ネットワークの上のコンテナネットワーク

Shishio Tsuchiya
shtsuchi@arista.com

ARISTA

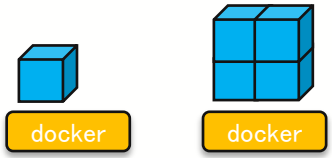
Agenda

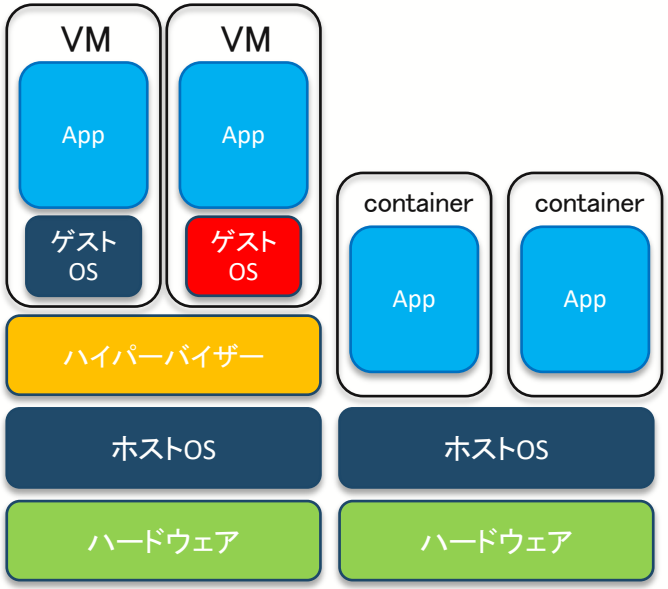
- コンテナとネットワーク
- ネットワーク機器上のコンテナ
- ネットワーク検証の為のコンテナ
- ネットワーク装置のインフラとしてのコンテナ

Containerとは

- Linux(現在はwindowsも)におけるリソースを分離させる手法
- OSレベルの仮想化
  - cgroups(コントロールグループ)
  - namespaces
- Kernelは共有される

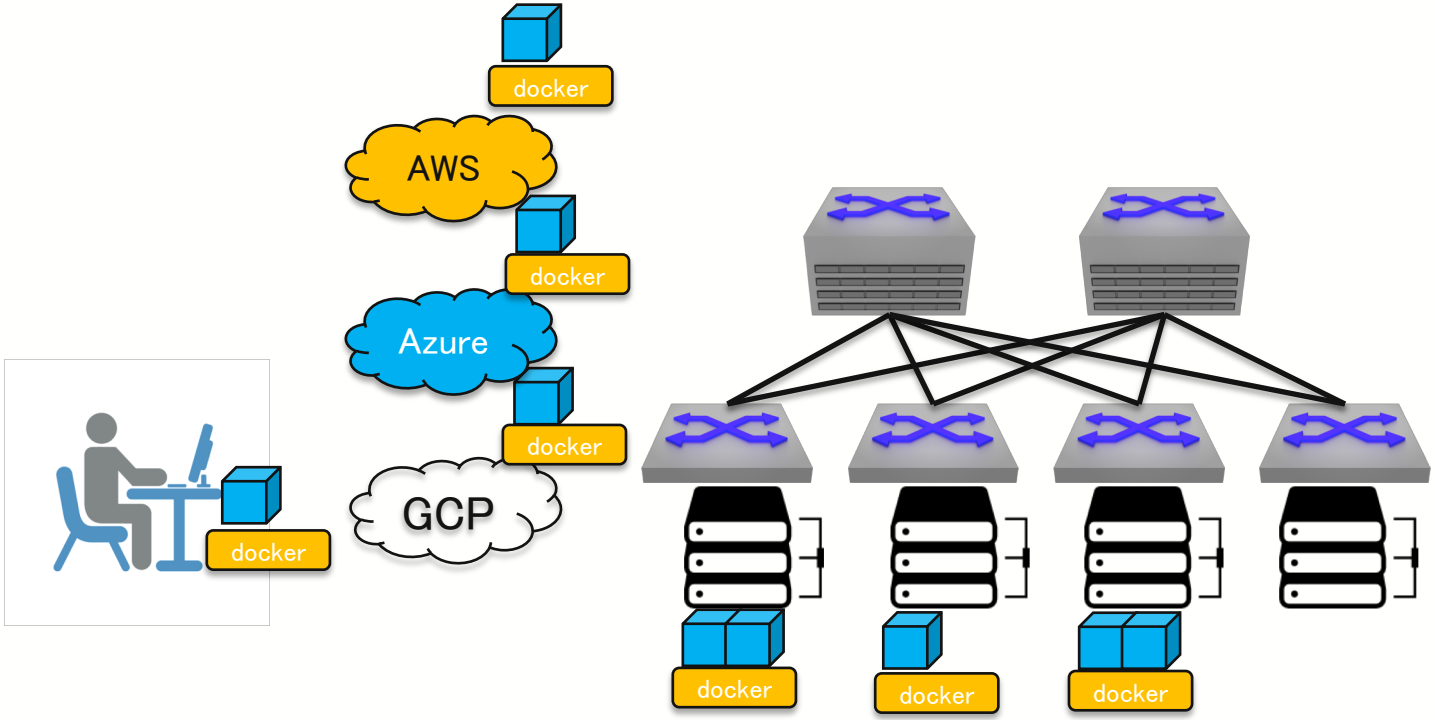
- Dockerにてコンテナ型アプリケーション実行環境を提供し、ライブラリ/環境変数などを保存/移動/展開可能





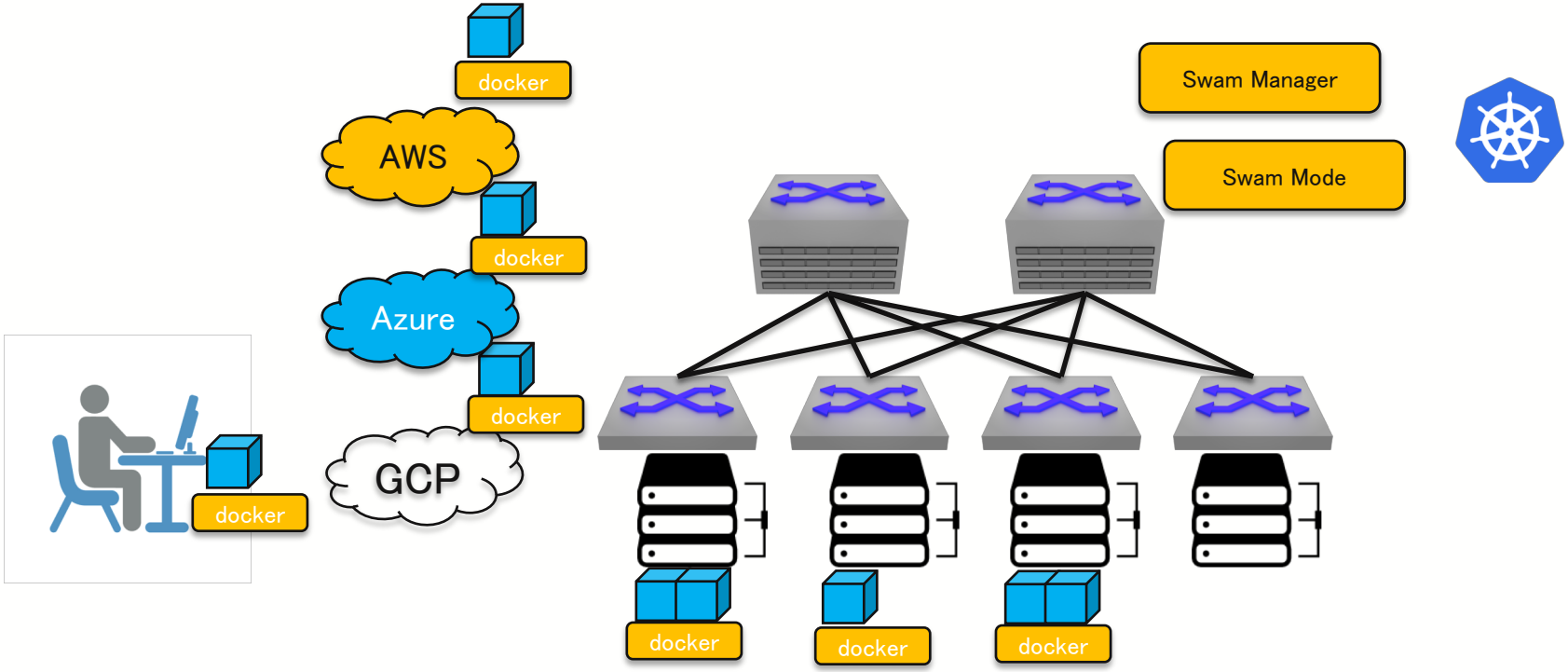
ARISTA

コンテナの特徴



- オンプレ/クラウドどこにおいても同じ環境で開発が可能

誰がコンテナクラスタを管理するのか?



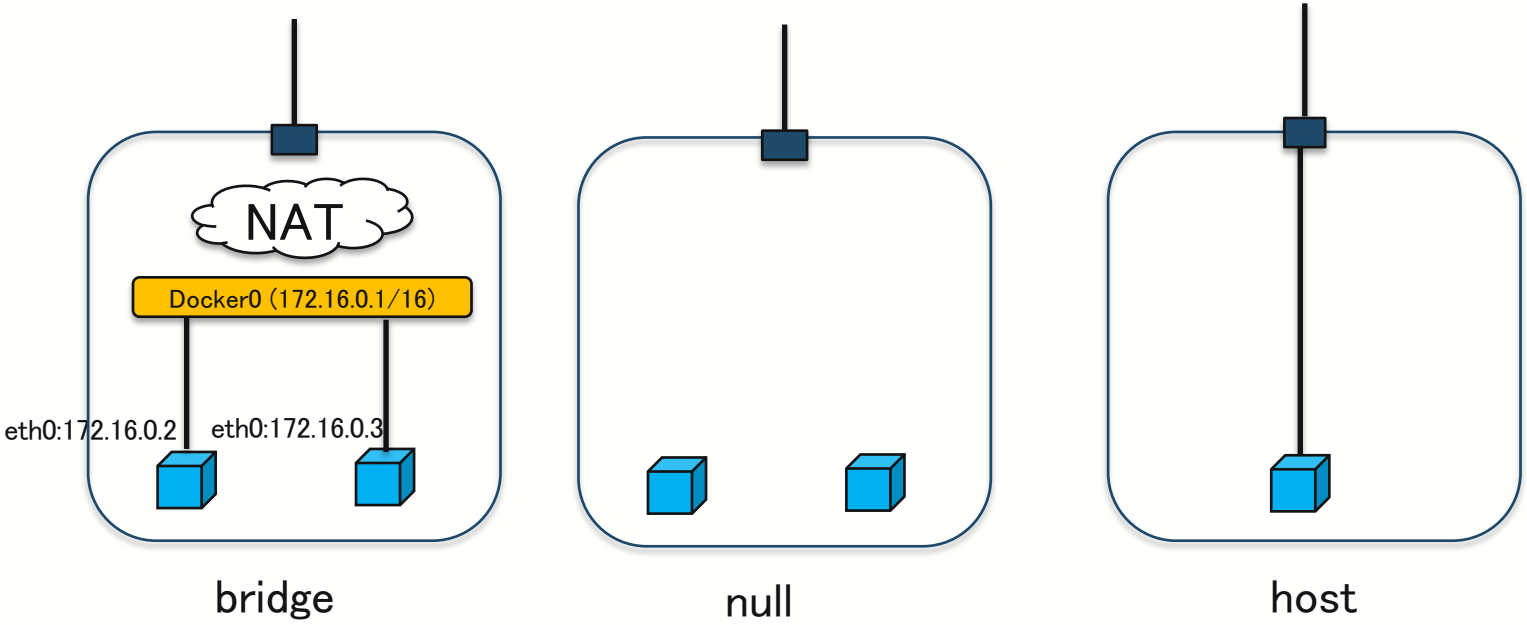
- Docker Swam Manager/Swam ModeやKubernetes(k8s)でリソースを管理
- 自動修復/自動複製/自動スケーリングなども可能

Dockerネットワークタイプ比較

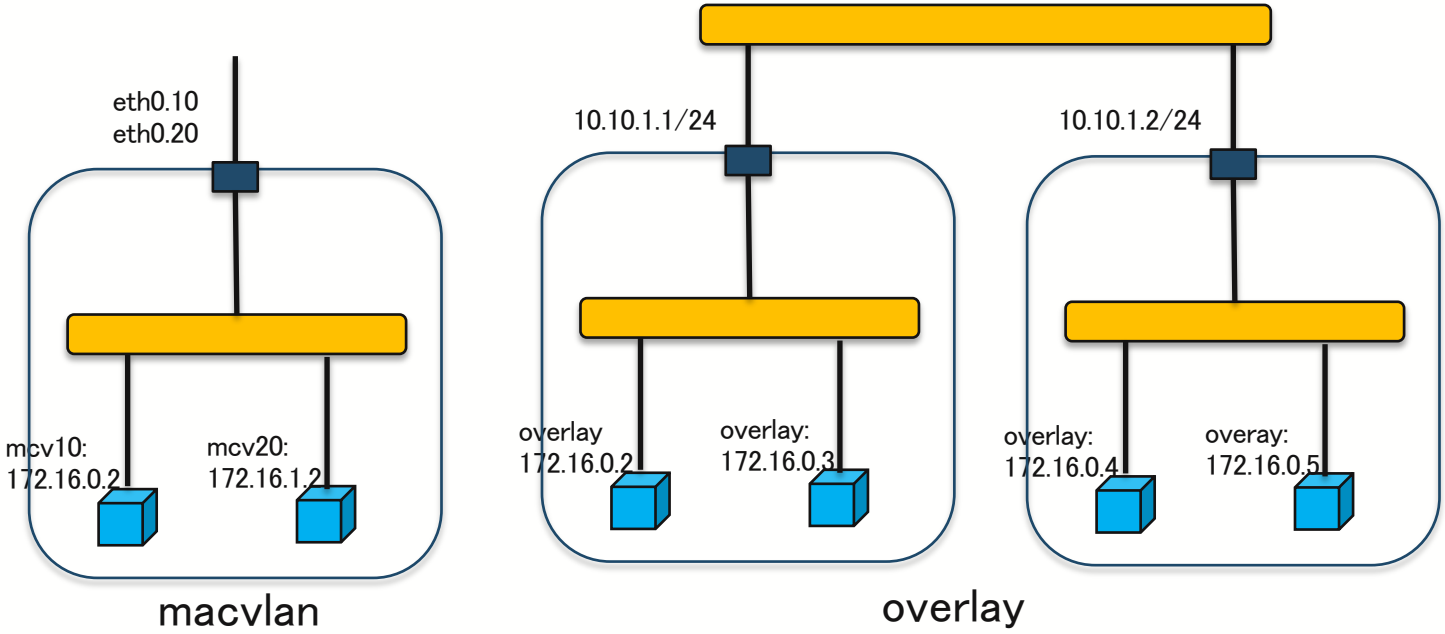
ネットワークタイプ	使用方法
bridge	• デフォルト • コンテナ間の接続にLinux bridgeを使用 • インバウンド/アウトバウンドトラフィックにはiptablesを使用
null	• コンテナでネットワーク・インターフェースを作成しない
host	• ホストネットワーク・インターフェースをコンテナに直接接続する
overlay	• ホスト間にマルチホストのレイヤー2ネットワークを作成 • Linuxブリッジを利用して、コンテナをオーバーレイで接続 • VXLANエンキャップスレーション
macvlan	• VLAN単位でMACアドレスを提供 • 802.1qトランクの上流とサブインターフェースで組み合わせ可 • コンテナ毎のMACアドレス

- コンテナとネットワークの接続の仕方は上記の様なものがある

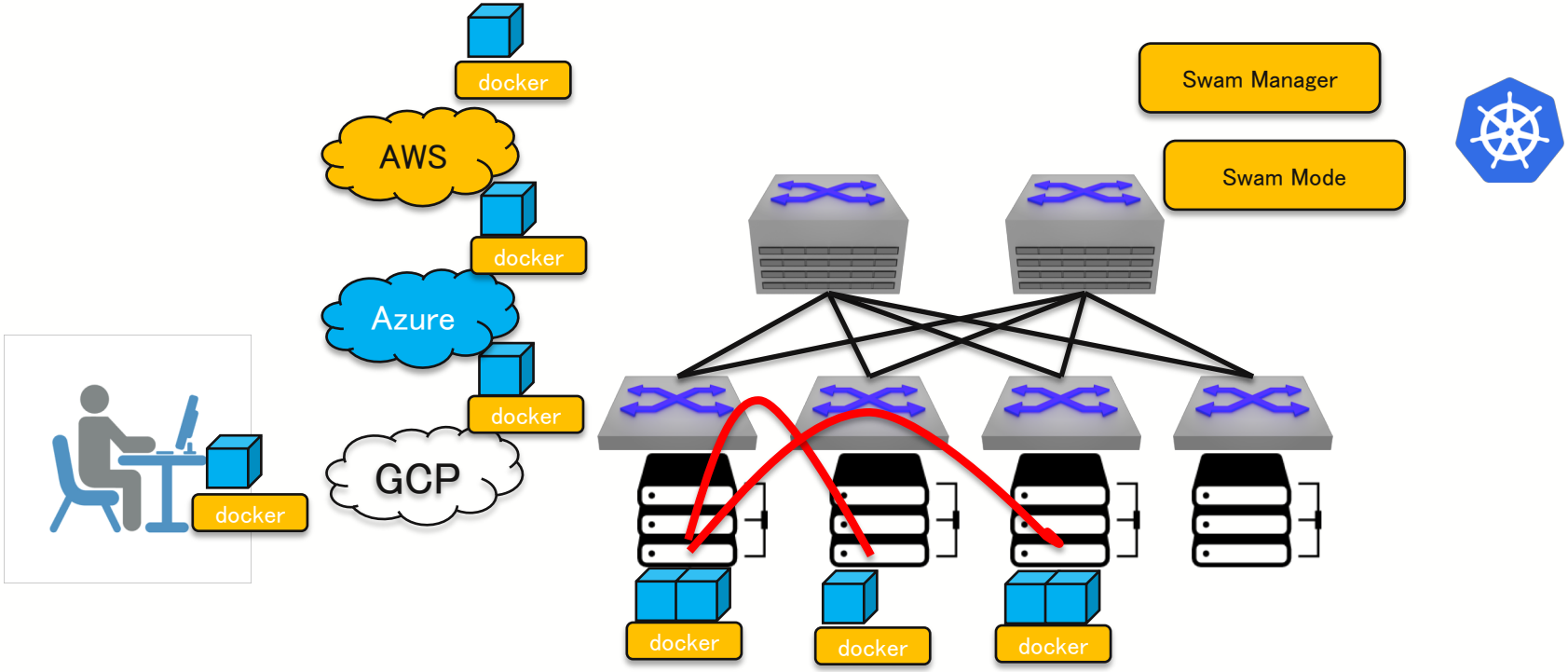
Dockerネットワークタイプ



Dockerネットワークタイプ

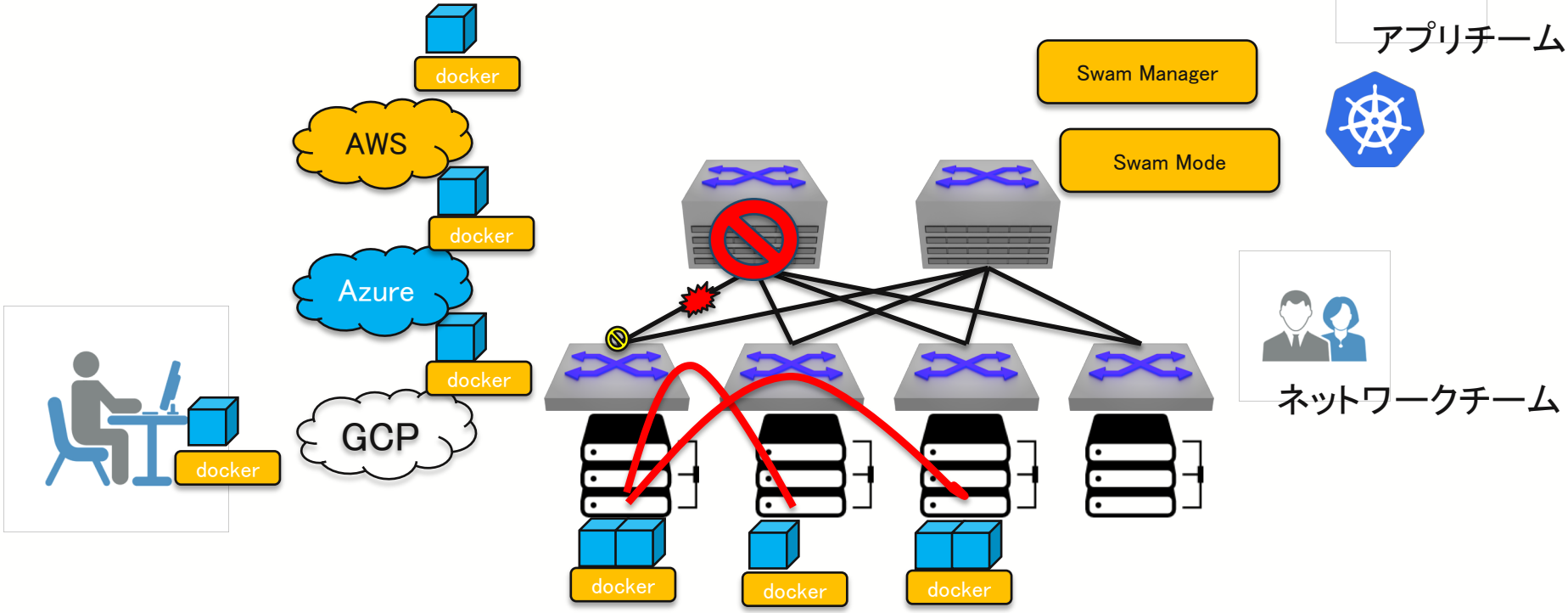


コンテナのネットワーク作り



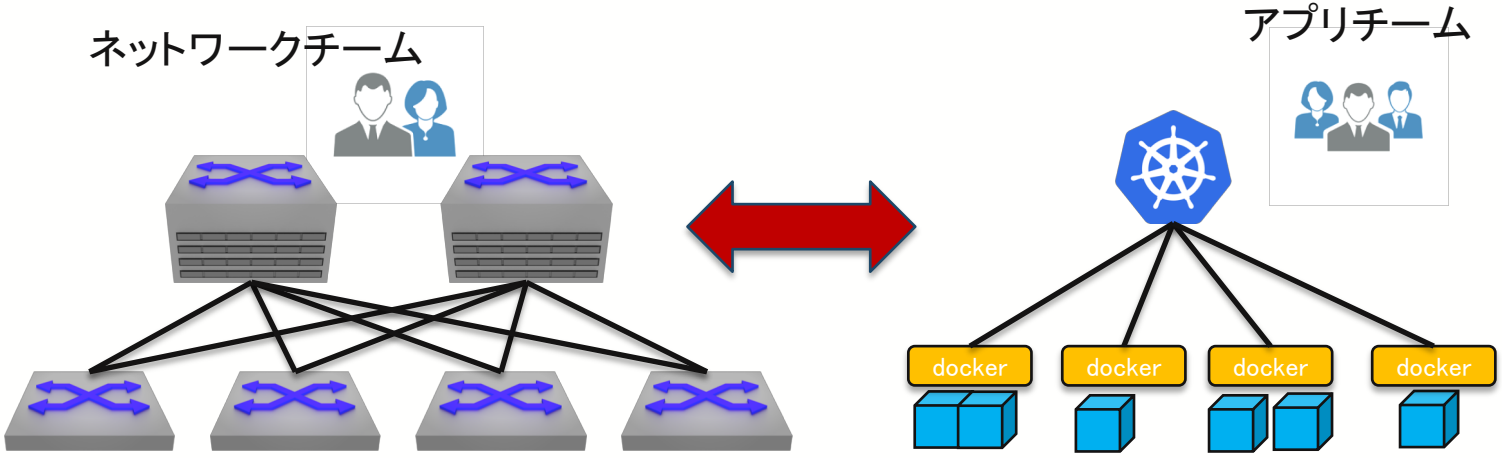
- overlayを使ったコンテナ間のvxlanを実現出来る
- アンダーレイのネットワークに依存せず、柔軟なLayer2ネットワークを構築可能

ネットワークチームが知ってる事



- 詳細ネットワーク・トポロジー
- インターフェース障害/Queue枯渇/機器メンテナンス

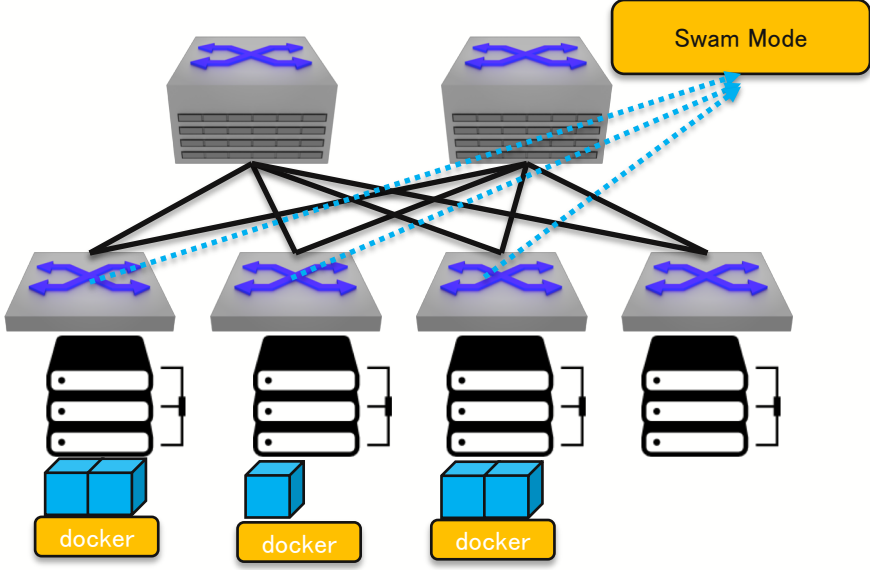
情報差分を埋める事で



- 平均復旧時間(MTTR)の削減と可視化の向上

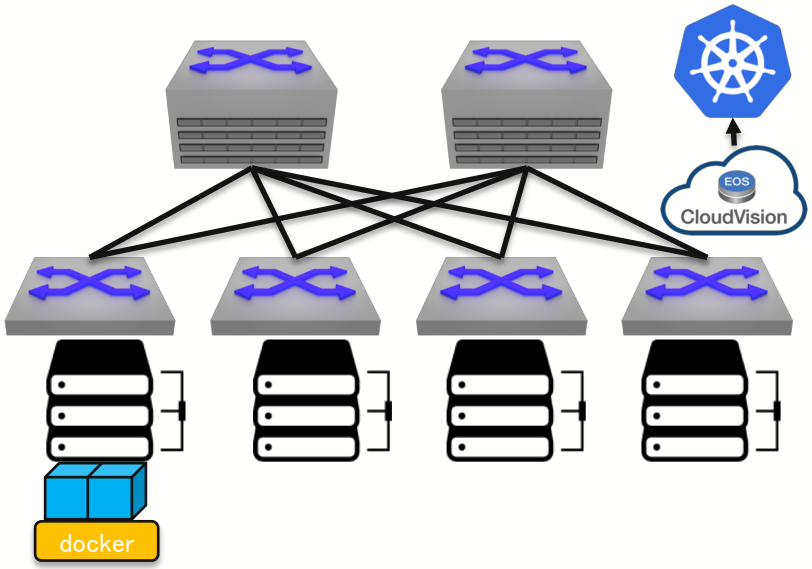
一つの解決策

```
Arista# show containertracer swarm
SERVICE         CONTAINER           NODE          SERVICE              LOCAL
NAME            ID                                PORTS                INTERFACE
--------------- ------------------- ------------ -------------------- ----------
acb             06b146f697cc        lnx151        []                   Ethernet15
nginx           1acda552204c        lnx150        ['80:80']            Ethernet13
testport        95280d29c276        lnx150        ['5000:5000']        Ethernet13
acb             de38f69b96a4        lnx150        []                   Ethernet13
acb             37a063d10fbc        lnx151        []                   Ethernet15
testport        b7aabc91a93f        lnx151        ['5000:5000']        Ethernet15
acb             997286fcfcbb        lnx150        []                   Ethernet13
```



- 隣接ノードを理解出来るスイッチよりSwamModeにAPIで問い合わせで自ノードに接続されているホストの先のコンテナ情報をマッピングする

k8sの場合

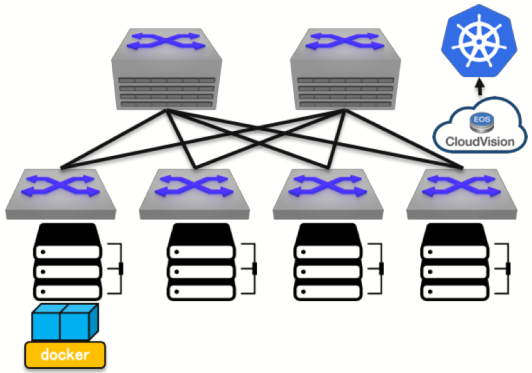


- よりダイナミックの動作が考えられる為、全体を把握してるものからの確認が必要

kubectl

```
[fredlhsu@kube-138 ~]$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
ceos-init-nocni-tg5np               1/1     Running   1          38d
skodb-deployment-69b4bfb948-t4kpd   2/2     Running   0          23d
[fredlhsu@kube-138 ~]$ kubectl get pods -o wide
NAME                                READY   STATUS    RESTARTS   AGE   IP              NODE                                NOMINATED NODE   READINESS GATES
ceos-init-nocni-tg5np               1/1     Running   1          38d   10.90.224.138   kube-138.sjc.aristanetworks.com   <none>           <none>
skodb-deployment-69b4bfb948-t4kpd   2/2     Running   0          23d   10.110.0.31     kube-138.sjc.aristanetworks.com   <none>           <none>
```

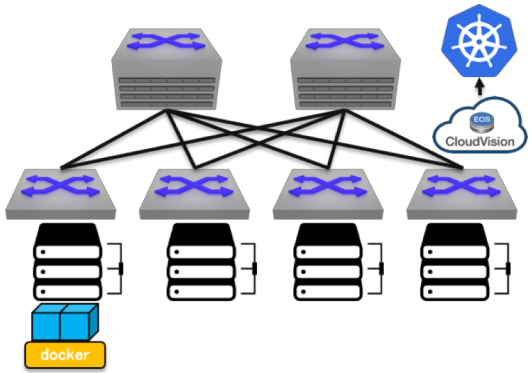
- kubectl get pod
  - POD一覧を表示
  - –wideで各Podの実行ホストの情報も表示



Container tracer

```
cvx-coresite#show service container tracer cluster demo pods
Node Name                          Switch                             Interface       Host Interface      Pod Name                                                    Pod Status
---------------------------------- ---------------------------------- --------------- ------------------- ----------------------------------------------------------- ----------
kube-138.sjc.aristanetworks.com    cs-lf14.sjc.aristanetworks.com     Ethernet26      Ethernet1           calico-node-b4djr                                           Running
                                                                                                          ceos-init-nocni-tg5np                                       Running
                                                                                                          coredns-86c58d9df4-j5mxh                                    Running
                                                                                                          coredns-86c58d9df4-q46b2                                    Running
                                                                                                          etcd-kube-138.sjc.aristanetworks.com                        Running
                                                                                                          kube-apiserver-kube-138.sjc.aristanetworks.com              Running
                                                                                                          kube-controller-manager-kube-138.sjc.aristanetworks.com     Running
                                                                                                          kube-proxy-jskg6                                            Running
                                                                                                          kube-scheduler-kube-138.sjc.aristanetworks.com              Running
                                                                                                          skodb-deployment-69b4bfb948-t4kpd                           Running


cvx-coresite#
cvx-coresite#
```

- kubectlの情報と自身が持っているネットワーク全体のトポロジー情報を組み合わせて表示

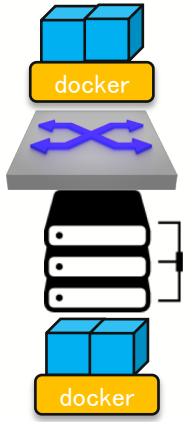


Agenda

- コンテナとネットワーク
- ネットワーク機器上のコンテナ
- ネットワーク検証の為のコンテナ
- ネットワーク装置のインフラとしてのコンテナ

ネットワーク機器でのコンテナ利用

- ネットワーク機器もテレメトリー/コントローラーなど複雑なインフラアプリケーションも増えてきている
- ネットワークOSでコンテナ対応可能なものも増えている
  - Docker on EOS
    - >> https://youtu.be/P3wSSgtHe4k



アプリケーション例

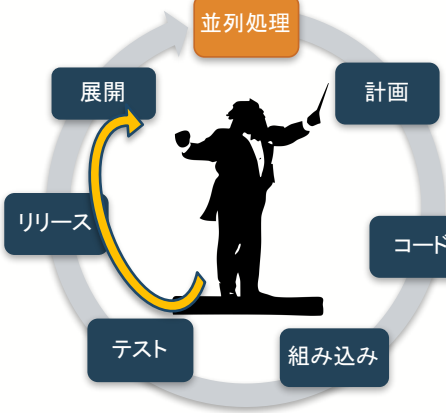
- sflowtool
  - http://www.inmon.com/technology/sflowTools.php
  - sflowデータを解析するためのコマンドラインとスクリプトを提供
  - ただし、ソースコードで提供されている為、環境毎にコンパイルが必要
  - コンテナで提供:https://store.docker.com/community/images/fredhsu/sflowtool
- HAProxy
  - TCPロードバランサのHAProxy機能をスイッチにインストール
  - 小規模/簡易サーバー環境を構築
  - https://hub.docker.com/_/haproxy/
- ルーティング
  - 独自ルーティングプロトコルをコンテナ上のOSSで受けて、eBGPに再配信
  - 正確にいうとEIGRPをFRRで受けて、eBGPに再配信

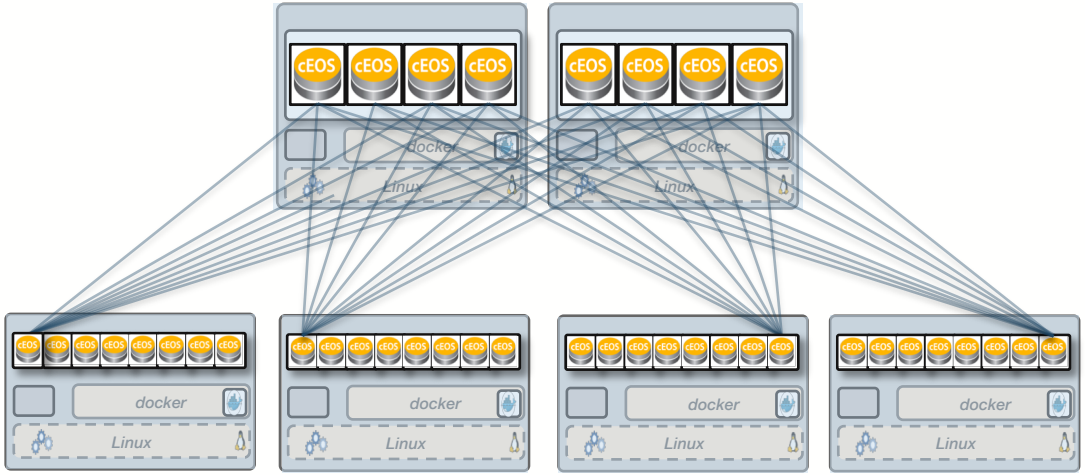
Agenda

- コンテナとネットワーク
- ネットワーク機器上のコンテナ
- ネットワーク検証の為のコンテナ
- ネットワーク装置のインフラとしてのコンテナ

シュミレーションと検証の為のコンテナ利用

クラウドの為の大規模スケールでのネットワークデザイン
とアプリケーションの継続的なテスト、組み込み、展開

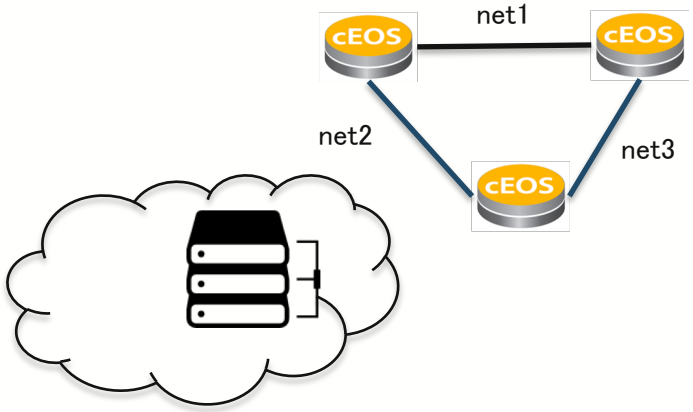




ネットワークモデリングのための高速で軽量なアプローチ

ARISTA

cEOS-Lab

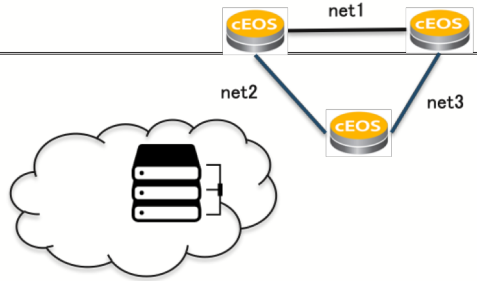


https://youtu.be/hcI_FW_mAt4

- クラウド上のUbuntsuにコンテナ版EOS cEOS-labを展開
- 3つのコンテナイメージを立ち上げ、相互に接続する
- OSPFのネットワーク環境を作る
- 手元(MacOS)でも同様な環境を構築する

cEOS-Lab

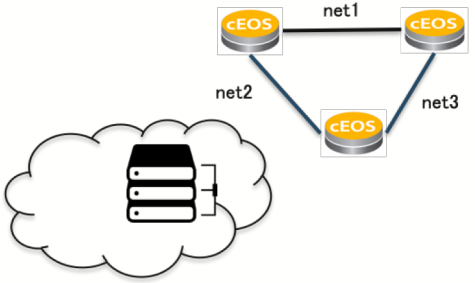
```
arista@arista:~$ uname -a
Linux arista 4.4.0-141-generic #167-Ubuntu SMP Wed Dec 5 10:40:15 UTC 2018 x86_64 x86_64 x86_64 GNU/Linux
arista@arista:~$ docker ps
CONTAINER ID        IMAGE                       COMMAND                  CREATED             STATUS              PORTS                                      NAMES
fc8b329300d8        ceosimage:latest            "/sbin/init"             16 hours ago        Up About an hour    0.0.0.0:5443->5443/tcp                     ceos3
057286a81654        ceosimage:latest            "/sbin/init"             16 hours ago        Up About an hour    0.0.0.0:5442->5442/tcp                     ceos2
afbce3a6b9f3        ceosimage:latest            "/sbin/init"             16 hours ago        Up About an hour    0.0.0.0:5441->5441/tcp                     ceos1
c461f8cfc166        ansible/awx_task:latest     "/tini -- /bin/sh -c…"   14 months ago       Up About an hour    8052/tcp                                   awx_task
f4aceb1803a3        ansible/awx_web:latest      "/tini -- /bin/sh -c…"   14 months ago       Up About an hour    0.0.0.0:80->8052/tcp                       awx_web
d3792424571d        memcached:alpine            "docker-entrypoint.s…"   14 months ago       Up About an hour    11211/tcp                                  memcached
6f867b6d9d63        rabbitmq:3                  "docker-entrypoint.s…"   14 months ago       Up About an hour    4369/tcp, 5671-5672/tcp, 25672/tcp         rabbitmq
7b349517d521        postgres:9.6                "docker-entrypoint.s…"   14 months ago       Up About an hour    5432/tcp                                   postgres
arista@arista:~$ docker network ls
NETWORK ID          NAME                DRIVER              SCOPE
2e3943d9f232        bridge              bridge              local
7fb729a7212d        host                host                local
e8eef88f1d74        net1                bridge              local
1b2f932d2658        net2                bridge              local
814aaf20b687        net3                bridge              local
b0dbf134d48f        none                null                local
arista@arista:~$
```



ARISTA

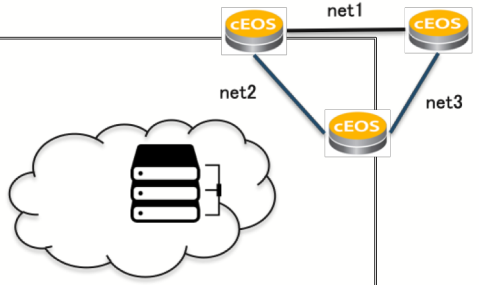
cEOS-Lab

```
arista@arista:~$ docker network inspect net1 net2 net3
[
    {
        "Name": "net1",
        "Id": "e8eef88f1d74737cf88714889c780aed699c12fedeef08550df6c7d62d56b59c",
        "Created": "2019-01-19T19:49:37.58356277+09:00",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": {},
            "Config": [
                {
                    "Subnet": "172.21.0.0/16",
                    "Gateway": "172.21.0.1"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": {
            "057286a816545ba149965de2e10f294093654d8851d1d5a934a94d86a032f167": {
                "Name": "ceos2",
                "EndpointID": "5a9d4179b5ac764efa946b4f85eb94bf15f00ca3d80afa521e5e7e5276e2ae0e",
                "MacAddress": "02:42:ac:15:00:03",
                "IPv4Address": "172.21.0.3/16",
                "IPv6Address": ""
            },
            "afbce3a6b9f3ddc2696be1a8c8ea3eb05bd76f1a6cb90b9c7a96103e24543dff": {
                "Name": "ceos1",
                "EndpointID": "ad25fa61107c1f32bddfb8db089ca83781ffd2ca0c0b31914816256d04a9719c",
                "MacAddress": "02:42:ac:15:00:02",
                "IPv4Address": "172.21.0.2/16",
-snip-
```



ARISTA

cEOS-Lab

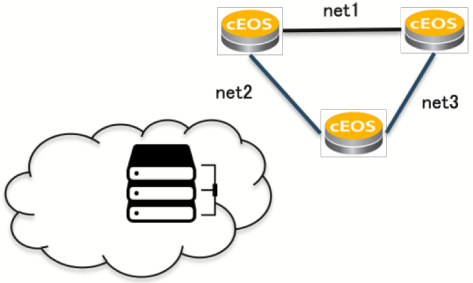



```
arista@arista:~$ docker exec -it ceos1 Cli
cEOS1>ena
cEOS1#show version
 cEOSLab
Hardware version:
Serial number:
System MAC address:  0242.ac2a.8f7d

Software image version: 4.21.3F-10977770.4213F (engineering build)
Architecture:           i386
Internal build version: 4.21.3F-10977770.4213F
Internal build ID:      3bfe7e26-c46a-40e0-a112-5229042ace9f

cEOS tools version: 1.1

Uptime:                 0 weeks, 0 days, 1 hours and 10 minutes
Total memory:           4046124 kB
Free memory:            1259188 kB

cEOS1#
cEOS1#exit
arista@arista:~$ docker exec -it ceos1 bash
bash-4.3# uname -a
Linux cEOS1 4.4.0-141-generic #167-Ubuntu SMP Wed Dec 5 10:40:15 UTC 2018 x86_64 x86_64 x86_64 GNU/Linux
bash-4.3#
```


ARISTA

cEOS-Lab

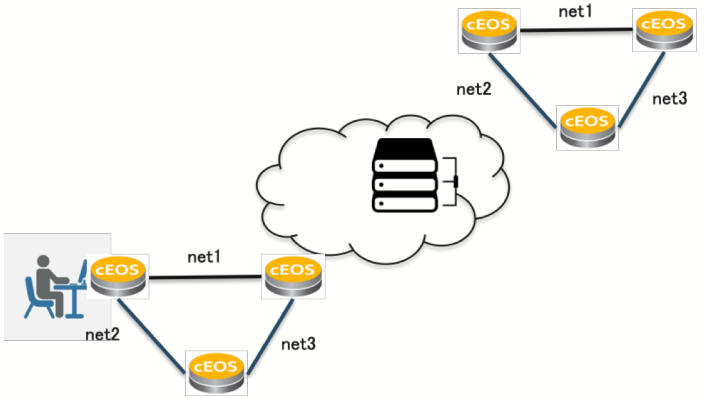
```
arista@arista:~$ docker exec -it ceos1 Cli -c "show ip route"

VRF: default
Codes: C - connected, S - static, K - kernel,
       O - OSPF, IA - OSPF inter area, E1 - OSPF external type 1,
       E2 - OSPF external type 2, N1 - OSPF NSSA external type 1,
       N2 - OSPF NSSA external type2, B I - iBGP, B E - eBGP,
       R - RIP, I L1 - IS-IS level 1, I L2 - IS-IS level 2,
       O3 - OSPFv3, A B - BGP Aggregate, A O - OSPF Summary,
       NG - Nexthop Group Static Route, V - VXLAN Control Service,
       DH - DHCP client installed default route, M - Martian,
       DP - Dynamic Policy Route, L - VRF Leaked

Gateway of last resort is not set

 C       10.10.1.0/30 is directly connected, Ethernet1
 C       10.10.1.4/30 is directly connected, Ethernet2
 O       10.10.1.8/30 [110/20] via 10.10.1.2, Ethernet1
                               via 10.10.1.6, Ethernet2
 C       10.255.255.1/32 is directly connected, Loopback0
 O       10.255.255.2/32 [110/20] via 10.10.1.2, Ethernet1
 O       10.255.255.3/32 [110/20] via 10.10.1.6, Ethernet2
```



cEOS-Lab

イメージをimportする

```
docker import cEOS-lab.tar.tar ceosimage:latest
```

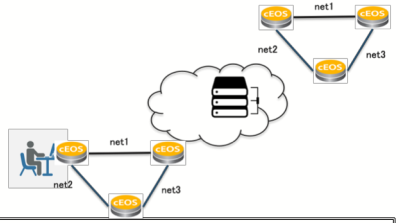


イメージを確認

```
shtsuchi:software shtsuchi$ docker image ls
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
ceosimage           latest              9a75bca440a7        12 hours ago        1.54GB
hello-world         latest              fce289e99eb9        2 weeks ago         1.84kB
busybox             latest              3a093384ac30        2 weeks ago         1.2MB
```


ARISTA

cEOS-Lab



コンテナ作成

```
docker create --name=ceos1 --privileged -p 5441:5441 -e CEOS=1 -e container=docker -e EOS_PLATFORM=ceoslab -e
SKIP_ZEROTOUCH_BARRIER_IN_SYSDBINIT=1 -e ETBA=1 -e INTFTYPE=eth -i -t ceosimage:latest /sbin/init
docker create --name=ceos2 --privileged -p 5442:5442 -e CEOS=1 -e container=docker -e EOS_PLATFORM=ceoslab -e
SKIP_ZEROTOUCH_BARRIER_IN_SYSDBINIT=1 -e ETBA=1 -e INTFTYPE=eth -i -t ceosimage:latest /sbin/init
docker create --name=ceos3 --privileged -p 5443:5443 -e CEOS=1 -e container=docker -e EOS_PLATFORM=ceoslab -e
SKIP_ZEROTOUCH_BARRIER_IN_SYSDBINIT=1 -e ETBA=1 -e INTFTYPE=eth -i -t ceosimage:latest /sbin/init
```

ネットワーク作成

```
docker network create net1
docker network create net2
docker network create net3
```

ネットワークとコンテナを接続する

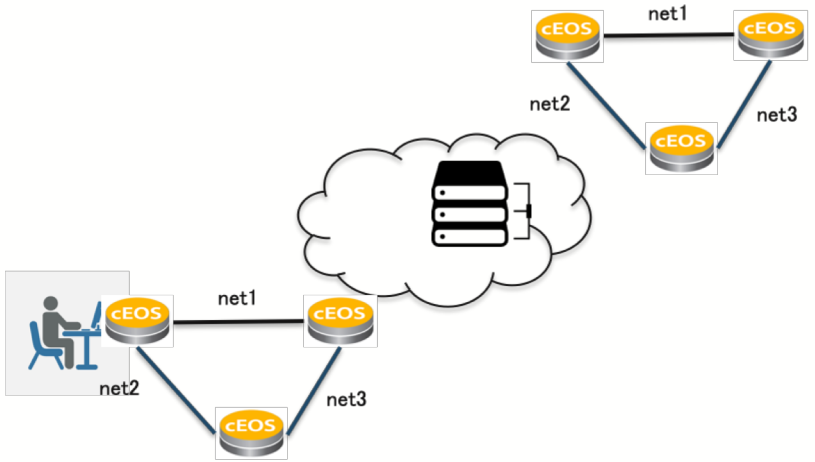
```
docker network connect net1 ceos1
docker network connect net1 ceos2
docker network connect net2 ceos1
docker network connect net2 ceos3
docker network connect net3 ceos2
docker network connect net3 ceos3
```


ARISTA

cEOS-Lab

コンテナ起動

```
docker start ceos1 ceos2 ceos3
```



起動してるコンテナを確認する

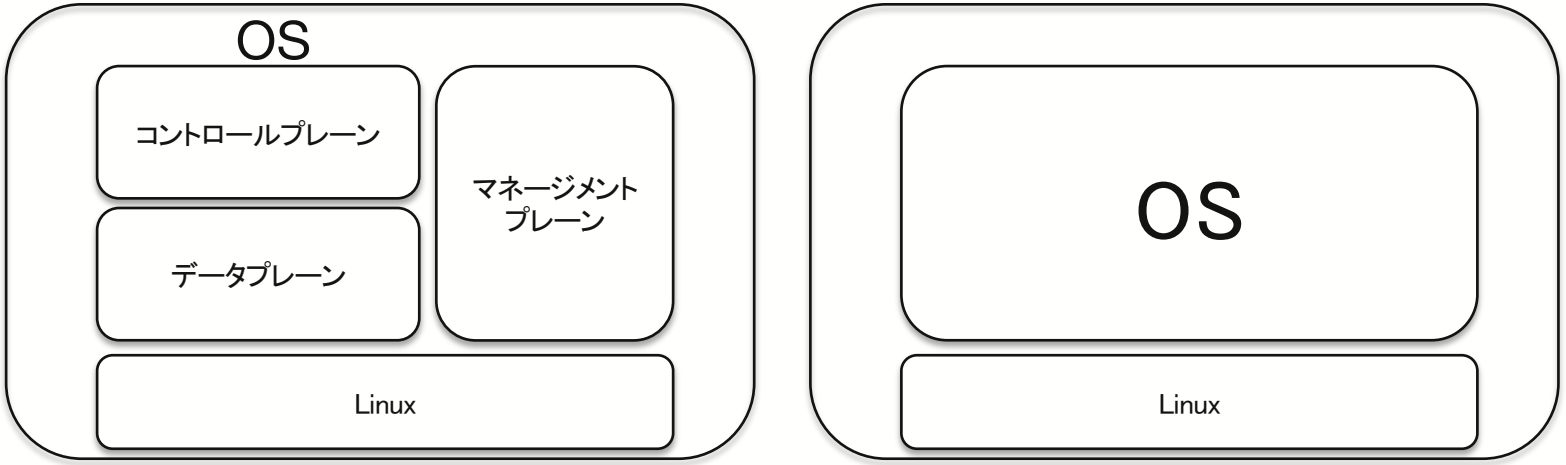
```
shtsuchi:software shtsuchi$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS                    NAMES
fb4e265b97db        ceosimage:latest    "/sbin/init"        29 minutes ago      Up 26 minutes       0.0.0.0:5443->5443/tcp   ceos3
b3e7ebd1f1bf        ceosimage:latest    "/sbin/init"        29 minutes ago      Up 26 minutes       0.0.0.0:5442->5442/tcp   ceos2
4ef23bad23a1        ceosimage:latest    "/sbin/init"        29 minutes ago      Up 26 minutes       0.0.0.0:5441->5441/tcp   ceos1
shtsuchi:software shtsuchi$
```


ARISTA

Agenda

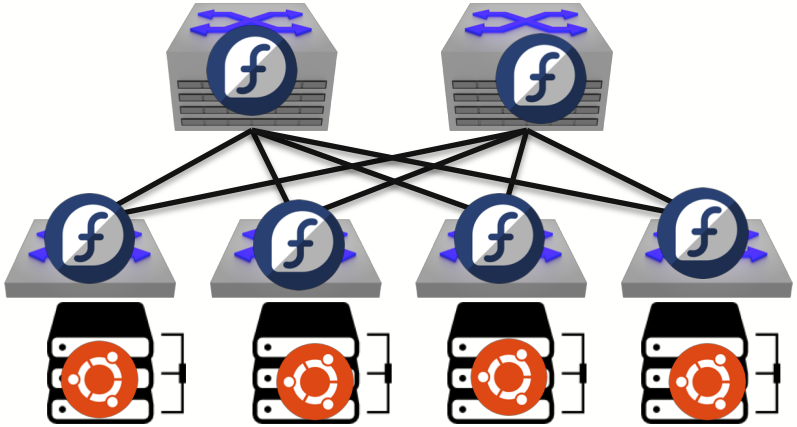
- コンテナとネットワーク
- ネットワーク機器上のコンテナ
- ネットワーク検証の為のコンテナ
- ネットワーク装置のインフラとしてのコンテナ

OSインフラとしてコンテナ利用



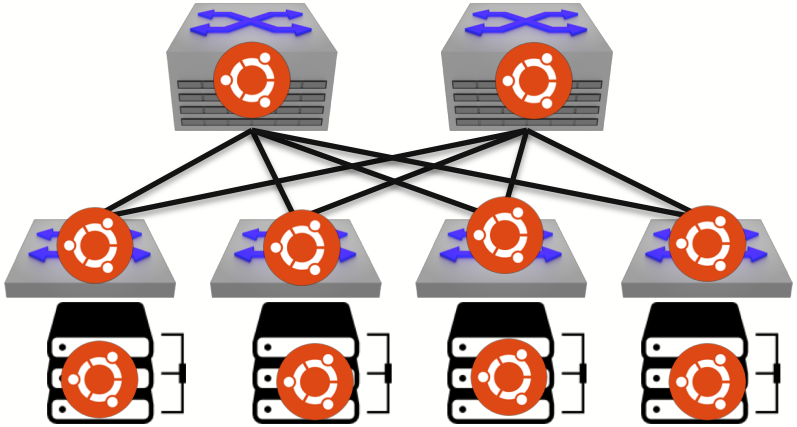
- コントロールプレーン/マネージメントプレーン/データプレーンをコンテナ化して、堅牢性を高める
- OS自体をコンテナ化し、柔軟性を高める

サーバーOSとネットワークOS



- 通常ネットワーク機器は提供者の用意したLinuxディストリビューションを使用
- サーバーは事業者が選定したLinuxディストリビューションを使用
- セキュリティ対策や方針などが統一はできない

サーバーOSとネットワークOS



- ネットワークOSをコンテナ化するとLinuxホストを選ばない

議論したい事

- どれくらいコンテナ化されてますか？
- ネットワークとコンテナの関係で実際の運用で困った事はありますか？
- ルータ/スイッチで乗せられると嬉しいアプリケーションってなんだろう？
- ネットワークOSがコンテナ化される事によるメリットや運用上の不安事ありますか？