Verlopen Kubernetes-certificaat veroorzaakt clusterbrede communicatiestop

Inhoud

Inleiding

In dit document wordt een mogelijk uitvalprobleem beschreven waarmee klanten worden geconfronteerd bij een op Kubernetes gebaseerd systeem dat meer dan 365 dagen is geïnstalleerd. Bovendien worden de nodige stappen ondernomen om de situatie te verbeteren en het op Kubernetes gebaseerde systeem weer op de rails te zetten.

Probleem

Na één jaar van de standaard geïnstalleerde Kubernetes cluster verlopen de client-certificaten af. U hebt geen toegang tot Cisco Cloud Center Suite (CCS). Hoewel het programma nu nog verschijnt, kunt u niet inloggen. Als u naar de kubectl CLI navigeert, ziet u deze fout, "Kan geen verbinding maken met de server: x509: het certificaat is verlopen of is nog niet geldig.".

U kunt dit basisscript gebruiken om de verloopdatum van de certificaten te zien:

```
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
```

U vindt ook een opensource-website voor Action Orchestrator die dit dagelijks controleert en hen waarschuwt voor problemen.

https://github.com/cisco-cx-workflows/cx-ao-shared-workflows/tree/master/CCSCheckKubernetesExpiration__definition_workflow_01E01VIRWZDE24mWlsHrqCGB9xUix0f9ZxG

Oplossing

Nieuwe certificaten moeten opnieuw worden afgegeven via Kubeadm over het cluster en dan moet je de werkknooppunten opnieuw aansluiten bij de meesters.

1. Meld u aan bij een hoofdknooppunt.

2. Ontvang zijn IP-adres **via ip-adresshow**.

```
[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. Navigeer naar de map Kubernetes via **cd/etc/kubernetes**.

4. Maak een bestand met de naam **kubeadmCERT.yaml** via **vi kubeadmCERT.yaml**.

5. Het bestand moet er als volgt uitzien:

```
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. Back-up van uw oude certificaten en toetsen. Dit is niet nodig maar aanbevolen. Maak een reservegids en kopieer deze bestanden.

```
#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. Als u Stap 6 overgeslagen hebt, kunt u de eerder genoemde bestanden gewoon verwijderen via rm-opdracht zoals **rm apiserver.crt**.

8. Navigeer terug naar de locatie van uw **kubeadmCERT.yaml-**bestand. Generate a new apiserver cert via **kubeadm —fig kubeadmCERT.yaml alfa fase certs apiserver**.

```
[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. Nieuwe apiserver-kubelet cert genereren via **kubeadm —Config kubeadmCERT.yaml alfa fase certs apiserver-kubelet-client**.

10. Geneer nieuwe front-proxy-client cert via **kubeadm —** mede **configuratie kubeadmCERT.yaml alpha-certs voor front-proxy-client**.

11. Maak een back-up van de **.conf-bestanden** in de map **enz/**kubernetes. Niet nodig maar aanbevolen. U dient **kubelet.conf** te hebben, **controller-Manager.conf**, **server.conf**, en mogelijk **admin.conf**. Je kunt ze verwijderen als je geen back-up wilt maken.

12. Nieuwe configuratiebestanden genereren via **kubeadm —configuratie kubeadmCERT.yaml alfa fase kubedisine all**.

```
[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. Exporteren van uw nieuwe **admin.conf-**bestand naar de host.

```
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. Herstart het hoofdknooppunt via **afsluiten - of nu**.

15. Controleer, zodra de kapitein steun heeft, of kubelet actief is via **systemische status kubelet**.

16. Controleer Kubernetes via **kubectl**.

```
[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. Herhaal stap 1. t/m 16. voor elk hoofdknooppunt.

18. Aan één meester, genereer een nieuw aansluit token via **kubeadm token creëert —**.. Kopieer die opdracht voor later gebruik.

```
[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. Haal de IP's van je werknemers via **kubectl krijgen knooppunten over de hele wereld**.

20. Meld u aan bij een werker zoals **ssh-i /home/cloud-user/keys/gen3-ao-prod.key cloud-user@192.168.1.17** en navigeer naar worteltoegang.

21. Stop de kubelet service via **systemctl stop kubelet**.

22. Verwijder de oude configuratiebestanden, waaronder **ca.crt**, **kubelet.conf** en **bootstrap-kubelet.conf**.

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

23. Typ de naam van het knooppunt in Stap 19.

24. Geef de opdracht voor de werknemer uit om zich opnieuw bij de cluster aan te sluiten. Gebruik de opdracht vanaf 18.0, maar voeg **—knooppunt-naam <naam van knooppunt>**toe aan het einde.

```
[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. Sluit de arbeider af en controleer de status op een meester via **kubectl knopen**. Het moet in Klaar-status staan.

26. Herhaal stap 20.0 tot 25.0 voor elke werknemer.

27. De laatste **kubectl krijgt knooppunten** die moeten laten zien dat alle knooppunten in "Klaar" status, terug online zijn en zich bij het cluster aansluiten.

```
[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
```