K8s 網路模型與 CNI

Group 2

Dragon、Oakley、Raymond、耀中

大綱

- K8s 網路基本原則
- 各情境下, Pod 的溝通方式
- CNI 的介紹
- 各式常見 CNI 特性與優缺點比較
- Lab 數據
- Live Demo tcpdump Calico Packages
- 總結
- Q&A
- 參考資料

K8s 網路基本原則

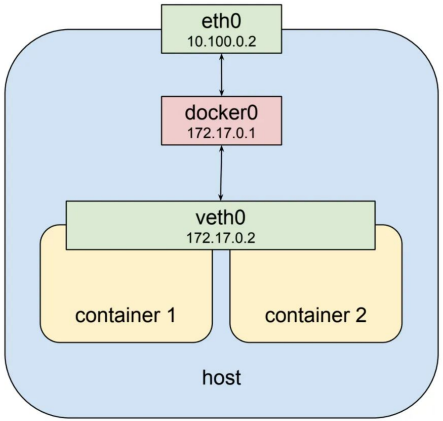
K8s 網路基本原則

- 在同一個 Cluster 內，**每一個 Pod 都有自己獨立且不重複的 IP**
  - 基於 **IP-per-Pod** 的方式
  - 不用考慮 Pod 之間額外溝通的方式
  - 不用處理容器端口暴露到主機端口的問題 → 跨 Node 溝通時
- Pod 之間溝通是不使用 NAT (Network Address Translation)
  - Pod 之間使用對方 IP 就可以直接找到，即使跨 Node 也是

各情境下，Pod 的溝通方式

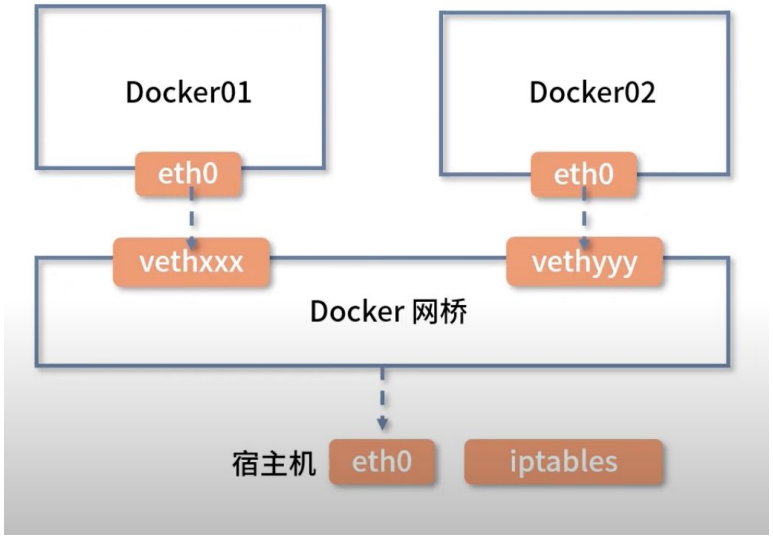
同一 Pod 內的 Container

- K8s 會替每一個 Pod 做一個獨立的 Netns (網路的 namespace)
  - 這個 Namespace 跟 K8s 的 namespace 不同，是 Linux 層的
  - 可以把每一個 Pod 當做一台獨立的主機
  - **Pod 內的所有 Container 共享同一個網卡 (veth0)、路由表、IPtable 等網路資源**
  - 每一個 Container 就是一個 Process
  - Container 之間溝通就像 Local 主機溝通一樣，用 Localhost



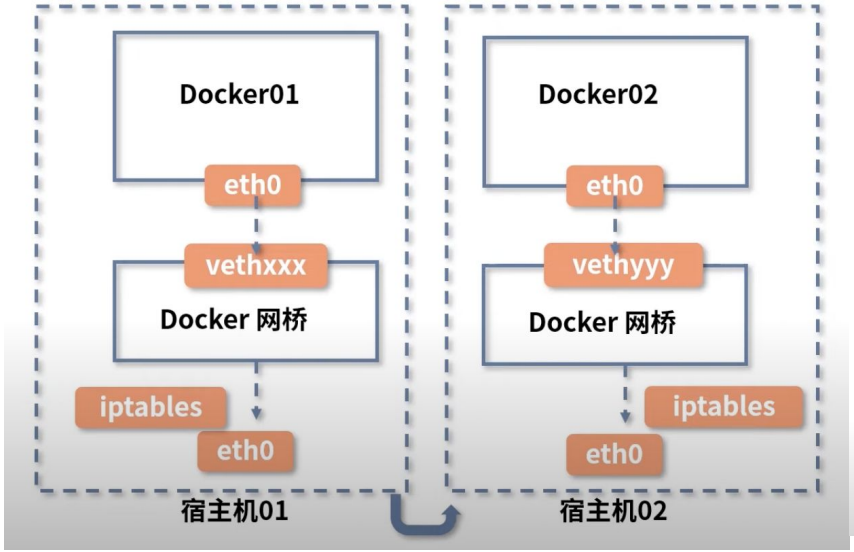
同一個 Node 內的 Pod

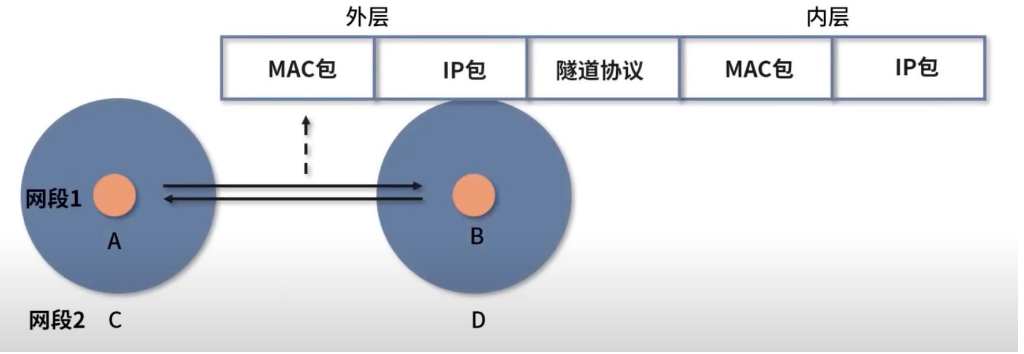
- 在同一個 Node 間的 Pod 是通過虛擬網卡 veth 來溝通
  - **所以每建立一個 Pod, 就會多一張 veth**
  - 這些虛擬網卡就會透過 cni0 (有些會寫 docker0 或是 cbr0) 網卡進行橋接



不同 Node 內的 Pod (透過 Overlay 的方式)

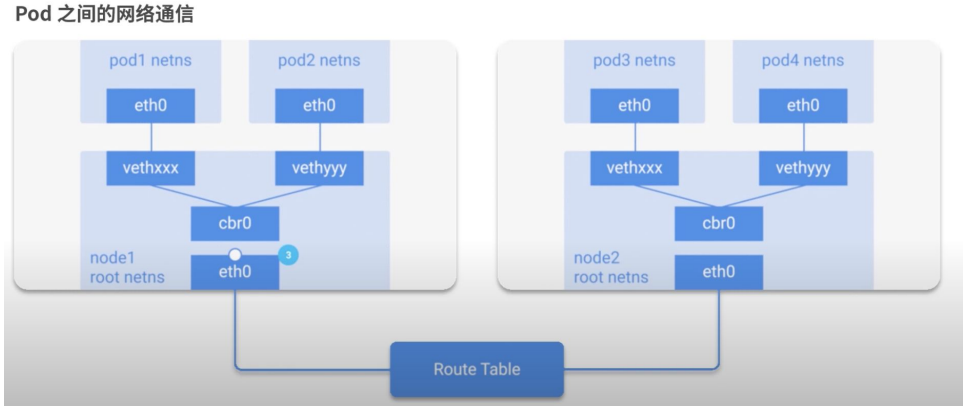
- 實作比較簡單，建立 Pod 的速度比較快，但網路傳輸效能損耗大

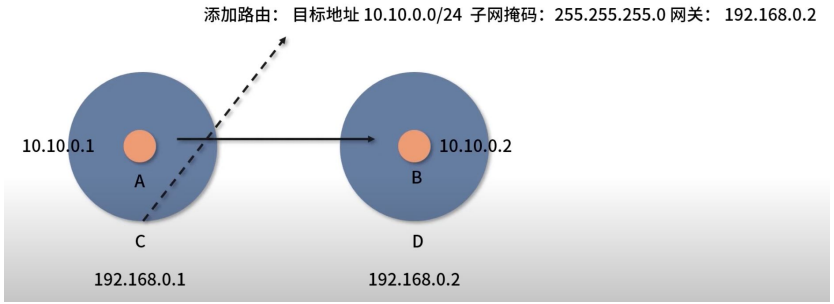




不同 Node 內的 Pod (透過 Routing 的方式)

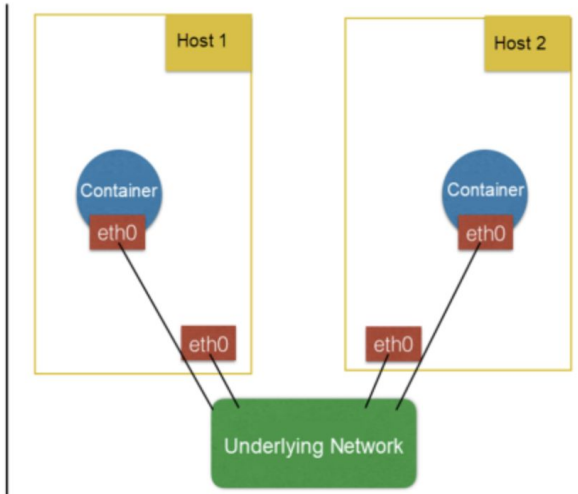
- 需要實作動態更新 Route Table 的方式，但網路傳輸效率比較好





不同 Node 內的 Pod (透過 Underlay 的方式)

- **Pod 和主機共用同一層網路資源** → 對於底層實作要求比較高
- 建立 Pod 的速度會比較慢，但網路傳輸效能較好
- AWS VPC CNI 的實作方式

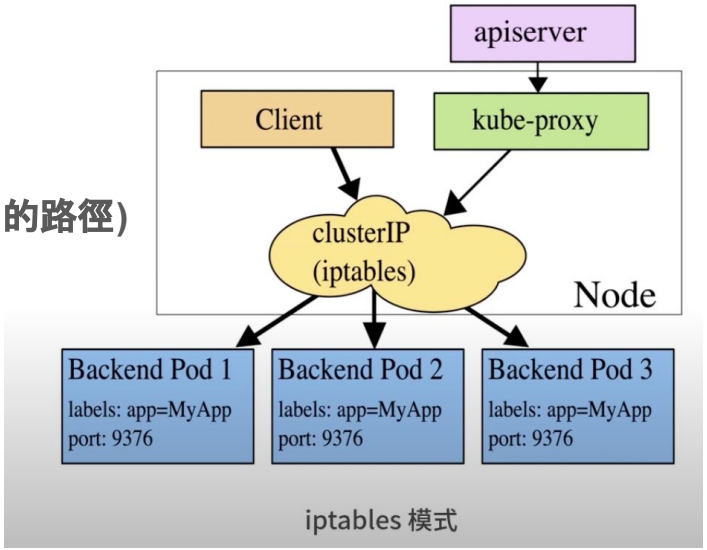


Underlay

Connections are established by using the underlying network capabilities and strongly depend on the underlying network.

Pod 到 Service 的處理

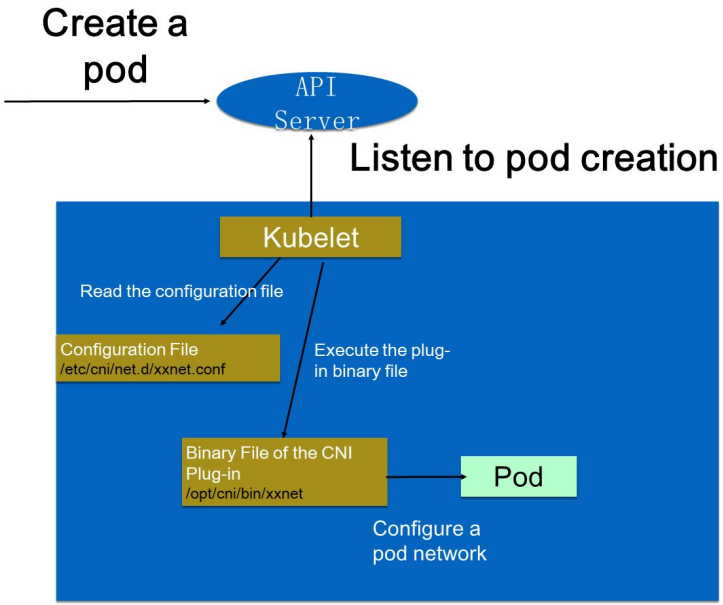
- kube-proxy 會負責動態處理 Node 內的 IPtable
  - 刪除 Service 時
    - Kube-proxy 會更新 Node 的 IPtable
  - 如果 Service 沒有使用 ClusterIP 的 Type 的話
    - Kube-proxy 不會做事
  - **建立 Service 時**
    - **Kube-proxy 讀取 Endpoint (Service 要連到 Pod 的路徑)**
    - **更新 Node 的 IPtable**



CNI 介紹

CNI 是什麼？

- 爲了因爲 K8s 內的網路對應不同情境會有不同實作的方式
  - 在 K8s 1.1 版後，將 CNI 改爲插件，讓使用者可以根據自己的需求去構建網路系統
  - CNI 定義網路接口和協議，負責處理 K8s 內網路資源的增刪
    - 創建 Pod 時，分配 IP 給 Pod
    - 建立 Pod 之間的溝通規則
    - 刪除 Pod 時，釋放被分配 IP



各式常見 CNI 特性與優缺點比較

Flannel

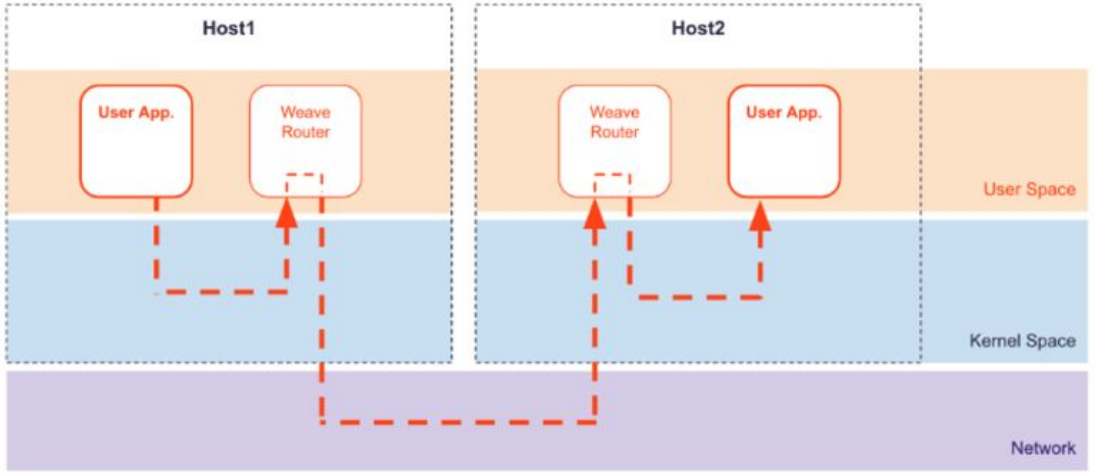
- CoreOS linux operating system
- **layer 3 overlay**
- Support VXLAN (default), UDP, Host-GW, AWS-VPC, GCE, Ali-VPC
- Easy to use
- No support for network policies.
- **Limited support for multi-cluster environments**.
- Support for IPsec encryption.
- **Suitable for small-scale or initial Kubernetes deployments without the need for network policies.**

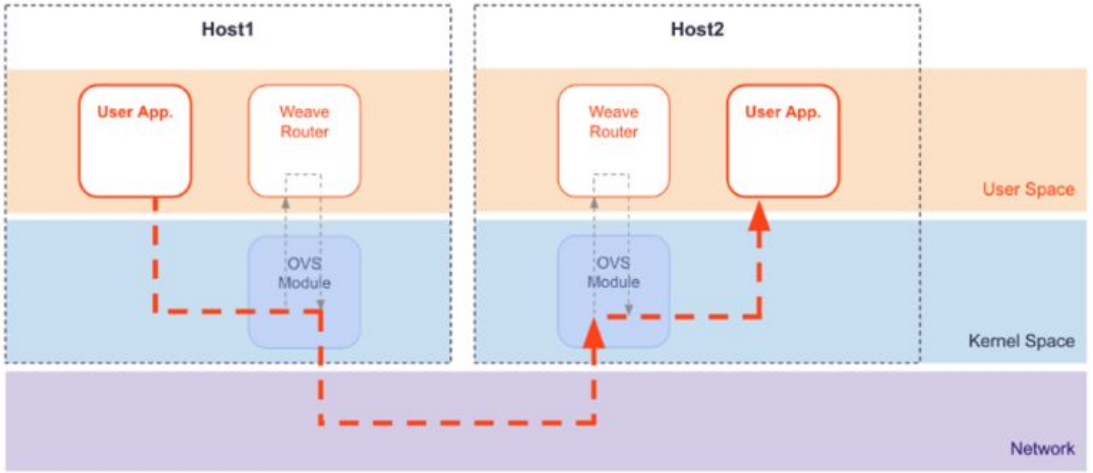
Calico

- **layer 3, layer 4 networking solution**
- **Support overlay & non-overlay network model**
- Support VXLAN (IPv6 is only supported for kernel versions ≥ 4.19.1 or redhat kernel version ≥ 4.18.0), IP in IP (only support IPv4) (for cross-subnet)
- Built-in data encryption
- **Multiple dataplanes: iptable, eBPF, windows HNS or VPP**
- Advanced IPAM management
- **Support network policy**
- **Support SCTP**
- cli: calicoctl
- Uses Prometheus to monitor Calico component metrics.
- No support VRF
- **Suitable for high availability, high scalability, high performance with network policy**

Weave

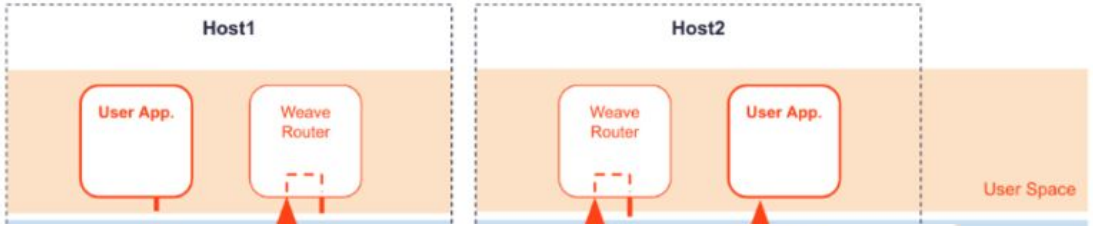
- layer 2 overlay
- Support UDP
- Support network policy
- Support network encryption
- Only support linux
- user space
- kernal space





Weave

- lay
- Su
- Su
- Su
- Or
- us
- ke



Alexis Richardson • 3 度 +
CEO Weaveworks
2 個月前 • 已編輯 •

+ 關注 ...

Hi everyone

I am very sad to announce – officially – that Weaveworks will be closing its doors and shutting down commercial operations. Customers and partners will be working with a financial trustee whom we shall announce soon.

+ 關注

very sad to announce – officially – that Weaveworks will be closi
loors and shutting down commercial operations. Customers and
ners will be working with a financial trustee whom we shall annou
n.

Cilium

- A newer CNI plugin
- **eBPF** technology
- **Supports multi-cluster**
- Includes load balancing capabilities (fully replace kube-proxy)
- **Network policies for layer 3, layer 4, and layer 7.**
- Supports metadata monitoring, such as package source, target IP, sender, receiver, and comprehensive tag information, etc.
- Support Metrics export via Prometheus
- **Hubble**: observability platform (cilium) provides service dependency maps, operational monitoring and alerting, and application and security visibility based on flow logs.
- **Suitable for scenarios requiring fine-grained security controls or reducing lookup latency in large k8s clusters.**

Feature	Calico	Flannel	Weave	Cilium
Networking model	Layer 3 (BGP) and IPIP overlay, host-gw	layer 2 (VXLAN, host-gw) overlay, direct routing	layer 2 (VXLAN, host-gw) overlay, direct routing	layer 3 (BPF) and Layer 4 (Socket)
Network policy	Yes, supports fine-grained network policies	Limited support network policies	Yes	Yes, with advanced BPF-based policies
Performance	High	Moderate	Moderate	High
Scalability	Highly scalable	Good	Good	Highly scalable
DNS support	Yes	Limited	Yes	Yes
Observability	Yes, with built-in monitoring and logs	Limited	Yes, with built-in monitoring and logs	Yes, with advanced observability
Security features	Advanced	Basic	Basic	Advanced
User case	wide range of use cases, including security-sensitive workloads	simple, small-scale deployments	sized deployments	high-performance environments

Lab 數據

比較項目

- 實驗環境
  - Local Kubernetes 環境
  - 雲端 Kubernetes 環境
- 實驗對象
  - Cilium
  - Calico
  - Flannel
  - AWS-Node
- 實驗情境
  - 同 Node 間的 Pod
  - 不同 Node 間的 Pod
- 實驗指標
  - Pod 建立時間 (從建立到 Ready 狀態所需時間)
  - Response time (使用 ab 工具模擬 10000 個請求)
  - CNI 資源使用量 (CPU 和記憶體使用量)

Local K8s

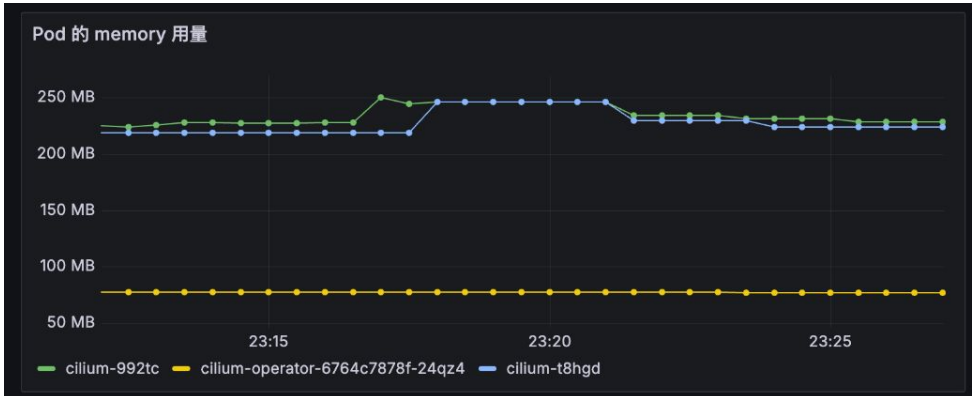
- 使用 Minikube 更換不同 CNI 進行測試

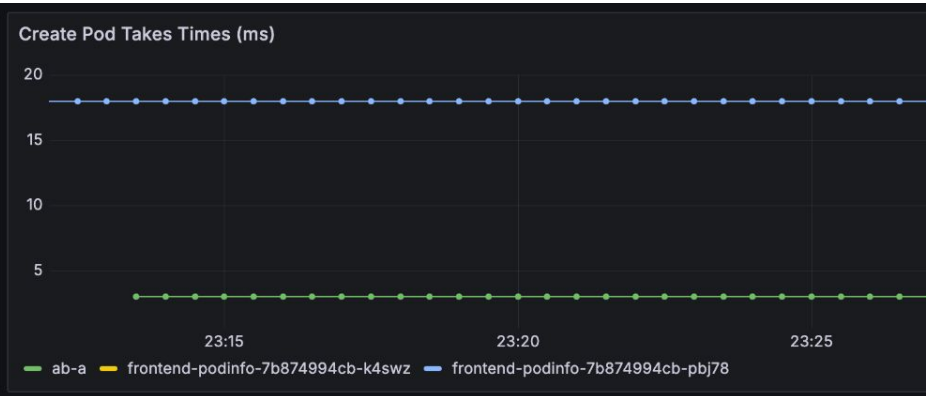
CNI	Networking Type	Pod 的建立所花費的時間 (ms)	回覆時間 (ms)		CNI 使用的運算資源	
			同 Node 間的 Pod	不同 Node 間的 Pod	CPU (Core/s)	Memory (MB)
Cilium	Overlay	63	**0.226**	**0.410**	0.015	200
Calico	Overlay	**54**	0.185	0.230	0.025	200
Flannel	Overlay	63	0.182	0.296	**0.009**	**16**

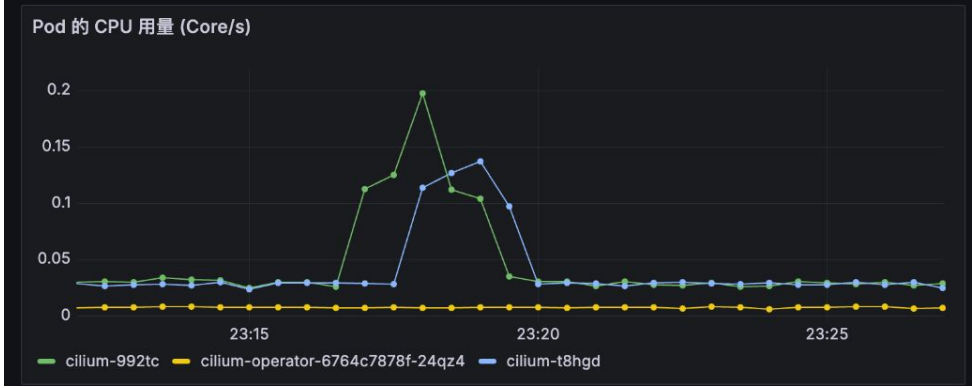
雲端 K8s

- 在 EKS 上更換不同 CNI 進行測試

CNI	Networking Type	Pod 建立所花費的時間 (ms)	回覆時間 (ms)		CNI 使用的運算資源	
			同 Node 間的 Pod	不同 Node 間的 Pod	CPU (Core/s)	Memory (MB)
Cilium	Overlay	18	**0.558**	**3.53**	0.2	250
AWS-Node	Underlay	**7**	2.46	15.8	**0.01**	**130**


Pod 的 memory 用量
250 MB
200 MB
150 MB
100 MB
50 MB
23:15
23:20
23:25
cilium-992tc
cilium-operator-6764c7878f-24qz4
cilium-t8hgd


Create Pod Takes Times (ms)
20
15
10
5
23:15
23:20
23:25
ab-a
frontend-podinfo-7b874994cb-k4swz
frontend-podinfo-7b874994cb-pbj78


Pod 的 CPU 用量 (Core/s)
0.2
0.15
0.1
0.05
0
23:15
23:20
23:25
cilium-992tc
cilium-operator-6764c7878f-24qz4
cilium-t8hgd

Cilium 同 Node

```
Server Software:
Server Hostname:        192.168.8.142
Server Port:            9898

Document Path:          /
Document Length:        401 bytes

Concurrency Level:      1
Time taken for tests:   5.582 seconds
Complete requests:      10000
Failed requests:        0
Total transferred:      5580000 bytes
HTML transferred:       4010000 bytes
Requests per second:    1791.48 [#/sec] (mean)
Time per request:       0.558 [ms] (mean)
Time per request:       0.558 [ms] (mean, across all concurrent requests)
Transfer rate:          976.22 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:        0    0   0.4      0      18
Processing:     0    0   0.4      0      14
Waiting:        0    0   0.4      0      14
Total:          0    0   0.6      0      18

Percentage of the requests served within a certain time (ms)
  50%      0
  66%      0
  75%      1
  80%      1
  90%      1
  95%      1
  98%      1
  99%      2
 100%     18 (longest request)
```

Cilium 不同 Node

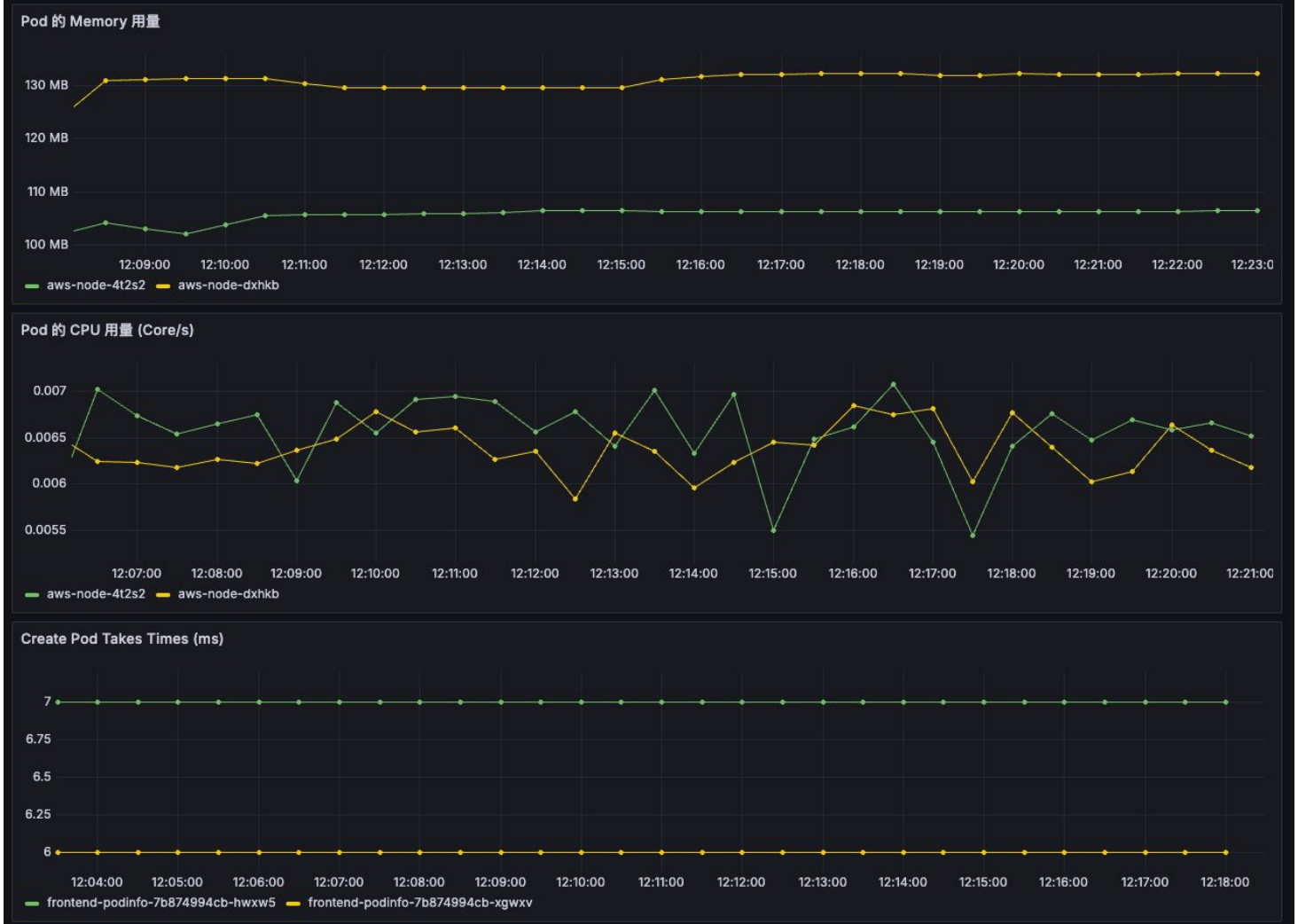
```
Server Port:            9898

Document Path:          /
Document Length:        401 bytes

Concurrency Level:      1
Time taken for tests:   35.395 seconds
Complete requests:      10000
Failed requests:        0
Total transferred:      5580000 bytes
HTML transferred:       4010000 bytes
Requests per second:    282.53 [#/sec] (mean)
Time per request:       3.539 [ms] (mean)
Time per request:       3.539 [ms] (mean, across all concurrent requests)
Transfer rate:          153.96 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:        1    2   0.8      1      18
Processing:     1    2   1.1      2      30
Waiting:        0    2   1.0      2      18
Total:          3    3   1.5      3      31
WARNING: The median and mean for the initial connection time are not within a normal deviation
        These results are probably not that reliable.

Percentage of the requests served within a certain time (ms)
  50%      3
  66%      3
  75%      3
  80%      4
  90%      5
  95%      6
  98%      8
  99%     10
 100%     31 (longest request)
```


Pod 的 Memory 用量
130 MB
120 MB
110 MB
100 MB
12:09:00
12:10:00
12:11:00
12:12:00
12:13:00
12:14:00
12:15:00
12:16:00
12:17:00
12:18:00
12:19:00
12:20:00
12:21:00
12:22:00
aws-node-4t2s2
aws-node-dxhkb
Pod 的 CPU 用量 (Core/s)
0.007
0.0065
0.006
0.0055
12:07:00
12:08:00
12:09:00
12:10:00
12:11:00
12:12:00
12:13:00
12:14:00
12:15:00
12:16:00
12:17:00
12:18:00
12:19:00
12:20:00
12:21:00
aws-node-4t2s2
aws-node-dxhkb
Create Pod Takes Times (ms)
7
6.75
6.5
6.25
6
12:04:00
12:05:00
12:06:00
12:07:00
12:08:00
12:09:00
12:10:00
12:11:00
12:12:00
12:13:00
12:14:00
12:15:00
12:16:00
12:17:00
12:18:00
frontend-podinfo-7b874994cb-hwxw5
frontend-podinfo-7b874994cb-xgwxv

AWS CNI 同 Node

```
# ab -n 10000 http://10.0.101.182/
This is ApacheBench, Version 2.3 <$Revision: 1879490 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking 10.0.101.182 (be patient)
apr_socket_recv: Connection refused (111)
# ab -n 10000 http://10.0.101.182/echo
This is ApacheBench, Version 2.3 <$Revision: 1879490 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking 10.0.101.182 (be patient)
apr_socket_recv: Connection refused (111)
# ab -n 10000 http://10.0.101.182:9898/echo
This is ApacheBench, Version 2.3 <$Revision: 1879490 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking 10.0.101.182 (be patient)
Completed 1000 requests
Completed 2000 requests
Completed 3000 requests
Completed 4000 requests
Completed 5000 requests
Completed 6000 requests
Completed 7000 requests
Completed 8000 requests
Completed 9000 requests
Completed 10000 requests
Finished 10000 requests


Server Software:
Server Hostname:        10.0.101.182
Server Port:            9898

Document Path:          /echo
Document Length:        168 bytes

Concurrency Level:      1
Time taken for tests:   24.556 seconds
Complete requests:      10000
Failed requests:        0
Total transferred:      3430000 bytes
HTML transferred:       1680000 bytes
Requests per second:    407.24 [#/sec] (mean)
Time per request:       2.456 [ms] (mean)
Time per request:       2.456 [ms] (mean, across all concurrent requests)
Transfer rate:          136.41 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:        0    0   0.1      0       4
Processing:     1    2   1.1      2      21
Waiting:        0    2   1.1      2      21
Total:          1    2   1.1      2      21

Percentage of the requests served within a certain time (ms)
  50%      2
  66%      2
  75%      2
  80%      3
  90%      3
  95%      4
  98%      6
  99%      8
 100%     21 (longest request)
```

AWS CNI 不同 Node

```
# ab -n 10000 http://10.0.102.102:9898/echo/
This is ApacheBench, Version 2.3 <$Revision: 1879490 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking 10.0.102.102 (be patient)
Completed 1000 requests
Completed 2000 requests
Completed 3000 requests
Completed 4000 requests
Completed 5000 requests
Completed 6000 requests
Completed 7000 requests
Completed 8000 requests
Completed 9000 requests
Completed 10000 requests
Finished 10000 requests


Server Software:
Server Hostname:        10.0.102.102
Server Port:            9898

Document Path:          /echo/
Document Length:        168 bytes

Concurrency Level:      1
Time taken for tests:   158.011 seconds
Complete requests:      10000
Failed requests:        0
Total transferred:      3430000 bytes
HTML transferred:       1680000 bytes
Requests per second:    63.29 [#/sec] (mean)
Time per request:       15.801 [ms] (mean)
Time per request:       15.801 [ms] (mean, across all concurrent requests)
Transfer rate:          21.20 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:        2    2   0.6      2      10
Processing:    11   14   2.7     13      57
Waiting:       10   14   2.6     13      57
Total:         13   16   2.8     15      59

Percentage of the requests served within a certain time (ms)
  50%     15
  66%     15
  75%     16
  80%     17
  90%     19
  95%     21
  98%     24
  99%     27
 100%     59 (longest request)
```

Live Demo tcpdump Calico Packages

Step by Step

總結

總結

- 簡單不複雜的架構 -> Flannel
- 使用 AWS -> AWS VPC CNI
- 複雜大型架構與設定詳細的 network policy -> calico, cilium

Q&A

Q&A

- 為何 EKS 會有 IP 不夠的情況？
- 別的 CNI 會不會有呢？
- 搭配使用 ALB/NLB 時，有什麼要注意的地方呢？

- 要如何驗證 docker in docker 跟真實的 K8s 的行爲一致？
- Docker compose 的網路處理方式？

参考資料

參考資料

- Getting Started with Kubernetes | Kubernetes CNIs and CNI Plug-ins
- Understanding kubernetes networking: pods
- 影片
  - 26 网络插件：Kubernetes 搞定网络原来可以如此简单？
  - k8s 主流网络方案（OVS、Flannel、Calico）及原理
  -