DockerクラスタのCPS-DRA VMの"JOINING"；状態の問題のトラブルシューティング

内容

はじめに

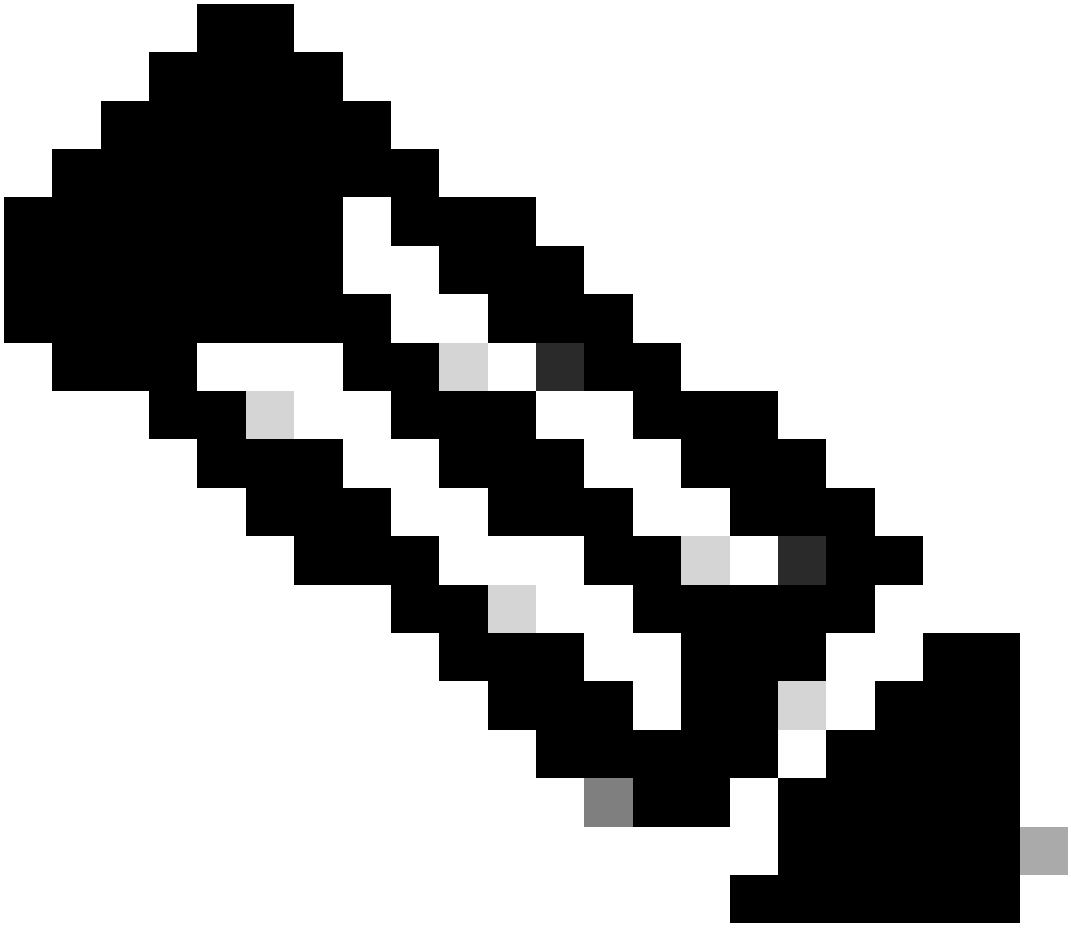
JOINING このドキュメントでは、Cisco Policy Suite(CPS)-Diameter Routing Agent(DRA)仮想マシン(VM)の状態の問題をトラブルシューティングする方法について説明します。

前提条件

要件

次の項目に関する知識があることが推奨されます。

- Linux
- CPS



注:CPS DRA CLIへのルートアクセス権限が必要です。

使用するコンポーネント

このドキュメントの情報は、次のソフトウェアとハードウェアのバージョンに基づいています。

- CPS-DRA 22.2
- ユニファイドコンピューティングシステム(UCS)-B

このドキュメントの情報は、特定のラボ環境にあるデバイスに基づいて作成されました。このドキュメントで使用するすべてのデバイスは、クリアな（デフォルト）設定で作業を開始しています。本稼働中のネットワークでは、各コマンドによって起こる可能性がある影響を十分確認してください。

背景説明

CPS Virtual Diameter Routing Agent(vDRA)は、ネットワーク内の運用コンポーネントとして機能し、ルーティングアルゴリズムを使用して、メッセージを目的の宛先ノードに導きます。

CPS vDRAの中心的な役割は、メッセージルーティングと、その後の元の起点(Point of Origin)への応答の伝送です。

Dockerエンジンを使用してクラスタとしてオーケストレーションされた仮想マシン(VM)の集合から構成されるCPS vDRAは、Master、Control、Director、Distributor、およびWorker VMなどの個別のエンティティで構成されます。

<#root>

admin@orchestrator[master-1]#

show docker engine

```
Fri Jul 14 09:36:18.635 UTC+00:00
MISSED
ID STATUS PINGS
---------------------------------
control-1 CONNECTED 0
control-2 CONNECTED 0
director-1 CONNECTED 0
director-2 CONNECTED 0
director-3 CONNECTED 0
director-4 CONNECTED 0
director-5 CONNECTED 0
director-6 CONNECTED 0
director-7 CONNECTED 0
director-8 CONNECTED 0
distributor-1 CONNECTED 0
distributor-2 CONNECTED 0
distributor-3 CONNECTED 0
distributor-4 CONNECTED 0
master-1 CONNECTED 0
worker-1 CONNECTED 0
worker-2 CONNECTED 0
worker-3 CONNECTED 0
admin@orchestrator[master-1]#
```

Status（ステータス）：スケジューリングアプリケーションがDockerエンジンに接続され、ホストで実行されているかどうかを示します。

Missed ping：特定のホストでpingが連続して失敗した回数。

問題

さまざまな理由で、CPS vDRA VMがJOINING状態のままになる場合があります。

<#root>

admin@orchestrator[master-1]#

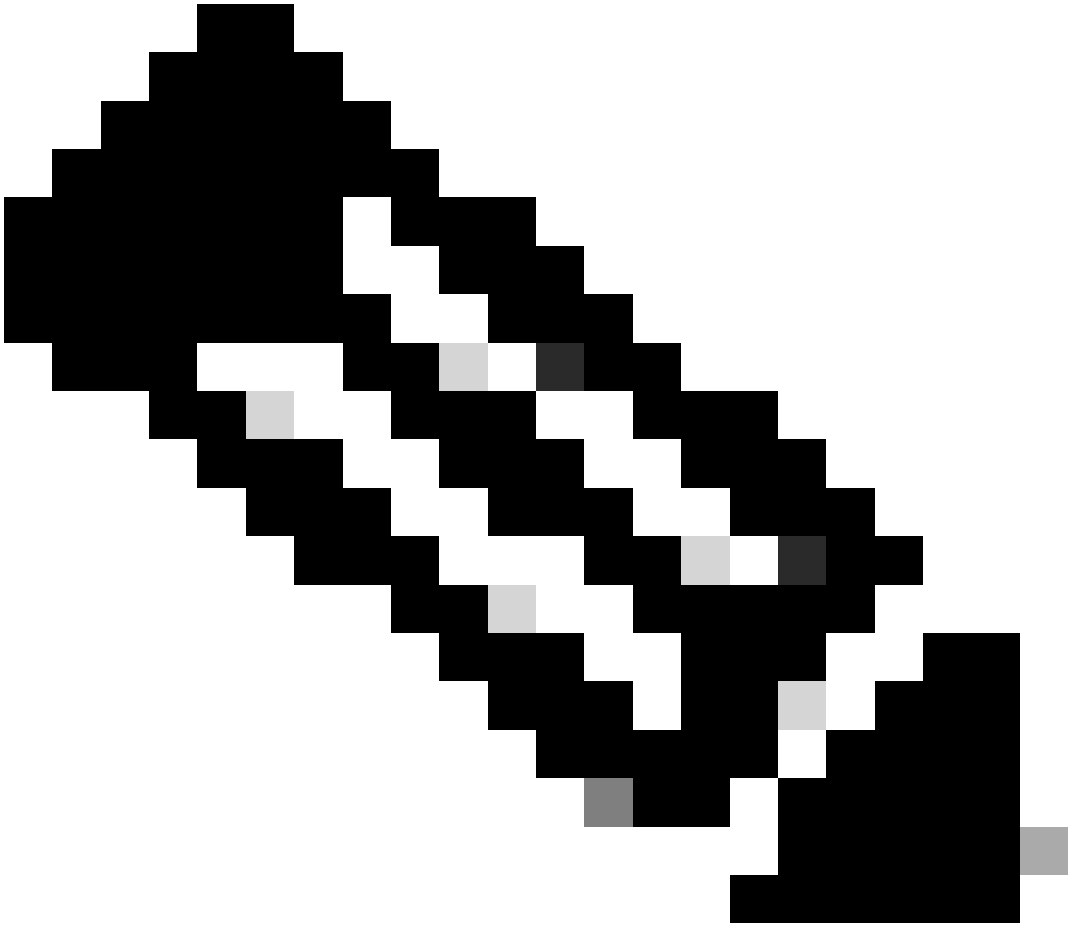
show docker engine

```
Fri Jul 14 09:36:18.635 UTC+00:00
MISSED
ID STATUS PINGS
---------------------------------
control-1 CONNECTED 0
control-2 CONNECTED 0
director-1 JOINING 57
director-2 JOINING 130
director-3 JOINING 131
director-4 JOINING 130
director-5 JOINING 30
director-6 JOINING 129
distributor-1 CONNECTED 0
distributor-2 CONNECTED 0
distributor-3 CONNECTED 0
distributor-4 CONNECTED 0
master-1 CONNECTED 0
worker-1 CONNECTED 0
worker-2 CONNECTED 0
worker-3 CONNECTED 0
admin@orchestrator[master-1]#
```

VMがJOINING状態のままになる理由、

1. マスターVMからVMに到達できない。

1.1.影響を受けるVMのウィーブ接続ステータスがスリーブモードであるかどうかを確認します。



注:Weave Netは、複数のホストに導入されたDockerコンテナを接続する仮想ネットワークを作成し、それらの自動検出を有効にします。Weave Netを使用すると、複数のコンテナで構成されるポータブルマイクロサービスベースのアプリケーションを、1つのホスト上、複数のホスト上、またはクラウドプロバイダーやデータセンター間で、どこでも実行できます。アプリケーションは、ポートマッピング、アンバサダー、またはリンクを設定しなくても、コンテナがすべて同じネットワークスイッチに接続されているかのようにネットワークを使用します。

CPS-DRAには、繰り接続の主要な状態として、fastdpとsleeveの2つがあります。CPS-DRAクラスタ内の設定は、fastdp 状態でのウィーブ接続に一貫して向けられます。

<#root>

cps@director-1:~$

weave status connections

```
-> xx.xx.xx.xx:6783 established sleeve 4e:5f:58:99:d5:65(worker-1) mtu=1438
-> xx.xx.xx.xx:6783 established sleeve 76:33:17:3a:c7:ec(worker-2) mtu=1438
<- xx.xx.xx.xx:54751 established sleeve 76:3a:e9:9b:24:84(director-1) mtu=1438
-> xx.xx.xx.xx:6783 established sleeve 6e:62:58:a3:7a:a0(director-2) mtu=1438
-> xx.xx.xx.xx:6783 established sleeve de:89:d0:7d:b2:4e(director-3) mtu=1438
```

1.2.これらの口グメッセージが影響を受けるVMの口journalctl グに存在するかどうかを確認します。

2023-08-01T10:20:25.896+00:00 docker-engine Docker engine control-1 is unreachable
2023-08-01T10:20:25.897+00:00 docker-engine Docker engine control-2 is unreachable
2023-08-01T10:20:25.935+00:00 docker-engine Docker engine distributor-1 is unreachable
2023-08-01T10:20:25.969+00:00 docker-engine Docker engine worker-1 is unreachable

INFO: 2023/08/02 20:46:26.297275 overlay_switch ->[ee:87:68:44:fc:6a(worker-3)] fastdp timed out waiting for vxlan heartbeat
INFO: 2023/08/02 20:46:26.297307 overlay_switch ->[ee:87:68:44:fc:6a(worker-3)] using sleeve

2. VMディスク領域が枯渇する。

2.1.影響を受けるVMのディスク領域使用率を確認し、ディスク領域使用率の高いパーティションを特定します。

<#root>

cps@control-2:~$

df -h

```
Filesystem Size Used Avail Use% Mounted on
udev 32G 0 32G 0% /dev
tmpfs 6.3G 660M 5.7G 11% /run
/dev/sda3 97G 97G 0 100% /
tmpfs 32G 0 32G 0% /dev/shm
tmpfs 5.0M 0 5.0M 0% /run/lock
tmpfs 32G 0 32G 0% /sys/fs/cgroup
/dev/sdb1 69G 4.7G 61G 8% /data
/dev/sda1 180M 65M 103M 39% /boot
/dev/sdb2 128G 97G 25G 80% /stats
overlay 97G 97G 0 100% /var/lib/docker/overlay2/63854e8173b46727e11de3751c450037b5f5565592b83112a3863fe
overlay 97G 97G 0 100% /var/lib/docker/overlay2/a86da2c7a289dc2b71359654c5160a9a8ae334960e78def78e6eece
overlay 97G 97G 0 100% /var/lib/docker/overlay2/9dfd1bf36282c4e707a3858beba91bfaa383c78b5b9eb3acf0e58f3
overlay 97G 97G 0 100% /var/lib/docker/overlay2/49ee42311e82974707a6041d82e6c550004d1ce25349478bb974cc0
cps@control-2:~$
```

CPS-DRA VMをJOINING状態から回復する手順

アプローチ1:

マスターVMからVMに到達できない場合は、このアプローチを使用します。

1. 影響を受けるVM/sがスリーブモードの場合は、そのVM/sのウィーブ接続ステータスを確認します。

#weave connection status

<#root>

Sample output:

cps@director-1:~$

weave status connections

```
-> xx.xx.xx.xx:6783 established sleeve 4e:5f:58:99:d5:65(worker-1) mtu=1438
-> xx.xx.xx.xx:6783 established sleeve 76:33:17:3a:c7:ec(worker-2) mtu=1438
<- xx.xx.xx.xx:54751 established sleeve 76:3a:e9:9b:24:84(director-1) mtu=1438
-> xx.xx.xx.xx:6783 established sleeve 6e:62:58:a3:7a:a0(director-2) mtu=1438
-> xx.xx.xx.xx:6783 established sleeve de:89:d0:7d:b2:4e(director-3) mtu=1438
```

2. それぞれのVMでweaveを再起動します。

#docker restart weave

3. ウィーブ接続の状態がfastdp状態に移行し、影響を受けるVMがCONNECTED状態に移行したかどうかを確認します。

4. VMがまだJOINING状態のままである場合、影響を受けるVMを再起動します。

<#root>

#sudo reboot now

or

#init 6

5. ここで、影響を受けるVMがCONNECTED状態に移行したかどうかを確認します。

<#root>

admin@orchestrator[master-1]#

show docker engine

```
Fri Jul 14 09:36:18.635 UTC+00:00
MISSED
ID STATUS PINGS
---------------------------------
control-1 CONNECTED 0
control-2 CONNECTED 0
director-1 CONNECTED 0
director-2 CONNECTED 0
director-3 CONNECTED 0
director-4 CONNECTED 0
distributor-1 CONNECTED 0
distributor-2 CONNECTED 0
distributor-3 CONNECTED 0
distributor-4 CONNECTED 0
master-1 CONNECTED 0
worker-1 CONNECTED 0
worker-2 CONNECTED 0
worker-3 CONNECTED 0
admin@orchestrator[master-1]#
```

6. vPASがトラフィックのケータリングを開始し、すべてのコンテナがアップ状態（特に直径エンドポイント）であることを確認します。そうでない場合は、drc01 VMでコンテナを再起動しますorchestrator-backup-a。

#docker restart orchestrator-backup-a

7. ここで、vPASがトラフィックの処理を開始したかどうかを確認します。

アプローチ2:

VMのディスク領域が枯渇した場合。

1. ディスク領域を大量に消費しているディレクトリを特定します。

<#root>

root@control-2:/var/lib/docker/overlay2#

du -ah / --exclude=/proc | sort -r -h | head -n 10

176G 9dfd1bf36282c4e707a3858beba91bfaa383c78b5b9eb3acf0e58f335126d9b7

2. 大量のディスク領域を消費するファイル/ログ/ダンプを検証します。

<#root>

root@control-2:/var/lib/docker/overlay2/9dfd1bf36282c4e707a3858beba91bfaa383c78b5b9eb3acf0e58f335126d9b7/diff#

ls -lrtha | grep G

```
total 88G
-rw------- 1 root root 1.1G Jul 12 18:10 core.22781
-rw------- 1 root root 1.2G Jul 12 18:12 core.24213
-rw------- 1 root root 1.2G Jul 12 18:12 core.24606
-rw------- 1 root root 1.1G Jul 12 18:12 core.24746
-rw------- 1 root root 1.1G Jul 12 18:13 core.25398
```

3. 影響を受けるVMで実行されているコンテナー（特に異常なコンテナー）を特定します。

<#root>

admin@orchestrator[master-1]#

show docker service | exclude HEALTHY

```
Fri Jul 14 09:37:20.325 UTC+00:00
PENALTY
MODULE INSTANCE NAME VERSION ENGINE CONTAINER ID STATE BOX MESSAGE
--------------------------------------------------------------------------------------------------------------
cc-monitor 103 cc-monitor 22.1.1-release control-2 cc-monitor-s103 STARTED true Pending health check
mongo-node 103 mongo-monitor 22.1.1-release control-2 mongo-monitor-s103 STARTED true Pending health ch
mongo-status 103 mongo-status 22.1.1-release control-2 mongo-status-s103 STARTED false -
policy-builder 103 policy-builder 22.1.1-release control-2 policy-builder-s103 STARTED true Pending hea
prometheus 103 prometheus-hi-res 22.1.1-release control-2 prometheus-hi-res-s103 STARTED true Pending h
prometheus 103 prometheus-planning 22.1.1-release control-2 prometheus-planning-s103 STARTED false -

admin@orchestrator[master-1]#
```

4. 大きなコアファイルをトリガーするコンテナを特定し、影響を受けるVMでホストされている各コンテナを1つずつ検査します

。

<#root>

Sample output for container "cc-monitor-s103":

root@control-2:/var/lib/docker/overlay2/9dfd1bf36282c4e707a3858beba91bfaa383c78b5b9eb3acf0e58f335126d9b7/merged#

docker inspect cc-monitor-s103| grep /var/lib/docker/overlay2/| grep merged

```
"MergedDir": "/var/lib/docker/overlay2/9dfd1bf36282c4e707a3858beba91bfaa383c78b5b9eb3acf0e58f335126d9b7/
root@control-2:/var/lib/docker/overlay2/9dfd1bf36282c4e707a3858beba91bfaa383c78b5b9eb3acf0e58f335126d9b
```

5. その特定のコンテナへのアクセス権があるかどうかを確認します。

#admin@orchestrator[master-0]# docker connect cc-monitor-s103

6. そのコンテナにアクセスできない場合は、大きなコアファイルを削除して空き領域を増やします。

<#root>

root@control-2:/var/lib/docker/overlay2/9dfd1bf36282c4e707a3858beba91bfaa383c78b5b9eb3acf0e58f335126d9b7/diff#

rm -rf core*

7. 影響を受けるVMから影響を受けるコンテナにログインします。

<#root>

#docker exec -it cc-monitor-s103 bash

8. コンテナ内のappプロセスを再起動して、大きなコアファイルの生成を停止します。

<#root>

root@cc-monitor-s103:/#

supervisorctl status

app STARTING

app-logging-status RUNNING pid 30, uptime 21 days, 23:02:17
consul RUNNING pid 26, uptime 21 days, 23:02:17
consul-template RUNNING pid 27, uptime 21 days, 23:02:17
haproxy RUNNING pid 25, uptime 21 days, 23:02:17
root@cc-monitor-s103:/#

root@cc-monitor-s103:/# date;

supervisorctl restart app

Fri Jul 14 09:08:38 UTC 2023
app: stopped
app: started
root@cc-monitor-s103:/#

root@cc-monitor-s103:/#

supervisorctl status

app RUNNING pid 26569, uptime 0:00:01
app-logging-status RUNNING pid 30, uptime 21 days, 23:02:44
consul RUNNING pid 26, uptime 21 days, 23:02:44
consul-template RUNNING pid 27, uptime 21 days, 23:02:44
haproxy RUNNING pid 25, uptime 21 days, 23:02:44
root@cc-monitor-s103:/#

9. 手順8.でバルクコアファイルの生成を停止できない場合は、影響を受けるコンテナを再起動します。

<#root>

#

docker restart cc-monitor-s103

10. バルク・コア・ファイルの生成が停止しているかどうかを確認します。

11. 影響を受けるVMを接続済みの状態に戻すには、コンテナにログインしてorchestrator、orchestration-engine 再起動を実行します。

<#root>

cps@master-1:~$ date;

docker exec -it orchestrator bash

```
Fri Jul 14 09:26:12 UTC 2023
root@orchestrator:/#
```

<#root>

root@orchestrator:/#

supervisorctl status

```
confd RUNNING pid 20, uptime 153 days, 23:33:33
consul RUNNING pid 19, uptime 153 days, 23:33:33
consul-template RUNNING pid 26, uptime 153 days, 23:33:33
haproxy RUNNING pid 17, uptime 153 days, 23:33:33
mongo RUNNING pid 22, uptime 153 days, 23:33:33
monitor-elastic-server RUNNING pid 55, uptime 153 days, 23:33:33
monitor-log-forward RUNNING pid 48, uptime 153 days, 23:33:33
orchestration-engine RUNNING pid 34, uptime 153 days, 23:33:33
orchestrator_back_up RUNNING pid 60, uptime 153 days, 23:33:33
remove-duplicate-containers RUNNING pid 21, uptime 153 days, 23:33:33
rolling-restart-mongo RUNNING pid 18, uptime 153 days, 23:33:33
simplehttp RUNNING pid 31, uptime 153 days, 23:33:33
root@orchestrator:/#
```

<#root>

root@orchestrator:/# date;

supervisorctl restart orchestration-engine

```
Fri Jul 14 09:26:39 UTC 2023
```

```
orchestration-engine: stopped
orchestration-engine: started
root@orchestrator:/#
```

12. ステップ11でVMを復元できない場合は、影響を受けるVMでengine-poxy再起動を実行します。

<#root>

cps@control-2:~$

docker ps | grep engine

```
0b778fae2616 engine-proxy:latest "/w/w /usr/local/bin…" 5 months ago Up 3 weeks
```

engine-proxy-ddd7e7ec4a70859b53b24f3926ce6f01

<#root>

cps@control-2:~$

docker restart engine-proxy-ddd7e7ec4a70859b53b24f3926ce6f01

```
engine-proxy-ddd7e7ec4a70859b53b24f3926ce6f01
cps@control-2:~$
```

<#root>

cps@control-2:~$

docker ps | grep engine

```
0b778fae2616 engine-proxy:latest "/w/w /usr/local/bin…" 5 months ago Up 6 seconds engine-proxy-ddd7e7ec
cps@control-2:~$
```

13. ここで、影響を受けるVMがCONNECTED状態に移行したかどうかを確認します。

<#root>

admin@orchestrator[master-1]#

show docker engine

```
Fri Jul 14 09:36:18.635 UTC+00:00
ID STATUS MISSED PINGS
-----------------------------------
control-1 CONNECTED 0
control-2 CONNECTED 0
director-1 CONNECTED 0
```

```
director-2 CONNECTED 0
director-3 CONNECTED 0
director-4 CONNECTED 0
distributor-1 CONNECTED 0
distributor-2 CONNECTED 0
distributor-3 CONNECTED 0
distributor-4 CONNECTED 0
master-1 CONNECTED 0
worker-1 CONNECTED 0
worker-2 CONNECTED 0
worker-3 CONNECTED 0
admin@orchestrator[master-1]#
```

翻訳について
シスコは世界中のユーザにそれぞれの言語でサポート コンテンツを提供するために、機械と人による翻訳を組み合わせて、本ドキュメントを翻訳しています。ただし、最高度の機械翻訳であっても、専門家による翻訳のような正確性は確保されません。シスコは、これら翻訳の正確性について法的責任を負いません。原典である英語版（リンクからアクセス可能）もあわせて参照することを推奨します。