26Kubernetes 系列（二十二）如何增强 Kubernetes ingress 安全？

Kubernetes 系列（二十二）如何增强 Kubernetes ingress 安全?

Ingress 的目的是通过利用在 Ingress 资源创建期间定义的流量路由规则来简化您创建对Kubernetes服务的访问的方式。这最终允许您从 Kubernetes 集群之外公开HTTP和HTTPS，因此您不再需要单独公开每个服务—随着应用程序的扩展，这可能会变得昂贵和乏味。

Ingress 在保护 Kubernetes 应用程序方面也扮演着至关重要的角色，因为您可以在 Ingress 资源本身提供 TLS。考虑到保护 Kubernetes 应用程序的其他形式(如网络策略)，这一点至关重要，网络策略是建立在以应用程序为中心的概念之上的，该概念确定并保护 pods 之间的通信方式。

限制 pod-to-pod 网络是确保 Kubernetes 应用程序安全的正确方向。但是网络策略 与防火墙不是同义词 ，因为这种方法只确定是否允许连接—不确定传输中的身份验证或加密，TLS允许并在Ingress资源中配置。

本文将研究如何通过向 ingress 添加 TLS，然后获取TLS/SSL证书来保护 ingress 资源。

适用于 Kubernetes Ingress 的 SSL/TLS

在 Kubernetes 应用程序中配置 Ingress 涉及配置 Ingress 控制器和定义 Ingress 资源。Ingress 控制器处理SSL，而TLS证书引用作为Kubernetes secret 对象添加到 ingress 资源中，该对象由 ingress 控制器访问并成为其配置的一部分。我们将使用 NGINX 控制器。

控制器中配置 SSL

让我们从配置控制器开始。这是相当简单的，因为我们使用的是NGINX。我们需要在 nginx.conf 中添加 ssl_certificate_by_lua_block。这是一个当 NGINX 要执行SSL握手时运行的指令

```
ssl_certificate_by_lua_block {
    certificate.call()
}
```

创建一个 TLS 安全对象

为此，您需要创建 Kubernetes secret 对象 ，它将包含与证书相关的信息。这个 secret 将在 Ingress 对象中引用，因此您不需要在资源定义本身中放入敏感信息。

secret 对象将包含 server.crt 和 server.key。我们从证书颁发机构获取的密钥。我们将在设置TLS资源后解释如何实际获取它。

您必须在 Kubernetes 应用程序所在的命名空间中创建这个 secret。因此，您需要一个如下所示的 secret YAML定义 :

```
apiVersion: v1
kind: Secret
metadata:
  name: armo-tls-example
  namespace: dev
type: kubernetes.io/tls
data:
  server.crt: |
       <crt contents here>
  server.key: |
       <private key contents here>
```

可以使用如下命令生成 secrets 文件:

```
kubectl create secret armo-tls-example \
    --namespace dev \
    --key server.key \
    --cert server.crt
```

您应该指定命令中 server.key 以及 server.crt 文件的路径。在上面的示例中，您可以看到我们将 secret 对象命名为 armo-tls-example。我们将在 Ingress 资源YAML定义中通过这个名称来引用这个对象。

向 Ingress 资源添加TLS 定义

现在您已经创建了 secret 对象，您需要在 Ingress 资源YAML定义中利用它。例如，如果Ingress资源YAML文件如下:

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: armo-ingress-example
spec:
  ingressClassName: nginx
  rules:
  - host: armoexample.com
    http:
      paths:
      - path: /kubernetes
        pathType: Prefix
        backend:
          service:
            name: armoservice1
            port:
              number: 80
      - path: /kubernetes/security/
        pathType: Prefix
        backend:
          service:
            name: armoservice2
            port:
              number: 80
```

对于 TLS 定义块，YAML定义应该如下所示:

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: armo-ingress-example
spec:
  ingressClassName: nginx
  tls:
  - hosts:
    - armoexample.com
    secretName: armo-tls-example
  rules:
  - host: armoexample.com
    http:
      paths:
      - path: /kubernetes
        pathType: Prefix
        backend:
          service:
            name: armoservice1
            port:
              number: 80
      - path: /kubernetes/security/
        pathType: Prefix
        backend:
          service:
            name: armoservice2
            port:
              number: 80
```

这将在YAML定义的spec字段下添加TLS块。这个 secret 由创建时选择的名称armo-tls-example 来引用。您还需要确保TLS块和 rules 块具有相同的 host 值，因为根据定义的规则将TLS应用于 Ingress 流量。

获得 SSL 证书

获取SSL证书需要设置颁发者和证书颁发机构(CA)。颁发者通过定义如何获取、更新和使用证书，简化了CA生成的证书的生命周期自动化。

我们将使用cert-manager(https://cert-manager.io/)，它是最流行的证书颁发者之一。对于CA，我们将使用Let's Encrypt(https://letsencrypt.org/)，这也是比较流行的CA工具之一。

您需要做的第一件事是在Kubernetes集群中安装证书管理器。如果使用Helm，这是相当简单的。如果需要，可以参考cert-manager的文档(https://cert-manager.io/docs/installation/) 。

在验证正确安装了证书管理器之后，继续为Let's Encrypt创建证书颁发者YAML定义。请记住，仅凭cert-manager无法颁发证书。在本例中，处理实际证书颁发的是CA(让我们加密)。

Let's Encrypt的证书颁发者YAML如下所示:

```yaml
apiVersion: cert-manager.io/v1
kind: ClusterIssuer
metadata:
  name: armo-letsencrypt-example
spec:
  acme:
    server: https://acme-v02.api.letsencrypt.org/directory
    email: example@ domain.com
    privateKeySecretRef:
      name: armo-letsencrypt-example
    solvers:
      - http01:
          ingress:
            class: nginx
```

从上面的YAML描述中，需要考虑的主要事项是:

• 构建资源时使用的API版本为 cert-manager 的API版本。检查cert-manager文档以确保您使用的是最新的稳定API版本。

• Let's Encrypt 访问的详细信息在 email 和 privateKeySecretRef 字段中被捕获。

• acme 字段表示颁发者类型。您可以在cert-manager文档中阅读更多关于 ACME 的信息。

现在已经设置了证书颁发者，您需要在Ingress资源YAML描述中引用它。这可以使用YAML文件中的 annotation 来完成。

考虑到我们在前几节中使用的Ingress YAML文件，新的YAML文件应该如下所示:

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: armo-ingress-example
  annotations:
    kubernetes.io/ingress.class: nginx
    cert-manager.io/cluster-issuer: armo-letsencrypt-example
spec:
  ingressClassName: nginx
  tls:
  - hosts:
    - armoexample.com
    secretName: armo-tls-example
  rules:
  - host: armoexample.com
    http:
      paths:
      - path: /kubernetes
        pathType: Prefix
        backend:
          service:
            name: armoservice1
            port:
              number: 80
      - path: /kubernetes/security/
        pathType: Prefix
        backend:
          service:
            name: armoservice2
            port:
              number: 80
```

这里已经添加了 cert-manager.io/cluster-issuer 确保 cert-manager 检测到在YAML中定义的armo-letsencrypt-示例集群颁发者的使用情况，并获取证书。这将用于前面添加的TLS主机名字段中定义的名称。

将这些 annotation 添加到 Ingress 资源定义应该可以完成提供SSL证书所需的步骤。

总结

Ingress 是一个强大的Kubernetes资源，能够有效地确保它允许您使用它的全部潜力。它还允许您加强Kubernetes应用程序的安全性。

本文介绍了在Ingress中设置 TLS 作为增强安全性的一种方法的基础知识。但是，您可以应用许多更复杂的安全措施和实践。这就是可以使用类似 Kubescape 等解决方案的地方。

Kubescape是Kubernetes的一个开源平台，提供了一个多云的K8s单一窗口，包括风险分析、安全遵从性、RBAC可视化工具和图像漏洞扫描。

为了更好地探索Kubescape如何帮助您提高Kubernetes环境和集群的安全性，请访问Kubescape GitHub页面(https://github.com/kubescape/kubescape)。