86Kubernetes 系列（七十九）揭开 DaemonSet 的神秘面纱：部署和管理集群范围的 Pods

介绍

随着容器化的普及程度不断提高，跨 Kubernetes 集群管理大规模部署带来了独特的挑战。其中一个挑战是确保某些系统级进程或应用程序在群集中的每个节点上运行。这就是 Kubernetes DaemonSet 发挥作用的地方。在这篇博文中，我们将深入探讨守护程序集，探讨它们的用途、用例以及部署和管理的最佳实践。

什么是 DaemonSet？

DaemonSet 是一个 Kubernetes 工作负载，可确保特定的 Pod 在集群中的每个节点上运行。它对于部署需要在每个节点上可用的群集级服务或后台进程特别有用。通过使用 DaemonSets，您可以毫不费力地管理这些工作负载，而无需担心手动分发或复制。

DaemonSet 用例

Daemon 具有广泛的用例，包括：

1. **日志收集和监控：** 部署日志收集器（如 Fluentd 或 Filebeat）作为 DaemonSet 可确保从集群中的每个节点收集日志，从而实现集中监控和分析。

2. **网络代理：** DaemonSet可用于部署网络代理，如 Calico 或 Weave，这些代理为群集中的所有节点提供网络和安全功能。

3. **资源监控：** 运行资源监控代理（如 Prometheus Node Exporter）作为 DaemonSet 允许您收集节点级指标并监控整个集群中的资源利用率。

4. **系统级服务：** DaemonSet 可用于部署系统级服务，例如群集范围的 DNS 解析器、存储配置程序或需要存在于每个节点上的任何其他进程。

部署 DaemonSet

部署 DaemonSet 很简单，就像任何其他 Kubernetes 资源一样。您可以通过指定容器模板（包括容器镜像、标签以及任何所需的卷或配置）来定义 DaemonSet 的所需状态。一旦你应用了DaemonSet清单，Kubernetes就会负责调度并确保每个节点上运行一个Pod。

管理 DaemonSet

管理 DaemonSet 涉及处理整个集群中底层 Pod 的生命周期。以下是需要考虑的一些基本方面：

1. **缩放：** DaemonSet 会在您在群集中添加或删除节点时自动缩放。当新节点加入集群时，会在该节点上自动调度一个 Pod。同样，当节点被删除时，关联的 Pod 将正常终止。

2. **滚动更新：** 更新 DaemonSet 涉及对集群中的每个 Pod 进行滚动更新。为此，您可以使用所需的更改更新容器模板规范并应用更新的 DaemonSet 清单。

3. **节点亲和：** 您可以利用节点亲和性或节点选择器来控制应在哪些节点上调度 DaemonSet Pod。这使您能够根据节点标签或其他条件在特定节点上部署特定工作负载。

4. **污点和容许：** 通过使用污点和容许，您可以自定义 DaemonSet 的调度行为。可以将污点应用于节点，并且可以将容错添加到 DaemonSet Pod，从而允许您对某些工作负载强制实施约束或优惠处理。

最佳实践

使用 DaemonSet 时，请考虑以下最佳实践：

1. **Pod 安全性：** 确保您的 DaemonSet Pod 在适当的安全上下文和资源限制下运行。这有助于防止资源争用并增强整体群集安全性。

2. **监视和日志记录：** 实施可靠的监视和日志记录解决方案，以收集和分析来自群集中的 DaemonSet Pod 的数据。这样可以实现主动故障排除和性能优化。

3. **更新策略：** 仔细规划更新策略，以避免停机或关键服务中断。利用滚动更新或金丝雀部署来确保平稳过渡。

4. **资源分配：** 根据 DaemonSet Pod 正在运行的工作负载的特定要求调整其资源分配。请注意 CPU 和内存利用率，以避免单个节点过载。

案例

下面是一个使用 Fluentd 部署日志收集器的 Kubernetes DaemonSet 规范示例：

```yaml
apiVersion: apps/v1
kind: DaemonSet
metadata:
  name: fluentd-daemonset
  labels:
    app: fluentd
spec:
  selector:
    matchLabels:
      app: fluentd
  template:
    metadata:
      labels:
        app: fluentd
    spec:
      containers:
        - name: fluentd
          image: fluent/fluentd:v1.12-debian-1
          resources:
            limits:
              memory: 200Mi
              cpu: 100m
          volumeMounts:
            - name: varlog
              mountPath: /var/log
      volumes:
        - name: varlog
          hostPath:
            path: /var/log
```

在此示例中：

- `metadata` 部分定义 DaemonSet 的名称和标签。
- `selector` 指定用于标识由 DaemonSet 管理的 Pod 的标签选择器。
- `template` 部分包含 DaemonSet 容器的容器模板。
- `containers` 部分定义 Pod 的容器规范。在本例中，我们有一个名为“fluentd”的容器运行 fluent/fluentd:v1.12-debian-1 镜像。
- `resources` 部分指定容器（内存和 CPU）的资源限制。
- `volumeMounts` 部分定义容器的卷装载，在本例中，将主机的 /var/log 目录装载到容器中。
- `volumes` 部分定义 Pod 的卷配置，使用 hostPath 卷挂载主机的 /var/log 目录。

当您使用 `kubectl apply -f fluentd-daemonset.yaml` 命令应用此 YAML 清单时，Kubernetes 将在集群中的每个节点上创建 DaemonSet Pod，确保 Fluentd 日志收集器正在运行，并从每个节点上的 /var/log 收集日志。

请注意，这是一个简化的示例，您可能需要根据您的特定要求对其进行修改，例如将 Fluentd 配置为将日志转发到中央日志管理系统或应用其他安全措施。

总结

Kubernetes DaemonSets 是一种强大的机制，用于在 Kubernetes 基础架构中部署和管理集群范围的 Pod。它们提供了一种方便的方法，可确保关键的系统级进程或应用程序在每个节点上运行，从而实现具有凝聚力和可扩展性的群集环境。通过有效利用 DaemonSets，您可以简化 Kubernetes 集群中各种工作负载的部署和管理。

无论是日志收集、网络代理、资源监控还是系统级服务，DaemonSet 都能提供灵活的解决方案来满足这些需求。通过遵循最佳实践并考虑基础架构的独特特征，您可以充分利用 DaemonSet 的全部潜力，并在 Kubernetes 部署中释放增强的可观测性、可扩展性和可靠性。