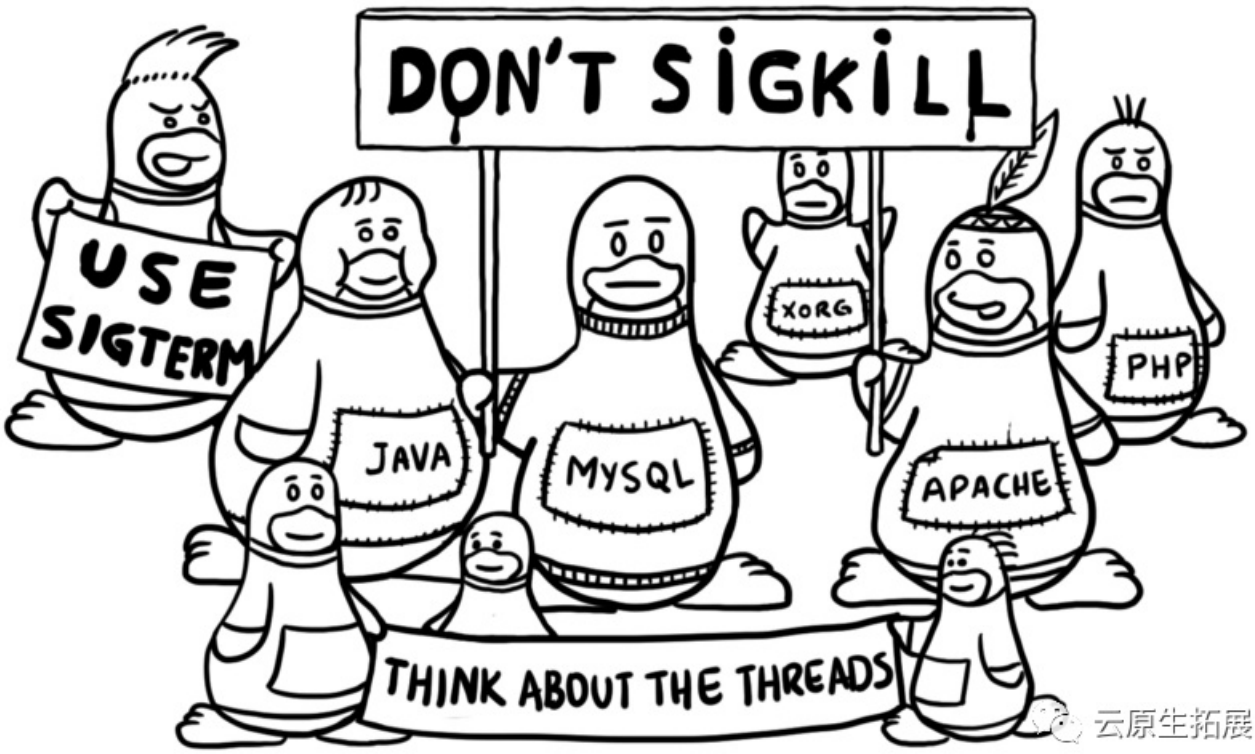
57Kubernetes 系列（五十四）云环境优雅停服务 (Kubernetes + Spring Boot 的例子)

Kubernetes 系列（五十四）云环境优雅停服务 (Kubernetes + Spring Boot 的例子)

许多人考虑在云中启动一个应用程序，但很少注意它是如何结束的。有一次，我们发现了一些与 Pod停止明确相关的错误。例如，我们看到Kubernetes偶尔会在释放资源之前杀死我们的应用程序，尽管这似乎不应该发生。立即重现这个问题是不可能的，我们想知道在这后面发生了什么?



在我们的研究中，我们发现服务需要几个优雅关机的点。通过本文的一些例子，我将展示为什么考虑这些点是至关重要的，以及如何实现这一任务。

我们使用 Kotlin/Spring Boot 开发。该项目在云环境中运行，Kubernetes 管理其生命周期。我们配置了应用程序的生命周期，Kubernetes 帮助我们处理其他细节。

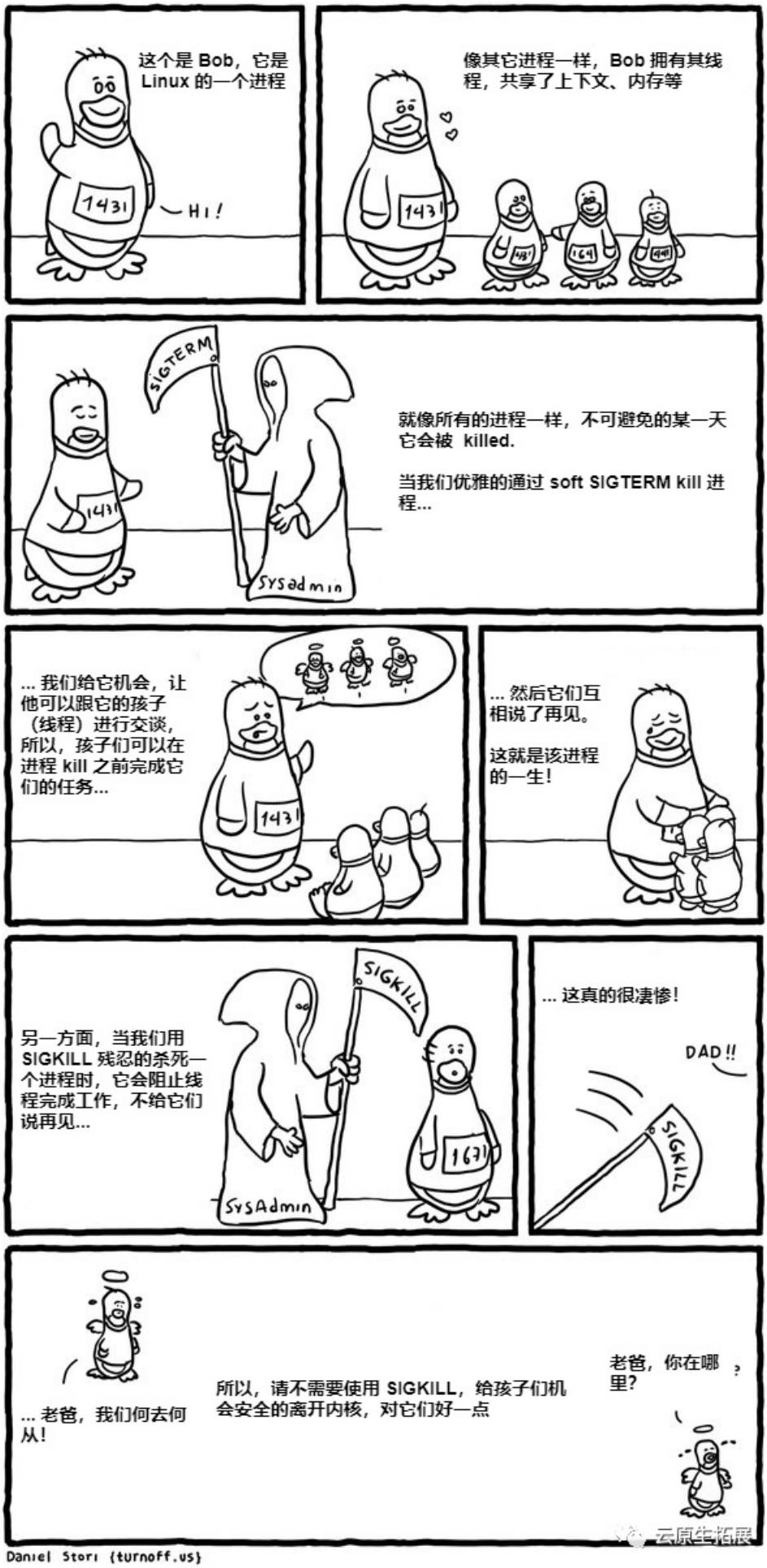
我们不理解它在应用程序生命周期内是如何工作的——它向 pods 发送什么信号，什么时候发送，以及它们如何处理这些信号。在本文中，我将通过两个简单的例子展示优雅关机的重要性。

操作系统应用程序生命周期

为了向应用程序传递信息，操作系统会向其发送带有特定代码的信号。这个想法出现在POSIX兼容的操作系统中，现在仍在使用。现在有大约30个不同的信号，但我只在这里提到与应用程序终止相关的信号:

- SIGINT 是一个信号，它应该在"正常模式"下终止正在运行的应用程序（不要匆忙）。在IDEA中单击Stop时，Java进程也会发生同样的情况。默认情况下，SIGINT被视为以交互模式终止进程。在接收到SIGINT时，进程请求用户确认或以其他方式使用信号（包括忽略）。
- SIGTERM 是默认的进程终止信号。这是 Linux 中 kill 命令从控制台发送的默认信号。但进程仍然有机会终止线程、释放资源，甚至忽略信号。
- SIGKILL-无需释放资源和终止线程即可明确地立即终止进程。

SIGTERM 和 SIGKILL 的不同



如果你经常需要点击IDEA中的停止（类似于SIGKILL）来终止一个进程或测试，会出现一个警告。当无法使用SIGTERM或SIGINT停止服务时，必须使用SIGKILL。在这种情况下，很可能会丢失请求或无法向文件中写入有价值的内容，从而导致严重的错误。

信号展示— Kubernetes 和 Spring Boot

在云环境中管理 Pod 时，Kubernetes 同样使用信号，根据其内部逻辑发送信号。它可以重新启动Pod，只是因为他决定将其移动到另一个节点，而无需我们的任何命令。它拉起一个新的Pod，然后杀死了旧的 Pod。看起来应用程序仍在继续工作，但它已经遇到了我们没有预料到的事情。

Kubernetes 不要求立即终止进程。向容器发送 SIGTERM需要等待一段时间（超时是可配置的，默认值为30秒），并且只有在进程没有终止且超时的时候才会发送 SIGKILL 。

这里可能会出什么问题?

SIGTERM 信号, 但是确以 SIGKILL 行动

问题:

当试图停止 Apache Tomcat 时，Kubernetes 会向 Apache 发送 SIGTERM，但Spring Boot会立即停止web服务器并中断线程，而不是预期的"常规"进程终止。所有传入请求的处理立马停止-服务器返回错误503。

解决方案:

在 Spring Boot 2.3.0 版本有一个配置:

```
server.shutdown=graceful
```

该设置允许您更合理地实现所有内容。收到SIGTERM后，服务器停止接受新请求，尝试在合理的时间内响应现有请求，更智能地完成繁重的请求，并在SIGKILL到达之前完成所有操作。

SIGKILL 以及 hang 住的 jobs

在项目的其他方面，我们对所有重复性作业使用SpringBatch框架（我们使用SpringBoot中的@Scheduled注释启动）。在引擎之上，它有一个数据库，它存储了有关执行的信息，何时，如何处理，以及结果是什么。

问题:

如果你在使用SIGKILL信号终止了SpringBatch应用程序，那么"挂起"作业将保留在启动历史记录中。它将永远保持"已启动"状态。

我们避免同时执行同一批处理的多个实例，因此"挂起"作业会阻止批处理的重新运行。通过从历史记录中删除挂起的作业，可以手动修复。

解决方案:

我们按照与上一个web服务器示例中相同的逻辑，为批处理实现了优雅的关闭:

- 收到 SIGTERM 后，我们禁止启动新任务;
- 尝试完成所有正在运行的任务;
- 我们等待一段时间（不比Kubernetes本身等待时间长）;
- 强制完成所有长任务（我们在数据库中相应地标记它们）。

收益 — 当 SIGKILL 来自 Kubernetes 时，所有资源都已释放。

这两个例子值得考虑，可以通过更仔细地处理流程终止来改进云应用程序的行为。

欢迎关注我的公众号“ **云原生拓展** ”，原创技术文章第一时间推送。