Kubernetes証明書の期限切れによりクラスタ全体の通信が停止する

内容

概要

このドキュメントでは、Kubernetesベースのシステムが365日以上インストールされている場合に、お客様が直面する可能性のある停止の問題について説明します。さらに、状況を修正し、Kubernetesベースのシステムを再び稼働させるために必要な手順を実行します。

問題

デフォルトでインストールされたKubernetesクラスタが1年間経過すると、クライアント証明書が期限切れになります。Cisco CloudCenter Suite(CCS)にアクセスすることはできません。 それでも表示されますが、ログインできません。kubectl CLIに移動すると、「Unable to connect to the server:x509:証明書が期限切れであるか、まだ有効ではありません。

このbashスクリプトを実行すると、証明書の有効期限を確認できます。

```
for crt in /etc/kubernetes/pki/*.crt; do
     printf '%s: %s\n' \
     "$(date --date="$(openssl x509 -enddate -noout -in "$crt"|cut -d= -f 2)" --iso-8601)" \
     "$crt"
done | sort
```

また、Action Orchestratorのオープンソースワークフローも見つけることができます。このワークフローは毎日監視され、問題が発生したときにアラートされます。

https://github.com/cisco-cx-workflows/cx-ao-shared-workflows/tree/master/CCSCheckKubernetesExpiration__definition_workflow_01E01VIRWZDE24mWIsHrqCGB9xUix0f9ZxG

解決方法

新しい証明書は、クラスタ全体でKubeadmを介して再発行する必要があります。その後、ワーカーノードをマスターに再度参加させる必要があります。

1. マスターノードにログインします。

2. IPアドレスを**ip address show**で取得します。

```
[root@cx-ccs-prod-master-d7f34f25-f524-4f90-9037-7286202ed13a3 kubernetes]# ip address show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
inet 127.0.0.1/8 scope host lo
valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 8920 qdisc pfifo_fast state UP group default
qlen 1000
link/ether fa:16:3e:19:63:a2 brd ff:ff:ff:ff:ff:ff
inet 192.168.1.20/24 brd 192.168.1.255 scope global dynamic eth0
valid_lft 37806sec preferred_lft 37806sec
3: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group
default
link/ether 02:42:d0:29:ce:5e brd ff:ff:ff:ff:ff:ff
inet 172.17.0.1/16 scope global docker0
valid_lft forever preferred_lft forever
13: tunl0@NONE: <NOARP,UP,LOWER_UP> mtu 1430 qdisc noqueue state UNKNOWN group default qlen
1000
link/ipip 0.0.0.0 brd 0.0.0.0
inet 172.16.176.128/32 brd 172.16.176.128 scope global tunl0
valid_lft forever preferred_lft forever
14: cali65453a0219d@if4: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1430 qdisc noqueue state UP
group default
link/ether ee:ee:ee:ee:ee:ee brd ff:ff:ff:ff:ff:ff link-netnsid 4
```

3. cd /etc/kubernetesを使用してKubernetesディレクトリに移動します。

4. vi kubeadmCERT.yamlを介してkubeadmCERT.yamlというファイルを作成します。

5. ファイルは次のようになります。

```
apiVersion: kubeadm.k8s.io/v1alpha1
kind: MasterConfiguration
api:
  advertiseAddress: <IP ADDRESS FROM STEP 2>
kubernetesVersion: v1.11.6
#NOTE: If the customer is running a load balancer VM then you must add these lines after...
#apiServerCertSANs:
#- <load balancer IP>
```

6. 古い証明書と鍵をバックアップします。これは必須ではありませんが、推奨されます。バックアップディレクトリを作成し、これらのファイルをコピーします。

```
#Files
#apiserver.crt
#apiserver.key
#apiserver-kubelet-client.crt
#apiserver-kubelet-client.key
#front-proxy-client.crt
#front-proxy-client.key

#ie
cd /etc/kubernetes/pki
mkdir backup
mv apiserver.key backup/apiserver.key.bak
```

7. ステップ6.をスキップした場合は、rm apiserver.crtのようなrmコマンドを使用して、上記のファイルを削除するだけで構いません。

8. kubeadmCERT.yamlファイルの場所に戻ってください。kubeadm —config

kubeadmCERT.yaml alpha phase certs apisserverを介して新しいapiserver証明書を生成します。

```
[root@cx-ccs-prod-master-d7f34f25-f524-4f90-9037-7286202ed13a3 kubernetes]# kubeadm --
config kubeadmCERT.yaml alpha phase certs apiserver
[certificates] Generated apiserver certificate and key.
[certificates] apiserver serving cert is signed for DNS names [cx-ccs-prod-master-d7f34f25-
f524-4f90-9037-7286202ed13a3 kubernetes kubernetes.default kubernetes.default.svc
kubernetes.default.svc.cluster.local] and IPs [10.96.0.1 192.168.1.20]
```

9. kubeadm —config kubeadmCERT.yaml alpha phase certs apiserver-kubelet-clientを介して新しいapiserver kubelet certを生成します。

10. kubeadm —config kubeadmCERT.yaml alpha phase certs front-proxy-clientを介して新しいfront-proxy-client certを生成します。

11. /etc/kubernetesフォルダで.confファイルをバックアップします。必須ではありませんが、推奨されます。kubelet.conf、controller-manager.conf、scheduler.conf、および場合によってはadmin.confが必要です。バックアップしない場合は削除できます。

12. kubeadm —config kubeadmCERT.yaml alpha phase kubeconfig allを使って新しい構成ファイルを生成します。

```
[root@cx-ccs-prod-master-d7f34f25-f524-4f90-9037-7286202ed13a3 kubernetes]# kubeadm --
config kubeadmCERT.yaml alpha phase kubeconfig all
[kubeconfig] Wrote KubeConfig file to disk: "/etc/kubernetes/admin.conf"
[kubeconfig] Wrote KubeConfig file to disk: "/etc/kubernetes/kubelet.conf"
[kubeconfig] Wrote KubeConfig file to disk: "/etc/kubernetes/controller-manager.conf"
[kubeconfig] Wrote KubeConfig file to disk: "/etc/kubernetes/scheduler.conf"
```

13. 新しいadmin.confファイルをホストにエクスポートします。

```
cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
chown $(id -u):$(id -g) $HOME/.kube/config
chmod 777 $HOME/.kube/config
export KUBECONFIG=.kube/config
```

14. shutdown -r nowを使用してマスター・ノードをリブートします。

15. マスターがバックアップされたら、systemctl status kubeletを介してkubeletが実行されているかどうかを確認してください。

16. kubectl経由でKubernetesのgetノードを確認します。

```
[root@cx-ccs-prod-master-d7f34f25-f524-4f90-9037-7286202ed13a3 ~]# kubectl get nodes
NAME                                                  STATUS     ROLES     AGE
VERSION
cx-ccs-prod-master-d7f34f25-f524-4f90-9037-7286202ed13a1   Ready      master    1y
v1.11.6
cx-ccs-prod-master-d7f34f25-f524-4f90-9037-7286202ed13a2   Ready      master    1y
v1.11.6
cx-ccs-prod-master-d7f34f25-f524-4f90-9037-7286202ed13a3   Ready      master    1y
v1.11.6
```

```
cx-ccs-prod-worker-d7f34f25-f524-4f90-9037-7286202ed13a1   NotReady   <none>    1y
v1.11.6
cx-ccs-prod-worker-d7f34f25-f524-4f90-9037-7286202ed13a2   NotReady   <none>    1y
v1.11.6
cx-ccs-prod-worker-d7f34f25-f524-4f90-9037-7286202ed13a3   NotReady   <none>    1y
v1.11.6
cx-ccs-prod-worker-d7f34f25-f524-4f90-9037-7286202ed13a4   NotReady   <none>    1y
v1.11.6
cx-ccs-prod-worker-d7f34f25-f524-4f90-9037-7286202ed13a5   NotReady   <none>    1y
v1.11.6
cx-ccs-prod-worker-d7f34f25-f524-4f90-9037-7286202ed13a6   NotReady   <none>    1y
v1.11.6
```

17. 各マスター・ノードに対して、ステップ1 ~ 16を繰り返します。

18. 1つのマスターで、**kubeadm token create —print-join-command**を使用して新しい**joinトークンを生成します。**後で使用するためにこのコマンドをコピーします。

```
[root@cx-ccs-prod-master-d7f34f25-f524-4f90-9037-7286202ed13a1 k8s-mgmt]# kubeadm token
create
--print-join-command kubeadm join 192.168.1.14:6443 --token m1ynvj.f4n3et3poki88ry4
--discovery-token-ca-cert-hash
sha256:4d0c569985c1d460ef74dc01c85740285e4af2c2369ff833eed1ba86e1167575
```

19. KubectlのGETノードを介して従業員の**IPを取得します – o wide**.

20. **ssh -i /home/cloud-user/keys/gen3-ao-prod.key cloud-user@192.168.1.17**などのワーカーにログイン**し、ルートアクセス**に移動します。

21. systemctl stop kubeletを介してkubelet**サービスを停止します。**

22. 古い設定ファイル(**ca.crt**、**kubelet**.conf、および**bootstrap-kubelet.conf**を含む)を削**除します**。

```
rm /etc/kubernetes/pki/ca.crt
rm /etc/kubernetes/kubelet.conf
rm /etc/kubernetes/bootstrap-kubelet.conf
```

23. ステップ19からノードの名前を取得します。

24. クラスタに再参加するには、workerコマンドを発行します。18.からのコマンドを使用して**、末尾に – node-name <name of node>を追**加します。

```
[root@cx-ccs-prod-worker-d7f34f25-f524-4f90-9037-7286202ed13a1 kubernetes]# kubeadm join
192.168.1.14:6443 --token m1ynvj.f4n3et3poki88ry4 --discovery-token-ca-cert-hash
sha256:4d0c569985c1d460ef74dc01c85740285e4af2c2369ff833eed1ba86e1167575 --node-name cx-
ccs-prod-worker-d7f34f25-f524-4f90-9037-7286202ed13a1
[preflight] running pre-flight checks
        [WARNING RequiredIPVSKernelModulesAvailable]: the IPVS proxier will not be used,
because the following required kernel modules are not loaded: [ip_vs_rr ip_vs_wrr
ip_vs_sh] or no builtin kernel ipvs support: map[ip_vs:{} ip_vs_rr:{} ip_vs_wrr:{}
ip_vs_sh:{} nf_conntrack_ipv4:{}]
you can solve this problem with following methods:
```

```
 1. Run 'modprobe -- ' to load missing kernel modules;
2. Provide the missing builtin kernel ipvs support

I0226 17:59:52.644282   19170 kernel_validator.go:81] Validating kernel version
I0226 17:59:52.644421   19170 kernel_validator.go:96] Validating kernel config
[discovery] Trying to connect to API Server "192.168.1.14:6443"
[discovery] Created cluster-info discovery client, requesting info from
"https://192.168.1.14:6443"
[discovery] Requesting info from "https://192.168.1.14:6443" again to validate TLS against
the pinned public key
[discovery] Cluster info signature and contents are valid and TLS certificate validates
against pinned roots, will use API Server "192.168.1.14:6443"
[discovery] Successfully established connection with API Server "192.168.1.14:6443"
[kubelet] Downloading configuration for the kubelet from the "kubelet-config-1.11"
ConfigMap in the kube-system namespace
[kubelet] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-
flags.env"
[preflight] Activating the kubelet service
[tlsbootstrap] Waiting for the kubelet to perform the TLS Bootstrap...
[patchnode] Uploading the CRI Socket information "/var/run/dockershim.sock" to the Node
API object "cx-ccs-prod-worker-d7f34f25-f524-4f90-9037-7286202ed13a1" as an annotation

This node has joined the cluster:
* Certificate signing request was sent to master and a response
  was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the master to see this node join the cluster.
```

25. ワーカーを終了し、kubectl getノードを介してマスターのステータスを確認します。[Ready]ステータスになっているはずです。

26. 各作業者について、手順20. ~ 25.を繰り返します。

27. 最後のkubectl getノードは、すべてのノードが「準備完了」状態で、オンラインに戻り、クラスタに参加していることを示します。

```
[root@cx-ccs-prod-master-d7f34f25-f524-4f90-9037-7286202ed13a1 ~]# kubectl get nodes
NAME                                                  STATUS    ROLES     AGE
VERSION
cx-ccs-prod-master-d7f34f25-f524-4f90-9037-7286202ed13a1   Ready     master    1y
v1.11.6
cx-ccs-prod-master-d7f34f25-f524-4f90-9037-7286202ed13a2   Ready     master    1y
v1.11.6
cx-ccs-prod-master-d7f34f25-f524-4f90-9037-7286202ed13a3   Ready     master    1y
v1.11.6
cx-ccs-prod-worker-d7f34f25-f524-4f90-9037-7286202ed13a1   Ready     <none>    1y
v1.11.6
cx-ccs-prod-worker-d7f34f25-f524-4f90-9037-7286202ed13a2   Ready     <none>    1y
v1.11.6
cx-ccs-prod-worker-d7f34f25-f524-4f90-9037-7286202ed13a3   Ready     <none>    1y
v1.11.6
cx-ccs-prod-worker-d7f34f25-f524-4f90-9037-7286202ed13a4   Ready     <none>    1y
v1.11.6
cx-ccs-prod-worker-d7f34f25-f524-4f90-9037-7286202ed13a5   Ready     <none>    1y
v1.11.6
cx-ccs-prod-worker-d7f34f25-f524-4f90-9037-7286202ed13a6   Ready     <none>    1y
v1.11.6
```