程序。应用程序容器旨在打包和运行单个服务，而系统容器则设计为运行多个进程，如虚拟机。因此，Docker被视为容器化系统上的容器管理或应用程序部署工具。

- 与虚拟机不同，容器不需要引导操作系统内核，因此可以在不到一秒的时间内创建容器。此功能使基于容器的虚拟化比其他虚拟化方法更加独特和可取。
- 由于基于容器的虚拟化为主机增加了很少或没有开销，因此基于容器的虚拟化具有接近本机的性能
- 对于基于容器的虚拟化，与其他虚拟化不同，不需要其他软件。
- 主机上的所有容器共享主机的调度程序，从而节省了额外资源的需求。
- 与虚拟机映像相比，容器状态（Docker或LXC映像）的大小很小，因此容器映像很容易分发。
- 容器中的资源管理是通过cgroup实现的。Cgroups不允许容器消耗比分配给它们更多的资源。虽然主机的所有资源都在虚拟机中可见，但无法使用。这可以通过在容器和主机上同时运行top或htop来实现。所有环境的输出看起来都很相似。

容器内部机制?

大约在2006年，包括Google的一些员工在内的人们实现了名为命名空间的新的Linux内核级功能（不过这个想法早在FreeBSD中就已存在）。操作系统的一个功能是允许将全局资源（如网络和磁盘）共享到进程。如果将这些全局资源包装在命名空间中，以使它们仅对在同一命名空间中运行的那些进程可见，该怎么办？比如说，你可以获得一大块磁盘并将其放在命名空间X中，然后在命名空间Y中运行的进程无法查看或访问它。类似地，名称空间X中的进程无法访问分配给名称空间Y的内存中的任何内容。当然，X中的进程无法查看或与名称空间Y中的进程通信。这为全局资源提供了一种虚拟化和隔离。

这就是Docker的工作原理：每个容器都在自己的命名空间中运行，但使用与所有其他容器完全相同的内核。发生隔离是因为内核知道分配给进程的命名空间，并且在API调用期间确保进程只能访问其自己的命名空间中的资源。

什么是Docker?

- Docker是一个容器化平台，它以容器的形式将您的应用程序及其所有依赖项打包在一起，以确保您的应用程序在开发，测试或生产的任何环境中无缝运行。
- Docker容器，将一个软件包装在一个完整的文件系统中，该文件系统包含运行所需的一切：代码，运行时，系统工具，系统库等可以安装在服务器上的任何东西。
- 这可以保证软件始终运行相同，无论其环境如何。

如何使用Docker构建与环境无关的系统?

有三个主要功能有助于实现这一目标：

- Volumes
- 环境变量注入

- 只读文件系统

什么是Docker镜像?

Docker镜像是Docker容器的源代码。换句话说，Docker镜像用于创建容器。使用build命令创建映像，并且在使用run启动时它们将生成容器。镜像存储在Docker注册表registry.hub.docker.com中，因为它们可能变得非常大，镜像被设计为由其他镜像层组成，允许在通过网络传输镜像时发送最少量的数据。

什么是Docker容器?

Docker容器包括应用程序及其所有依赖项，但与其他容器共享内核，作为主机操作系统上用户空间中的独立进程运行。Docker容器不依赖于任何特定的基础架构：它们可以在任何计算机，任何基础架构和任何云中运行。

什么是Docker Hub?

Docker hub是一个基于云的注册表服务，允许您链接到代码存储库，构建镜像并测试它们，存储手动推送的镜像以及指向Docker云的链接，以便您可以将镜像部署到主机。它为整个开发流程中的容器镜像发现，分发和变更管理，用户和团队协作以及工作流自动化提供了集中资源。

Docker容器有几种状态?

Docker容器可以有四种状态：

- 运行
- 已暂停
- 重新启动
- 已退出

我们可以通过运行命令来识别Docker容器的状态：

```
docker ps –a
```

这将依次列出所有可用的docker容器及其在主机上的相应状态。从那里我们可以很容易地识别感兴趣的容器，以相应地检查其状态。

什么类型的应用程序 - 无状态或有状态更适合Docker容器?

最好为Docker Container创建无状态应用程序。我们可以从应用程序中创建一个容器，并从应用程序中取出可配置的状态参数。现在我们可以在生产和具有不同参数的QA环境中运行相同的容器。这有助于在不同场景中重用相同的图像。使用Docker Containers比使用有状态应用程序更容易扩展无状态应用程序。

解释基本的Docker使用流程

1. 一切都从Dockerfile开始。Dockerfile是镜像的源代码。
2. 创建Dockerfile后，您可以构建它以创建容器的镜像。镜像只是“源代码”的“编译版本”，即Dockerfile。
3. 获得容器的镜像后，应使用注册表重新分发容器。注册表就像一个git存储库 - 你可以推送和拉取镜像。
4. 接下来，您可以使用该镜像来运行容器。在许多方面，正在运行的容器与虚拟机（但没有管理程序）非常相似。

Dockerfile中最常见的指令是什么?

Dockerfile中的一些常用指令如下：

- FROM：我们使用FROM为后续指令设置基本镜像。在每个有效的Dockerfile中，FROM是第一条指令。
- LABEL：我们使用LABEL按照项目，模块，许可等组织我们的镜像。我们也可以使用LABEL来帮助实现自动化。在LABEL中，我们指定一个键值对，以后可用于以编程方式处理Dockerfile。
- RUN：我们使用RUN命令在当前镜像之上的新图层中执行任何指令。使用每个RUN命令，我们在镜像顶部添加一些内容，并在Dockerfile的后续步骤中使用它。
- CMD：我们使用CMD命令提供执行容器的默认值。在Dockerfile中，如果我们包含多个CMD命令，则只使用最后一条指令。

Dockerfile中的命令COPY和ADD命令有什么区别?

一般而言，虽然ADD并且COPY在功能上类似，但是COPY是优选的。

那是因为它比ADD更透明。COPY仅支持将本地文件基本复制到容器中，而ADD具有一些功能（如仅限本地的tar提取和远程URL支持），这些功能并不是很明显。因此，ADD的最佳用途是将本地tar文件自动提取到镜像中，如ADD rootfs.tar.xz /中所示。

解释一下dockerfile的ONBUILD指令?

当镜像用作另一个镜像构建的基础时，ONBUILD指令向镜像添加将在稍后执行的触发指令。如果要构建将用作构建其他镜像的基础的镜像（例如，可以使用特定于用户的配置自定义的应用程序构建环境或守护程序），这将非常有用。

Docker镜像和层有什么区别?

- 镜像：Docker镜像是由一系列只读层构建的
- 层：每个层代表镜像Dockerfile中的一条指令。

下面的Dockerfile包含四个命令，每个命令都创建一个层。

```
FROM ubuntu:15.04
COPY . /app
RUN make /app
CMD python /app/app.py
```

重要的是，每个层只是与之前一层的一组差异层（相同的就不再放到新层中）。

什么是Docker Swarm?

Docker Swarm是Docker的本机群集。它将Docker主机池转变为单个虚拟Docker主机。Docker Swarm提供标准的Docker API，任何已经与Docker守护进程通信的工具都可以使用Swarm透明地扩展到多个主机。

如何在生产中监控Docker?

Docker提供docker stats和docker事件等工具来监控生产中的Docker。我们可以使用这些命令获取重要统计数据的报告。

- Docker统计数据：当我们使用容器ID调用docker stats时，我们获得容器的CPU，内存使用情况等。它类似于Linux中的top命令。
- Docker事件：Docker事件是一个命令，用于查看Docker守护程序中正在进行的活动流。

一些常见的Docker事件是：attach，commit，die，detach，rename，destroy等。我们还可以使用各种选项来限制或过滤我们感兴趣的事件。

Docker如何在非Linux系统中运行容器?

通过添加到Linux内核版本2.6.24的名称空间功能，可以实现容器的概念。容器将其ID添加到每个进程，并向每个系统调用添加新的访问控制检查。它由clone（）系统调用访问，该调用允许创建先前全局命名空间的单独实例。

如果由于Linux内核中可用的功能而可以使用容器，那么显而易见的问题是非Linux系统如何运行容器。Docker for Mac和Windows都使用Linux VM来运行容器。Docker Toolbox用于在Virtual Box VM中运行容器。但是，最新的Docker在Windows中使用Hyper-V，在Mac中使用Hypervisor.framework。

如何在多个环境中使用Docker?

可以进行以下更改：

- 删除应用程序代码的任何卷绑定，以便代码保留在容器内，不能从外部更改
- 绑定到主机上的不同端口
- 以不同方式设置环境变量（例如，减少日志记录的详细程度，或启用电子邮件发送）
- 指定重启策略（例如，重启：始终）以避免停机
- 添加额外服务（例如，日志聚合器）

因此，您可能希望定义一个额外的Compose文件，例如production.yml，它指定适合生产的配置。此配置文件只需要包含您要从原始Compose文件中进行的更改。

为什么Docker Compose不会等待容器准备就绪，然后继续以依赖顺序启动下一个服务?

Compose按照依赖顺服启动和停止容器，决定依赖关系语句有 depends_on, links, volumes_from, 和network_mode: "service:...".

但是，对于启动，Compose不会等到容器“准备好它运行“。这里有一个很好的理由：

- 等待数据库（例如）准备就绪的问题实际上只是分布式系统更大问题的一个子集。在生产中，您的数据库可能随时变得不可用或移动主机。您的应用程序需要能够适应这些类型的故障。
- 要处理此问题，请将应用程序设计为在发生故障后尝试重新建立与数据库的连接。如果应用程序重试连接，它最终可以连接到数据库。
- 最佳解决方案是在启动时以及出于任何原因丢失连接时，在应用程序代码中执行此检查。

更多关于Java的技术和资讯可以关注我的专栏：

Java架构筑基
zhuanlan.zhihu.com

专栏免费给大家分享Java架构的学习资料和视频

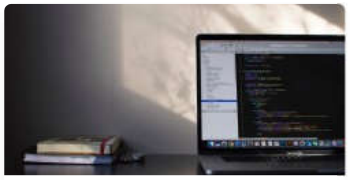
发布于 2019-04-16

文章被以下专栏收录

Java高级架构
欢迎大家加入Java进阶之路：908676731

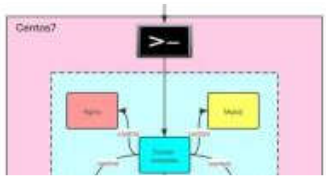
进入专栏

推荐阅读



用容器重新定义 Java 虚拟化部署

希云cSphere



Docker深入浅出 | Docker Compose多容器实战

Java架... 发表于Java程...

微服务 Docke

慕容千语

还没有评论

写下你的评论...