FUJITSU
shaping tomorrow with you

Kubernetesとエコシステム -2017.07-

Linux開発統括部
富士通株式会社
kamezawa.hiroyu@jp.fujitsu.com

今日の内容

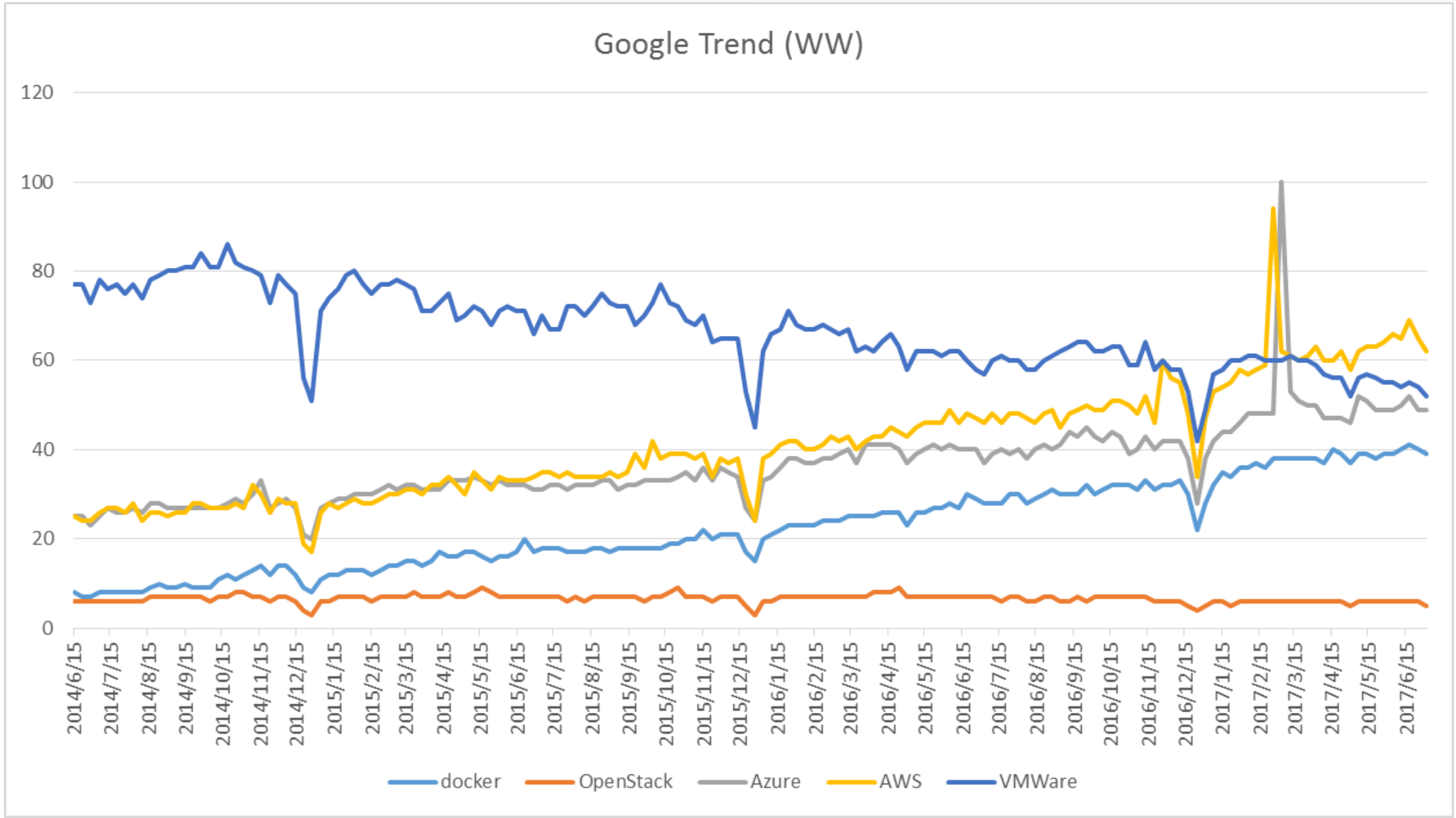
FUJITSU

- Kubernetesはわかったけど、どう成長しているのか、とか、そのまわりのOSSとか、どういうのがあるの？を駆け足でやりながら、３カ月に一度リリースされているKubernetesのここ最近の機能について駆け足で紹介します。
- じっくりやりませんけども、興味がわいたものは後から私を捕まえるか、ググってもらうかしてください。

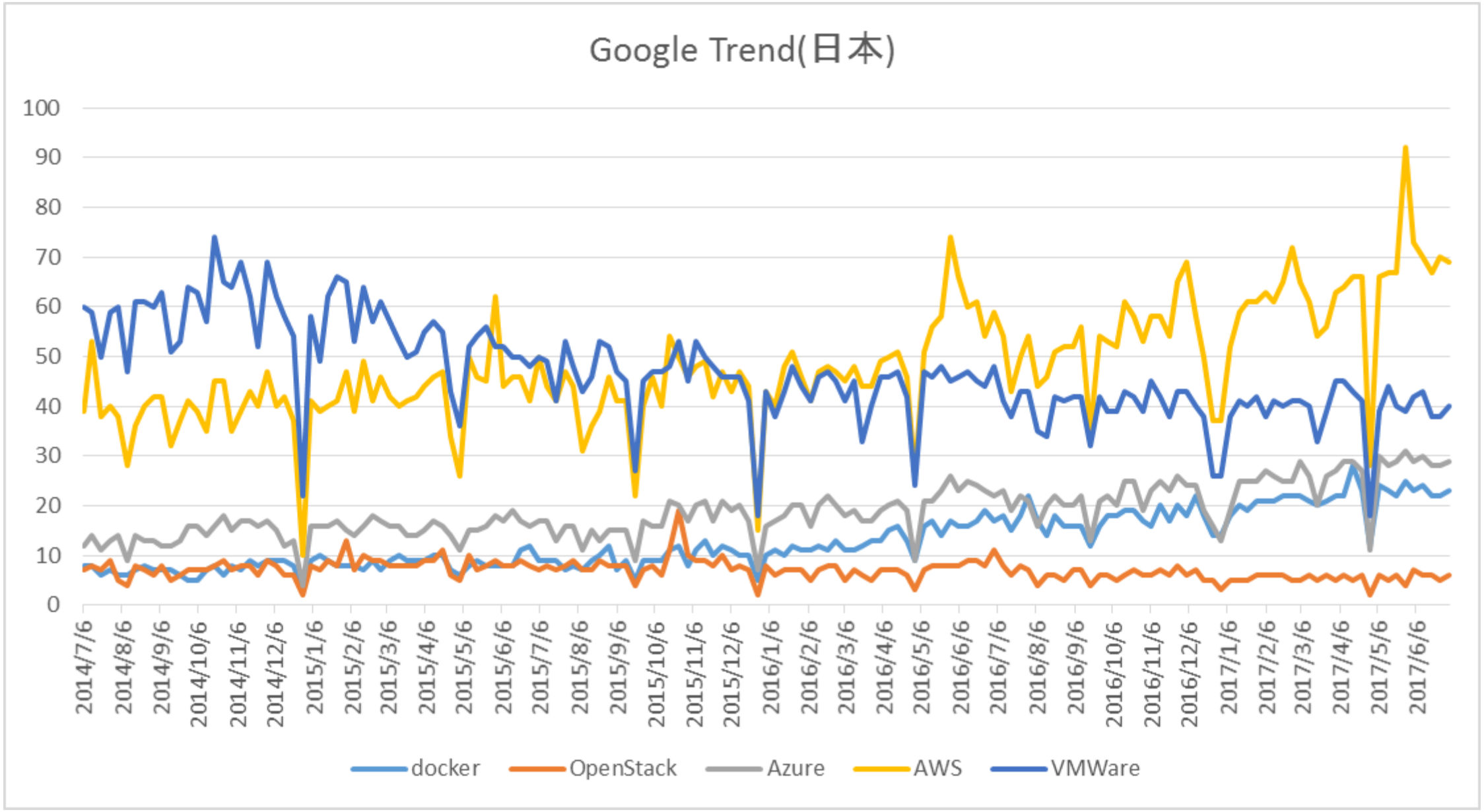
FUJITSU

コンテナの勢いとDocker

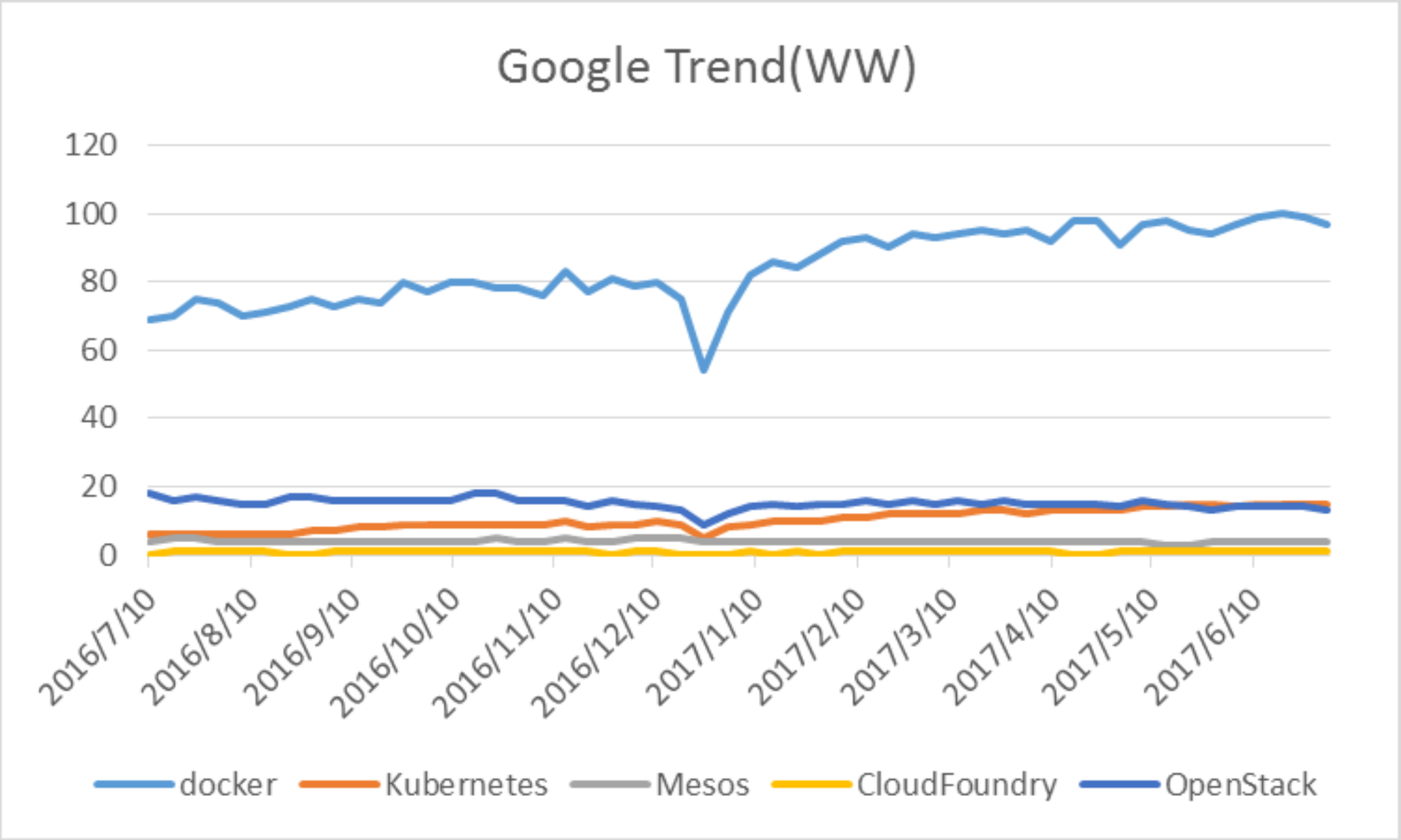
仮想化技術の勢い(世界)



仮想化技術の勢い(日本)

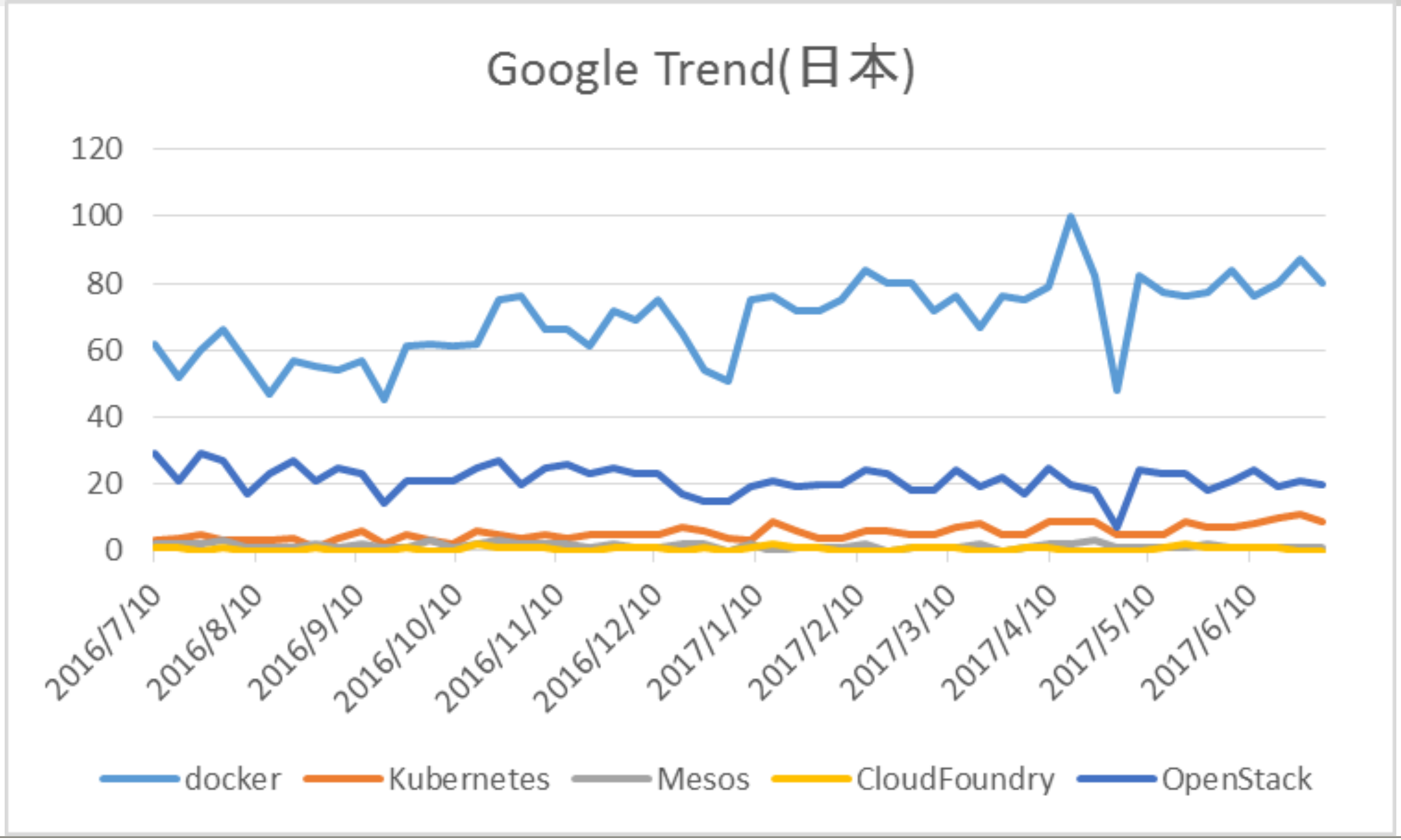


コンテナ技術の勢い(WW)



コンテナ技術の勢い(日本)

FUJITSU



KubeCon 2017 Europe

FUJITSU

- 2017年3月29-30日、ベルリン
  - Cloud Native Computing Foundation(CNCF)イベント
  - 開催地：ベルリンコングレスセンタ
  - 来場者：**1500名超**
    - キーノートも立ち見・座り見で非常な熱気
    - スポンサー43社： プラチナ： Red Hat, Huawei, CoreOS, Intel, Tigera
  - 概観
    - CNCFの傘下プロジェクトが増加し、技術分野が網羅的に
      - オーケストレーション・性能監視・ログ・トレース・サービス管理・サービス接続・コンテナ管理
      - 今後、ネットワーク・ストレージ関連のプロジェクト追加を予定
    - （ヨーロッパだからか）スーツ組がほとんどおらず、開発者が中心
      - 出展ブースも Huawei, Red Hat, VMWare, CA 等に加え、スタートアップが多い印象
    - 日本からは Yahoo, NTTから参加者（Yahoo, NTTとは7月のOpenstack Days Tokyoでコンテナセッション企画)
    - 12月の北米イベントは**3000人規模で開催予定**

標準化の動き

FUJITSU

- Open Container Initiative
  - コンテナ標準フォーマットを定義する非営利団体
    - Docker 単独支配にストップをかける業界からの一撃
  - **コンテナ標準規格のバージョン1.0（昨日リリースされました！）**
  - 3Qにコンテナ認証(イメージとRuntimeに規格準拠の印をつける）を提供予定
- Kubernetes Certified Engineer
  - Kubernetesの認定エンジニア制度。オンラインテストを受験する。βテスト中
- Kubernetes Confirmation
  - まだ始まったばかりだが、KuberentesのAPI互換保証/認定を作る仕組みの協議開始

FUJITSU

Kubernetesと
Cloud Native Computing Foundation

Cloud Native Computing Foundation

FUJITSU

Orchestration

Monitoring

Tracing

Logging

Service Mesh

Remote Procedure Call

Networking

Service Discovery

Container Runtime

Container Runtime

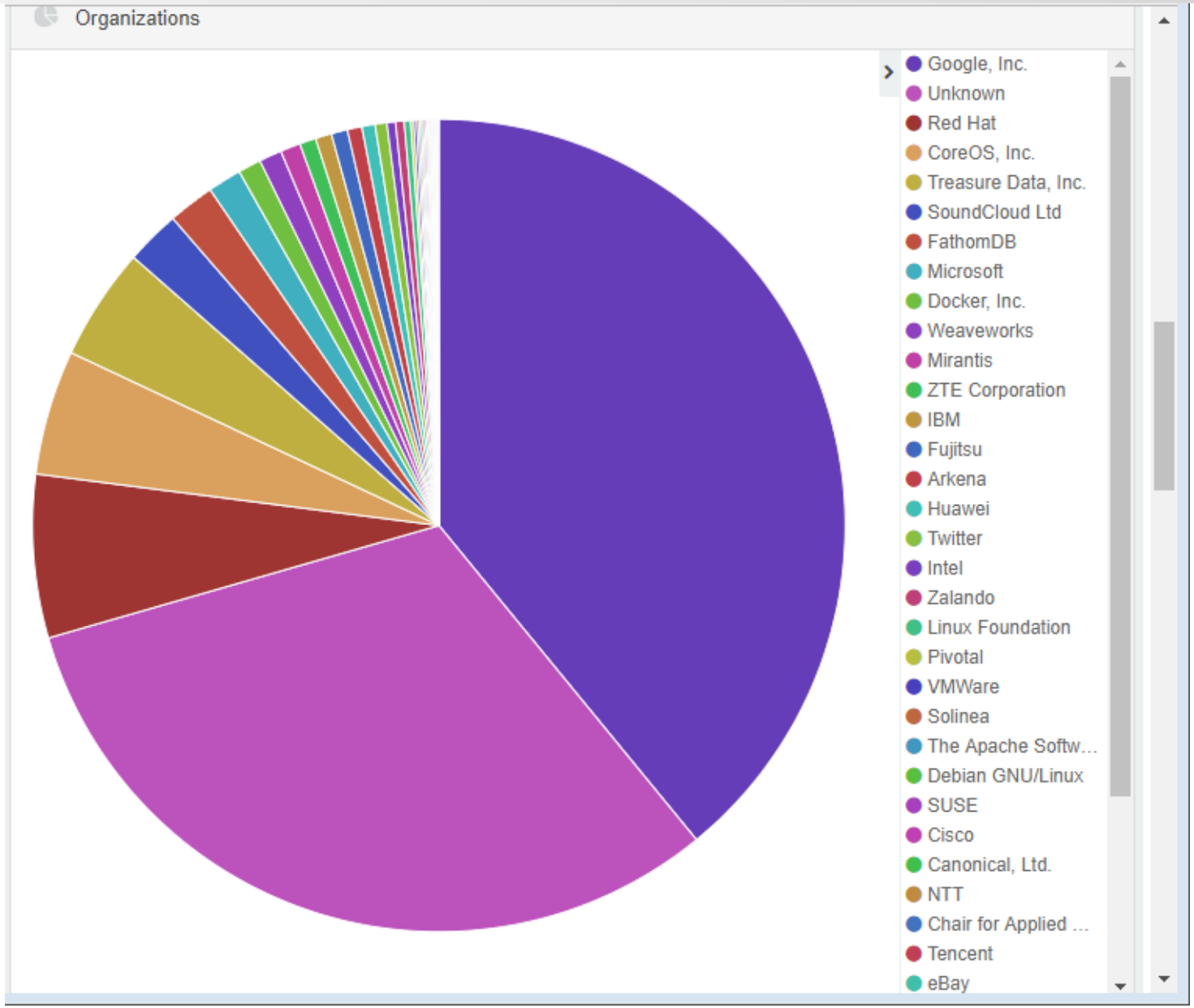
- 参加企業は80社以上
  - Platinum Member(ボード)
    - Cisco, CoreOS, Dell, Docker, Fujitsu, Google, Huawei, IBM, Intel, Joyent, Mesosphere, Red Hat, Samsung SDS, SuperNAP
- ホストするプロジェクト数は１０に増加
  - 今後はストレージ関連プロジェクト、メトリクスＡＰＩの共通化などが俎上に
  - 規模的にはKuberentesが巨大プロジェクトで他は小さめ

CNCF のプロジェクト規模①

FUJITSU

Project	Commits	Authors	Repositories
kubernetes	**83982**	**2461**	**58**
grpc	21849	366	12
prometheus	8479	547	35
Fluentd	7672	212	31
rkt	3566	142	2
linkerd	2746	64	8
opentracing	2532	102	35
containerd	2265	107	7
cncf	2002	67	19
CoreDNS	1005	44	8
CNI	637	64	2

CNCFのプロジェクト規模②



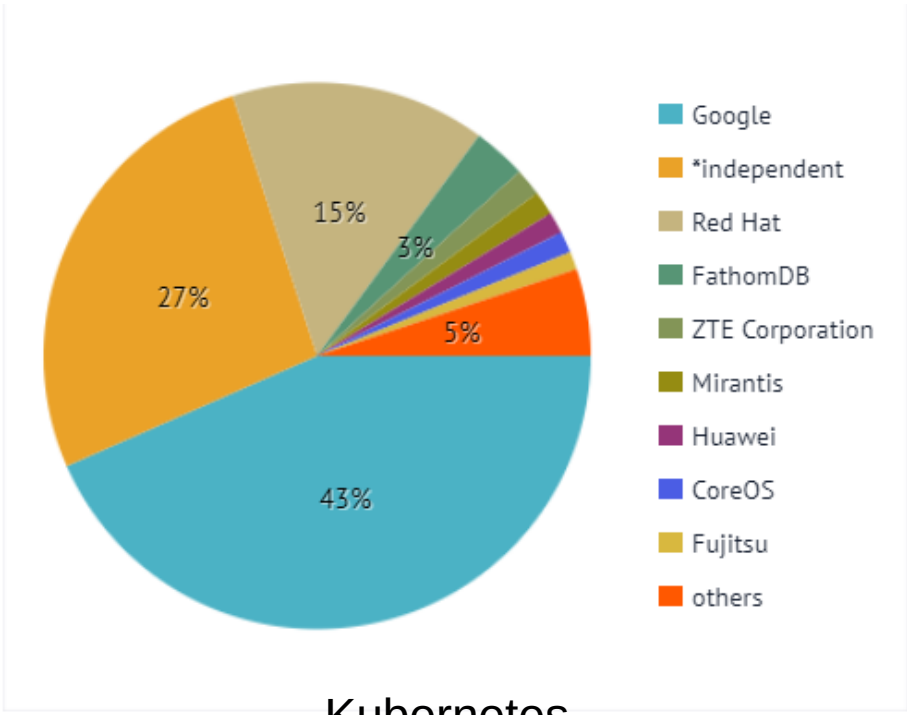
CNCFの活動

FUJITSU

- ボード会議と TOC(Technical Oversite Committee)そのサブ会議で運営
  - ボード会議：投資判断や施策、メッセージングについての決定・合意
    - マーケティングや認定試験等のサブ会議を持つ
    - イベント運営、SNSを使ったマーケティング、記事の公開
  - TOC：プロジェクト間の課題の解決、新規プロジェクトの採用判断
    - 新規のプロジェクト採用の仕組みと権限
    - KubernetesのAPI標準化やメトリクスAPIの標準化
    - CI環境の整備・整理
- 運営には、Google, Docker, CoreOS, Red Hat, IBMなど積極的

Kubernetes

FUJITSU



Kubernetes
コミッターの所属分布

- コンテナオーケストレーションのスタンダード
- 以下の機能を持つ
  - あらゆるワークロードへの対応
    - ステートレス、デーモン、バッチ、ステートフル、定期
  - リソーススケジュール
  - ライフサイクルマネジメント
  - サービスとコンテナの紐付け
    - DNS
    - Load Balancing
  - ネットワーク(plugin)
  - ストレージ管理(plugin)
  - 認証・認可・監査
  - ログ管理….等

Kubernetesを利用する

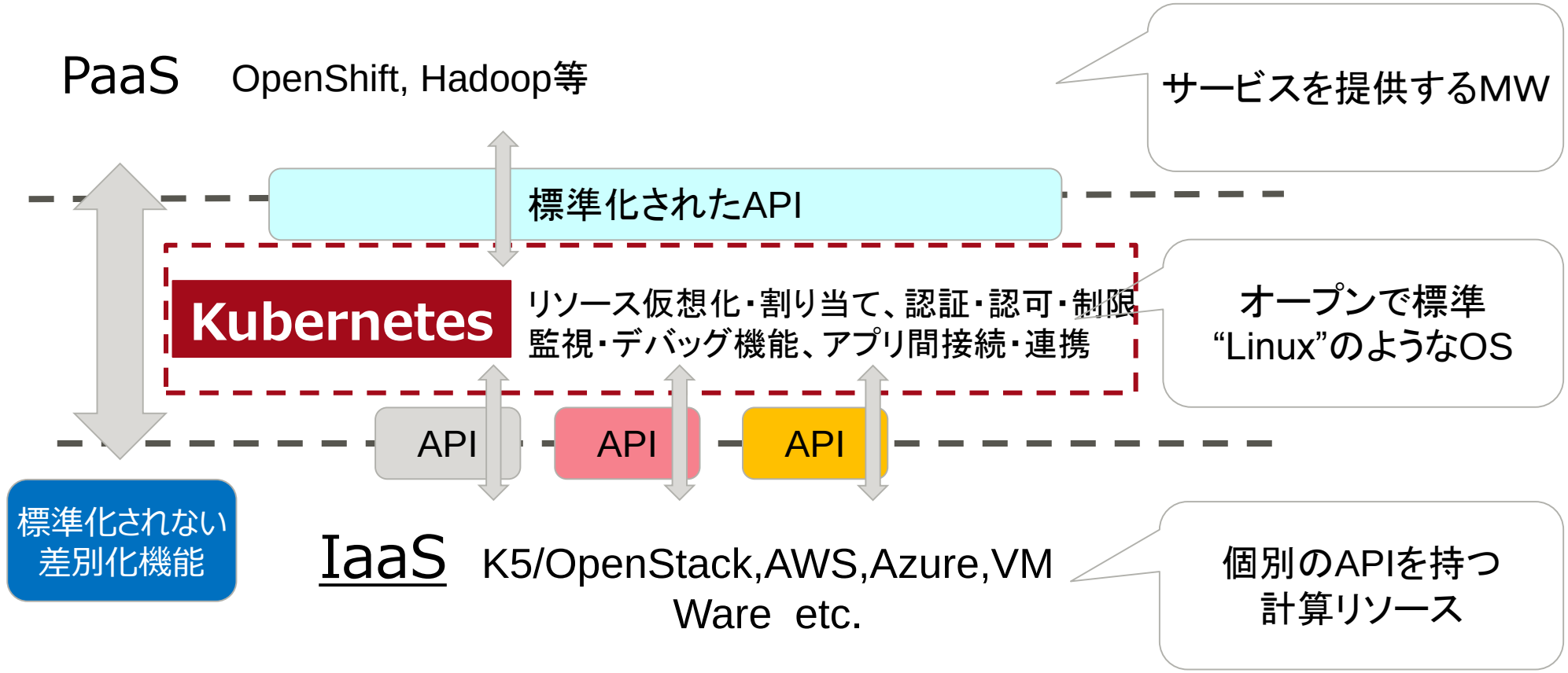
FUJITSU

- 以下のプロダクト・サービスでKubernetesをサポート
  - Google Compute Engine
  - Google Container Engine
  - AWS EC2
  - Azure (Azure Container Service)
  - IBM Blue Mix
  - Red Hat OpenShift
  - SUSE Container as a Service Platform
  - The Canonical Distribution of Kubernetes
  - Tectonic by CoreOS
  - RancherOS/Kubernetes
  - VMWare Photon
  - Mirantis

Etc…..

位置づけ

クラウドMWとクラウドインフラ(IaaS)を繋ぐオープンな共通クラウドOS, Kubernetes



Kubernetes周辺で起きているコト

- KubernetesはアプリケーションとクラウドのAPIを定義
  - クラウドインフラはKubernetesのイベントに連動

- ミドルウェアはKubernetes上で動作
  - OpenStackをKubernetes上で
  - CloudFoundryをKubernetes上で
  - ビッグデータ、AI、DBをKubernetesで

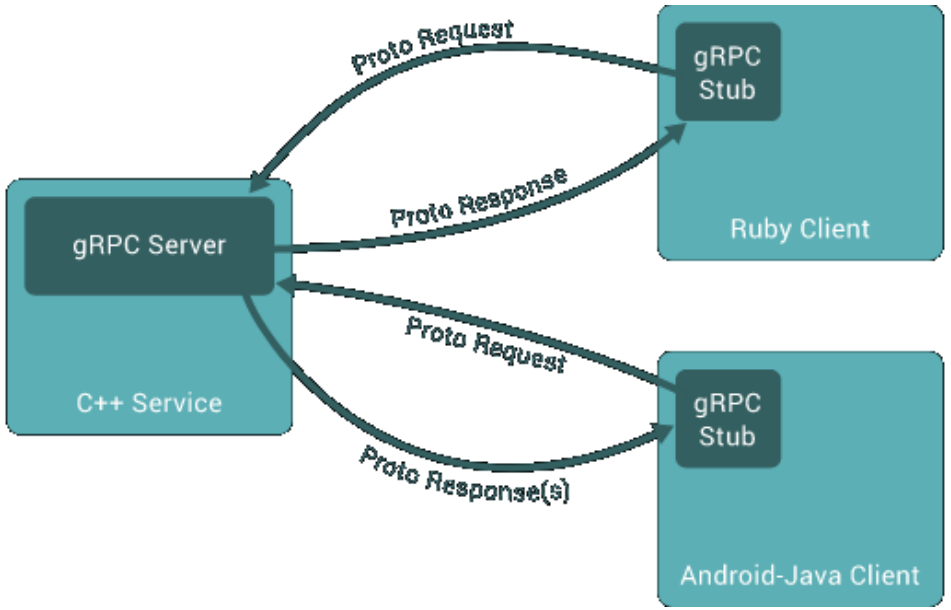
CNCFのプロジェクト pick up①

https://prometheus.io/

- Prometheus
  - モニタリングソフトウェア
    - ハードやＶＭ監視特化ではなく、任意のサービスを監視
    - Prometheusに対応した統計情報APIからデータ収集
  - 時系列データベース、クエリ、可視化
  - アラート
  - アプリやノードの情報をメトリクス情報API(exporter)から収集
    - KubernetesやetcdなどはPrometheusAPIを標準装備
  - Kubernetesのイベントと同期した情報収集(Beta)
- CNCFでは OpenMetrics としてのAPI標準化を提案中

CNCFのプロジェクト pick up ②

https://grpc.io



- GRPC
  - http/2 ベースのRPCフレームワーク、Protocol Buffersで値をやりとりする
  - 同期/非同期モデルを両方をサポート
  - 多くのプロジェクトで使われ始めている
  - 機能が拡張されてきており、Load Balancing のフレームワークなども組み込まれてきている
    - 簡単な Client Side Load Balancing + 頭の良い外部ロードバランサー
    - https://github.com/grpc/grpc/blob/master/doc/load-balancing.md

CNCFのプロジェクト pick up ③

FUJITSU

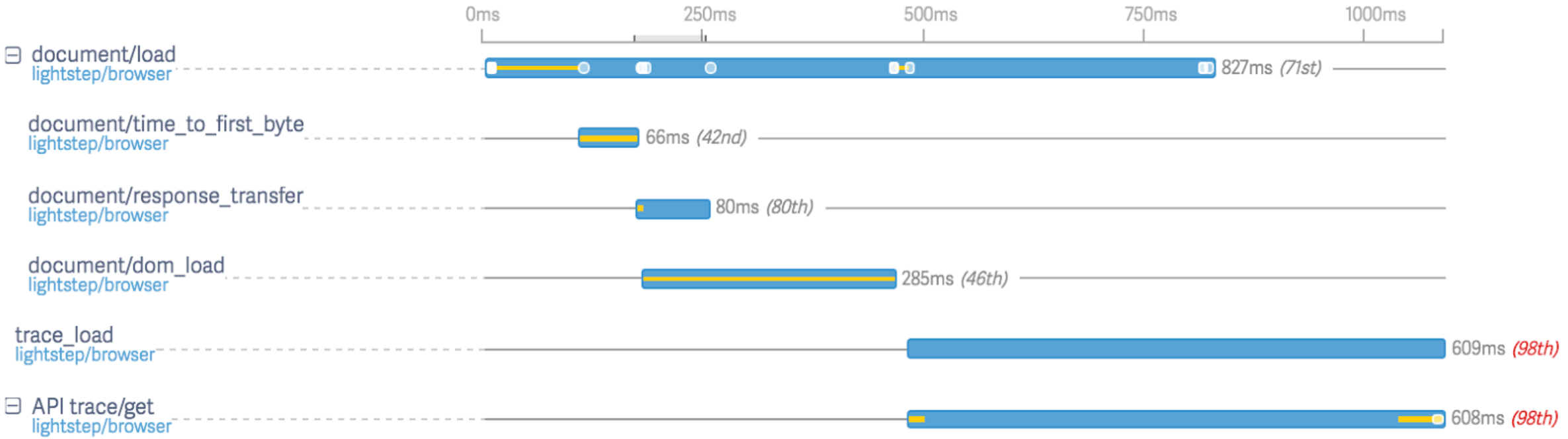
https://linkerd.io

- Linkerd
  - サービスメッシュ
  - マイクロサービスにおけるサービス管理
  - サービス間のトラフィックをコントロール
    - ルーティング
    - トランスポート
    - セキュリティ
    - ロギング
    - 多重化
    - タイムアウト
    - ロードバランシング
    - リトライ
  - サービスの論理名⇔バックエンドのマッピング

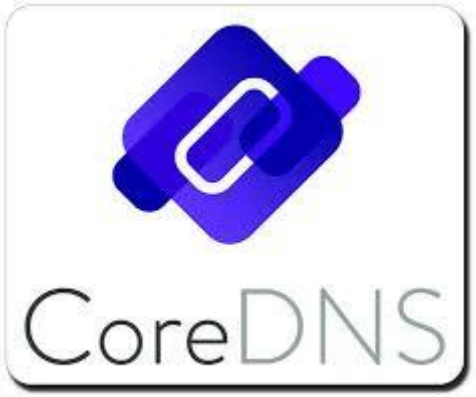
CNCFのプロジェクト pick up ④

FUJITSU

- OPENTRACING
  - トレース解析だとZipkinなどが有名だが、ツール毎にトレースの入れ方がバラバラだと困る
  - 中立なトレーシングアノテーション”OpenTracing”
  - Go, JavaScript, Java, Python, Ruby, Objective-C, C++, C# をサポート
  - Zipkinを含め、トレーサーも多くサポート

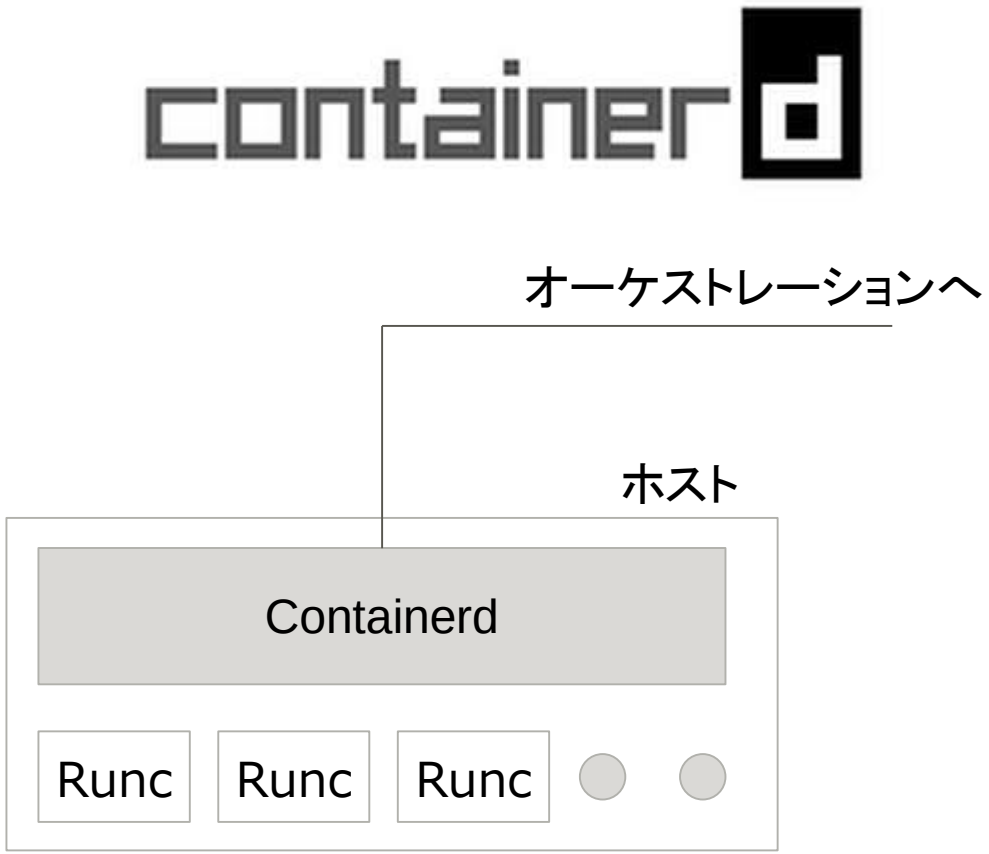


CNCFのプロジェクト pick up ⑤



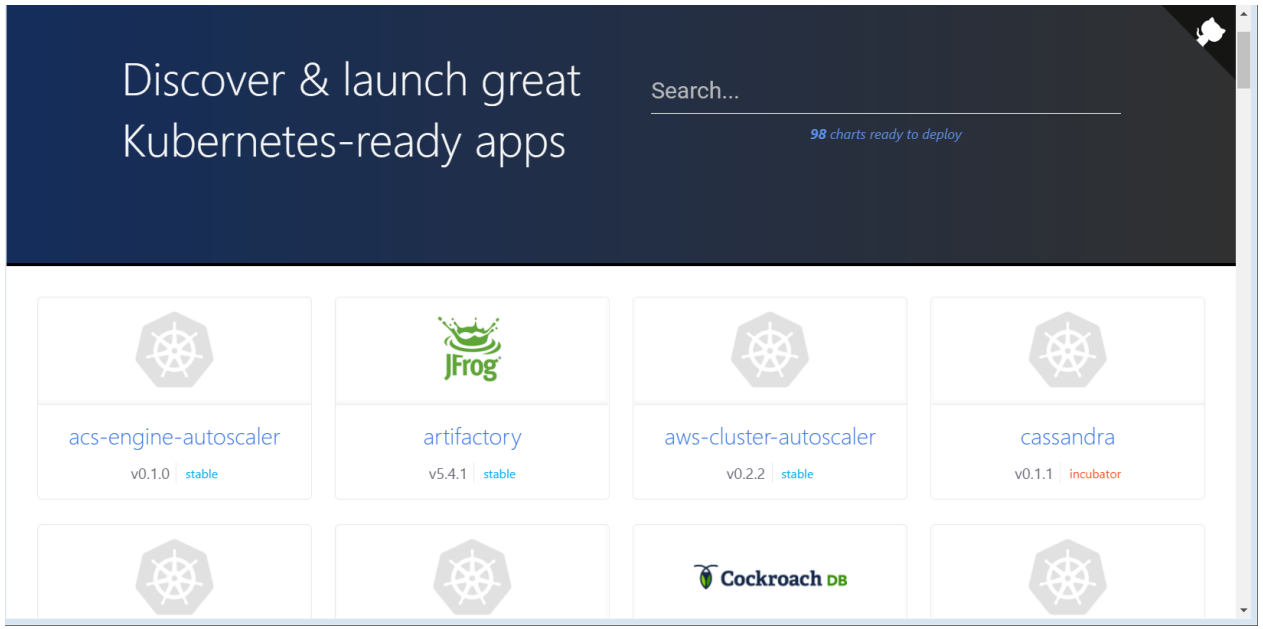
- CoreDNS
  - Service Discovery
    - Kubernetesと連動できるDNSサービス
  - 高速かつ柔軟なDNSサーバ
    - “middleware”プラグインで機能拡張
  - 標準ポート(53)、TLS(DNS by https)、gRPC

CNCFのプロジェクト pick up⑥



- Containerd
- 元dockerエンジンのコア部
  - ノードローカルのコンテナ管理OSS
  - Cloud Native Computing Foundation寄贈

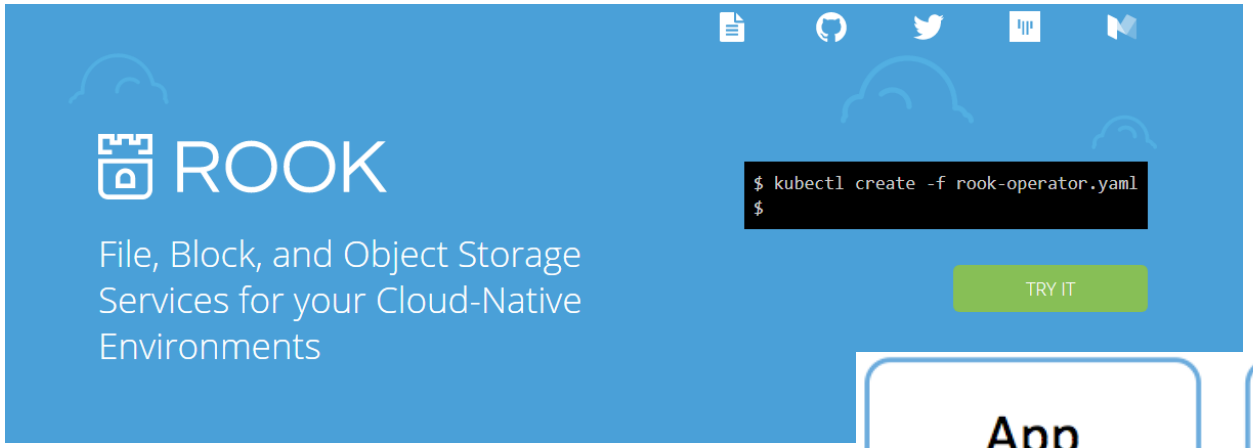
関連プロジェクト①



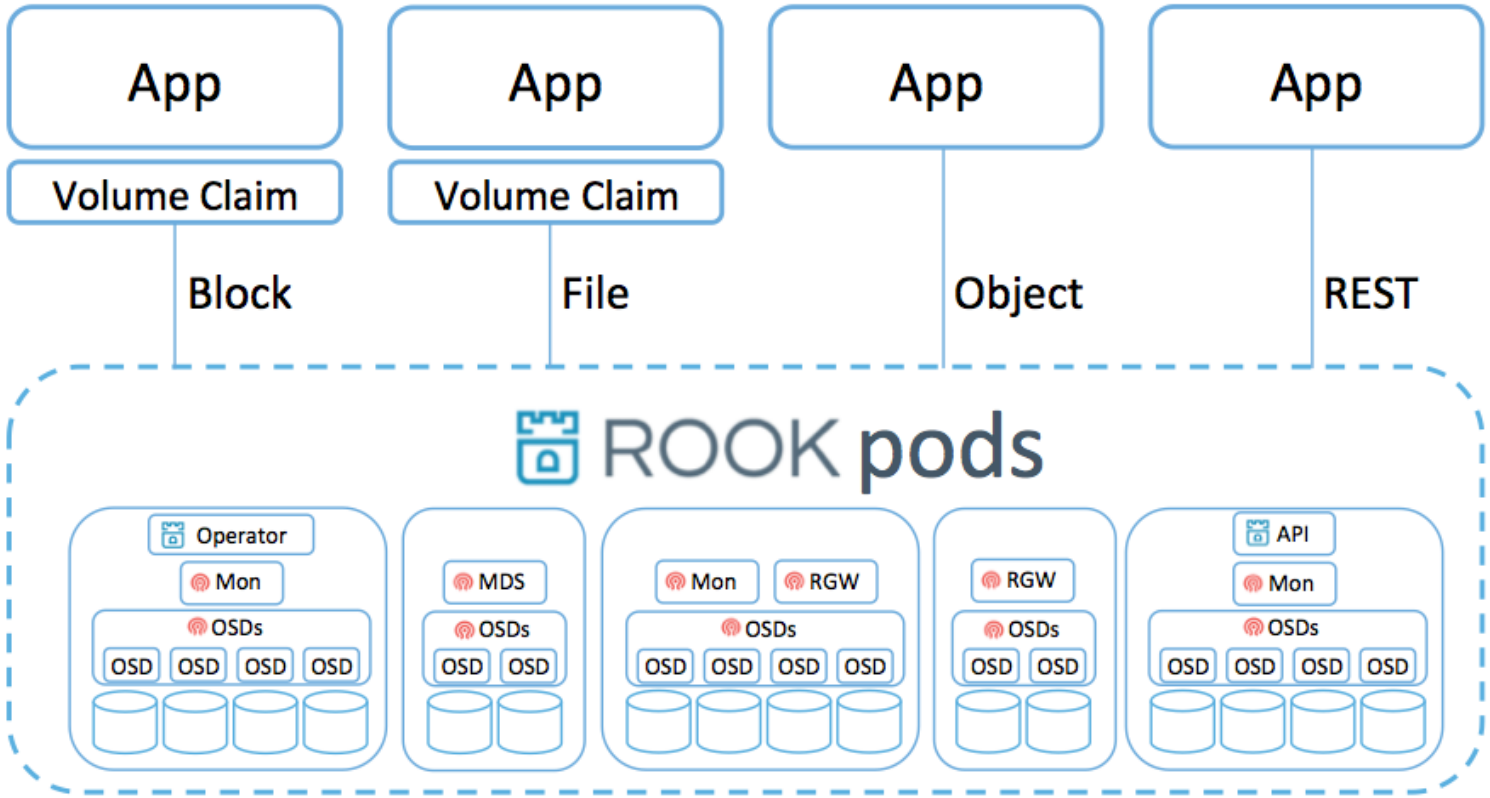
Kubernetes Application の配布・実行には HelmというOSSが人気があり、
http://kubeapps.com で配布されている
※このkuberentesアプリ配布用のWebサイト自体もOSS公開されている

パッケージングには　Helm Chart というツールを使う。
Helmは”LinuxにおけるRPM”(公式サイトより)

関連プロジェクト②

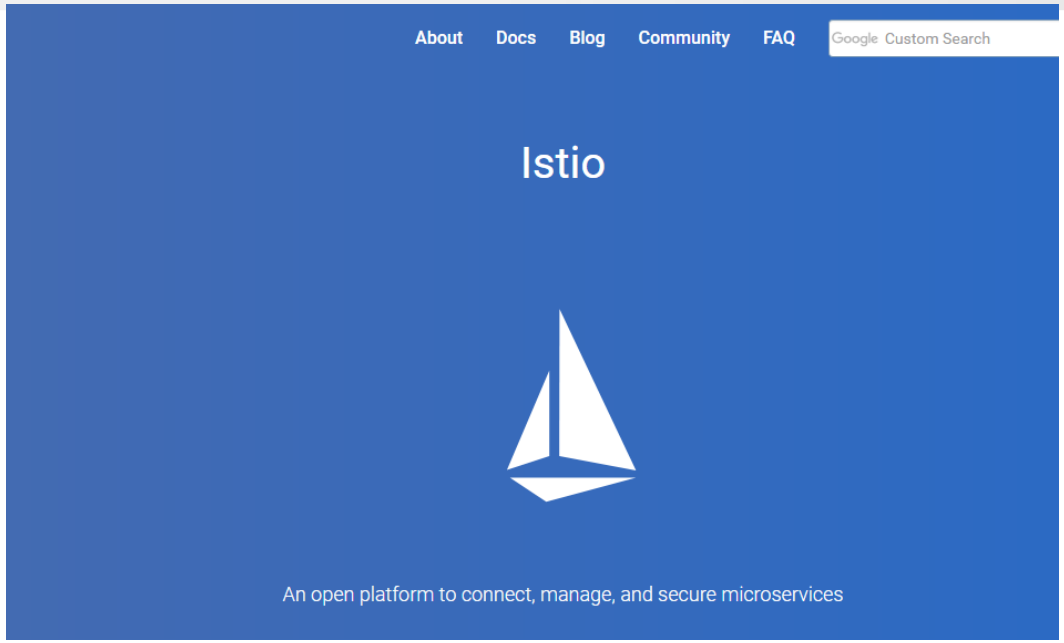


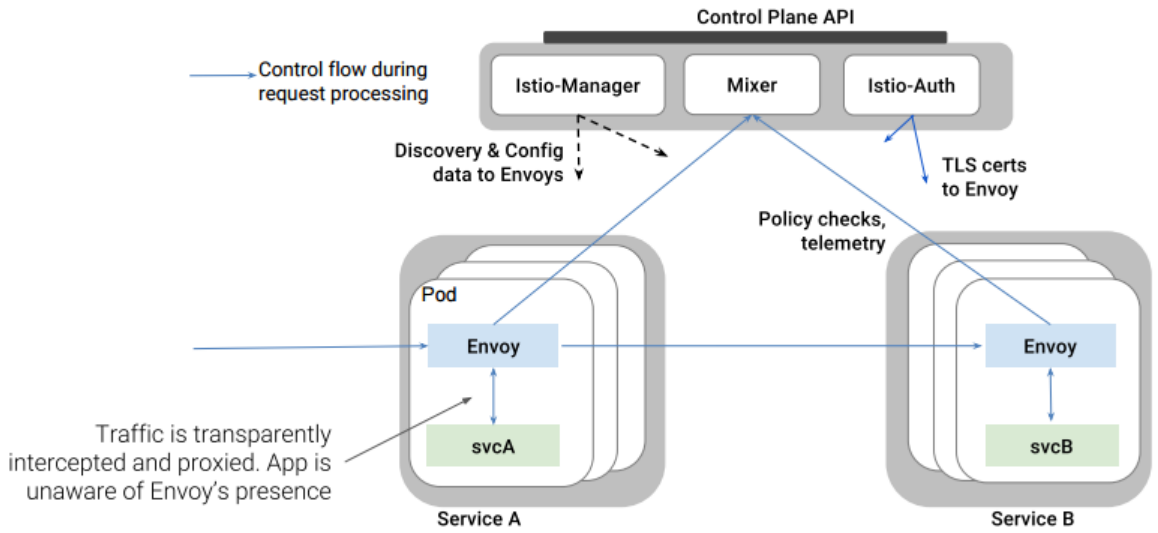
Kubernetesを利用するストレージオーケストレーション
Ceph on-top of Kubernetes



- これとは別にCSI（Container Storage Interface）を定義する動きアリ

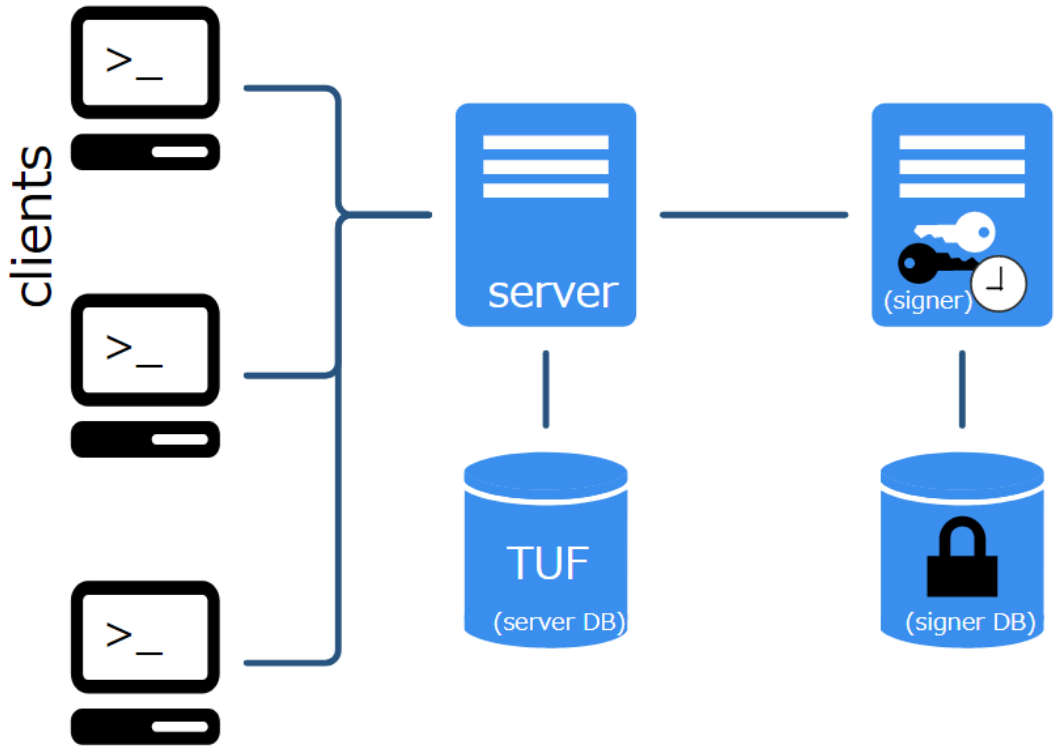
関連プロジェクト③





- Istio
- サービスメッシュ
  - “サービスとネットワーク間のインフラストラクチャ
  - サービスに必要なコントロールを提供
    - HTTP, gRPC, TCPトラフィックによる自動ロードバランス
    - リッチなルーティングルールによるきめ細かいトラフィック制御
    - トラフィックの暗号化、サービスからサービスへの認証、強力な同一性の表明
    - 全体的なポリシー強制
    - 綿密な測定とレポート
  - ローカルのＬＢはサービスメッシュに吸収される
  - 構造的にはEnvoy(LB)をサイドカーでくっつけてIstio Managerからコントロールする構成

関連プロジェクト④



- (Docker)Notary/TUF
  - 公開鍵暗号を使ったコンテンツの内容保証システム
  - 安全な／サイン付きのイメージの公開
  - 安全なイメージの検証
  - The Update Framework (TUF)
    - https://theupdateframework.github.io/

FUJITSU

Kubernetes 1.6/1.7ハイライト

最近のKubernetes

FUJITSU

- Schedulingについて
- 色々な機能
  - RBAC
  - kubeadm
  - StatefulSet
  - InitContainers
  - PodDisruptionBudget
  - NetworkPolicy
  - Node Authorization
- スケーラビリティ関連
  - スケーラビリティ
  - KubeFed
  - オートスケール

Scheduling

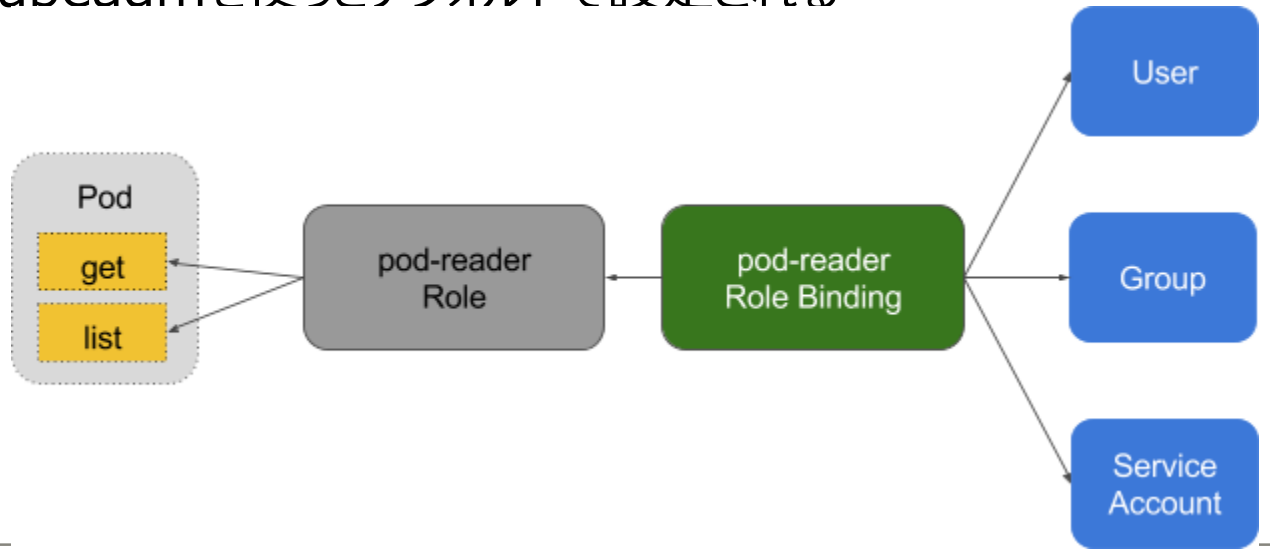
FUJITSU

- MilliCPU
  - KubernetesのCPUリソースはミリＣＰＵという単位で管理され、例えば４ＣＰＵのホストなら4000MiliCPUのリソースを持つ。基本的にはコンテナの要求CPUと各ノードの容量を比較する
- Affinity
  - v1.6 で Node Affinity/Anti-Affinity, taints and tolerations, , pod affinity/anti-affinity追加
    - http://blog.kubernetes.io/2017/03/advanced-scheduling-in-kubernetes.html
- 複数のスケジューラ
  - https://kubernetes.io/docs/tasks/administer-cluster/configure-multiple-schedulers/
  - Kubernetes1.6からスケジューラを複数搭載し、目的に応じて使い分けることが出来る
    - 何か特別についてくるわけじゃないのでスケジューラは自作する
    - Pod<->Nodeのバインド関係の拡張機能などもAlpha化
  - 追加スケジューラはコンテナにしてクラスタに乗せ、アクセス権(RBAC)を設定する
  - Podの spec で schedulerNameを指定する
  - デフォルトスケジューラ＞ https://kubernetes.io/docs/admin/kube-scheduler/

RBAC

FUJITSU

- Role Based Access Control
  - v1.6からBeta　（富士通も頑張って貢献）
  - 役割と紐づけてあらゆるリソースへのアクセスを認可
    - http://blog.kubernetes.io/2017/04/rbac-support-in-kubernetes.html
    - https://kubernetes.io/docs/admin/authorization/rbac/
    - RBAC以前はkubernetesのスタート時にファイルを食わせて設定するような認可(ABAC)しかなかった。
  - Role と Role Bindingを使って設定する。
    - Role <guest> は “pod”のget/listができる…とか。
  - 標準のインストーラ：kubeadmを使うとデフォルトで設定される



kubeadm

FUJITSU

- インストールを含めたKubernetes Clusterのデプロイ管理機能
- インストールで起こること
  - コンテナランタイムを準備して(docker)
  - TLSの為のカギの生成を行い
  - クラスタリングの為のトークンその他を設定し
  - Kubelet を OSサービスとして起動後
  - api server や controller, proxy 等々をPodとして起動する
    ⇒ kubelet 以外は全てコンテナ化

  そこそこ動くけど、たまにハマルことがある・・・・・
- 現在、kubeadmを使ったクラスタ全体のローリングアップデートなどが開発中

期待大

StatefulSet

FUJITSU

- v1.5からBeta　ステートのあるPodを扱うための仕組み
- StatefulSets を使うと以下の特徴を持つPodを作ることが出来る
  - 各Replicaに通し番号を付与、ホスト名が <statefulset name>-<number>になる
    - 番号の小さい順に生成され、番号の大きな順に終了される
    - 前のPodがreadyになるまで後ろのPodは生成されない。
    - 後ろのPodがshutdownするまで前のPodは終了されない
    - V1.7でRolling Update追加。一つずつ順番にアップデートされる
  - 各Podに固有のDNS名がつけられる
  - volumeClaimを定義に内包、各Podにストレージをアタッチする
  - Pod終了の猶予期間(GracePeriod)に０を設定できない
- StatefulSetでPostgresを扱う例
  - http://blog.kubernetes.io/2017/02/postgresql-clusters-kubernetes-statefulsets.html
  - 番号と順番がついているので replicaをつくってもどちらがmasterでどちらがslaveかはっきりする

InitContainers

FUJITSU

- InitContainersはPod中の他のコンテナが動作する前に動作するコンテナ
  - 順番に実行され、実行が終わるまでメインのコンテナは動作しない
- 用途
  - 待ち合わせ
  - 動的なコードのpull
  - テンプレートツール

めっちゃ便利

v1.6からの記法

```
spec:
 containers:
 - name: myapp-container
   image: busybox
   command: ['sh', '-c', 'echo The app is running! && sleep 3600']
 initContainers:
 - name: init-myservice
   image: busybox
   command: ['sh', '-c', 'until nslookup myservice; do echo waiting for myservice; sleep 2; done;']
 - name: init-mydb
   image: busybox
   command: ['sh', '-c', 'until nslookup mydb; do echo waiting for mydb; sleep 2; done;'
```

PodDisruptionBudget

FUJITSU

- 通常、Podは作成されると状態が維持されるが、強制停止される場合もある。
  大まかにPodの強制停止は２種類のケースがある
  - 強制的なもの：ハード故障やVM停止でNodeが消えた場合、Out-Of-Resource　等
  - 自発的なもの：何等かの操作(deploymentの削除等)、Rolling Update 等
- PodDisruptionBudget
  - 同時にダウンしても良いレプリカの数の指定を行う
  - kubectl drain でnodeを落とす際にも必要なpod数を計算して必要ならブロックさせる
    - Drainはノードをクラスタから切り離すコマンド（node上の全Podが安全に終了する）

NetworkPolicy

FUJITSU

- Namespace間のファイアウォール機能、ついに正式機能化(v1.7)
  - https://kubernetes.io/docs/concepts/services-networking/network-policies/
  - 利用するNetwork Pluginに依存する機能なので、例えばflannelでは使えない。
  - いまのところ、CalicoとRomana, Weave Netでサポート
- Kubernetesらしく、Labelをフル活用した機能
  - NamespaceにNetworkPolicy オブジェクトを関連付ける
  - NetworkPolicyはPodSelectorで指定したラベルを持つポッドに対して適用される
  - NetworkPolicyで①特定のラベルを持つネームスペースの②特定のラベルを持つポッドからのアクセスを許可する
  - 許可されていないアクセスは全て禁止される（PodSelectorにヒットするPodのみ）

  アクセスの許可以外の機能はいまのところなし
- PodSelectorを空欄にして、全Podに適用する操作も可能

NodeAuthorization

FUJITSU

- v1.7から追加。
- Node上のKubeletからのアクセスを規定し、Node上のPodがアクセスしているオブジェクト以外のアクセスを禁止する
  - https://kubernetes.io/docs/admin/authorization/node/
  - http://qiita.com/tkusumi/items/f6a4f9150aa77d8f9822

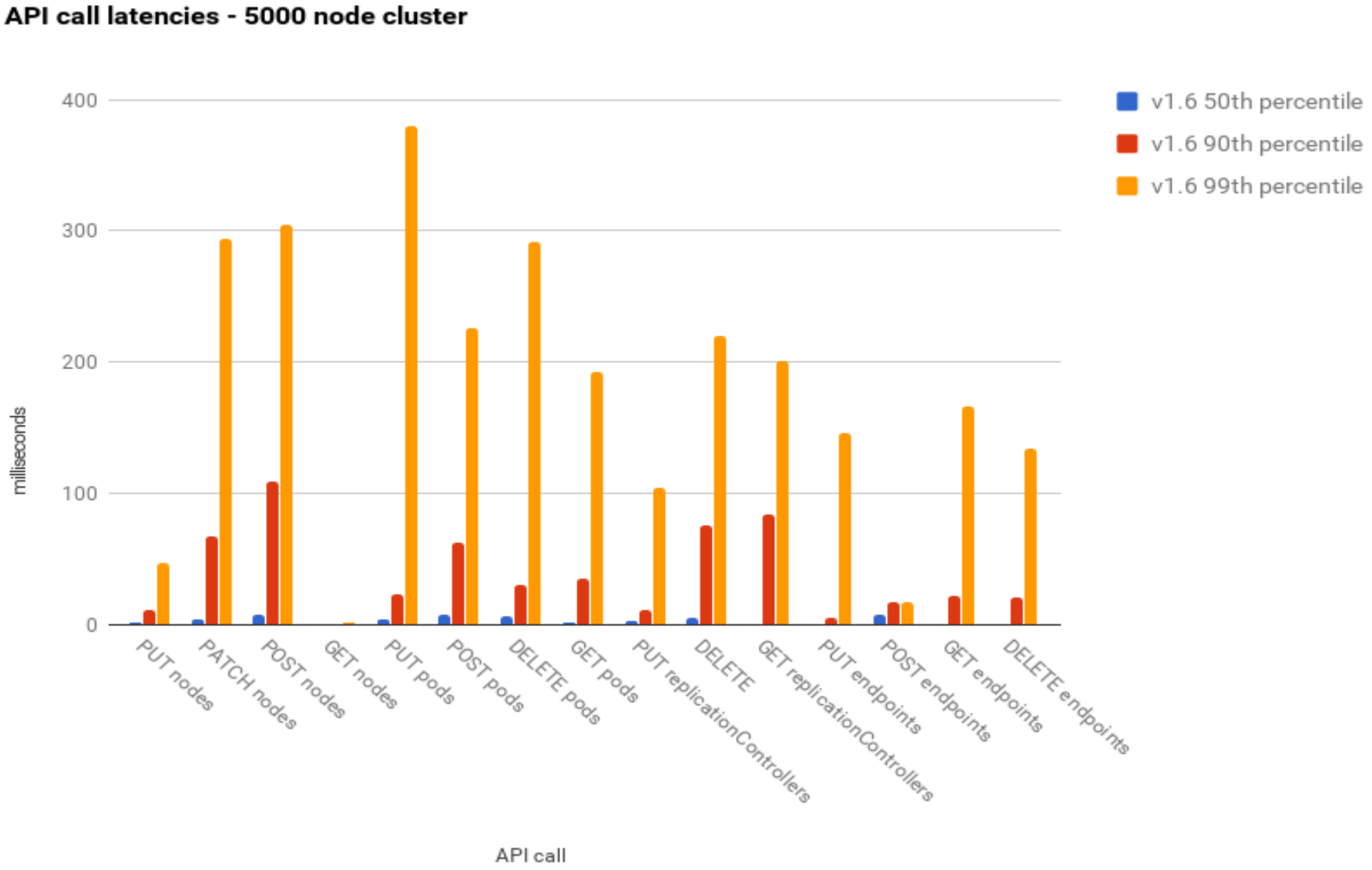
  Kubeadmだとデフォルトでenabled.
- ある Pod X が Secret Y にアクセスし、Node-O で動作しているとして
  - Node-O上から Secret Y にアクセス可能
  - Pod Xが別のノードに動くと、Node-OからはSecretYにはアクセスできない

スケーラビリティ

FUJITSU

- Kubernetesにおける”スケーラビリティ”の基準
  - ＡＰＩの反応： 99%のリクエストが１秒以内
  - Podのスタートが５秒以内 (イメージをＤＬする時間を除く)
- 現在のスケーラビリティ： 5000 node / 150,000 pods
  - Kuberentes 1.0 では 100ノードまでしかスケールしなかった
  - Kubernetes 1.2 では 2000ノード
  - Kuberentes 1.6 で、5000ノードに
- まず悪かったのはREST/JSON と API Server内部データの変換効率(v1.2)
  - 昔からある話だが、ASCIIとGOの内部表現との変換が重かった
  - Protocol buffer フォーマットを使って打開
- Etcd2 から etcd3 になってパフォーマンスが向上(v1.6)
  - 500watch event/sec https://github.com/kubernetes/kubernetes/issues/32361
- いまのところ、「5000以上はFederationを使ってくれ」とのこと

スケーラビリティ



KubeFed

FUJITSU

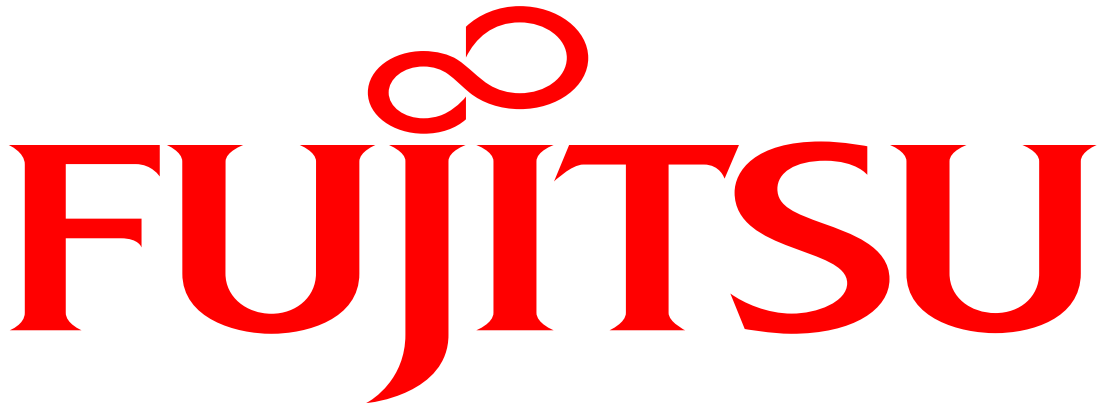
- 複数のkuberentesクラスタを繋げる。V1.5から提供されているコマンド
- Kubernetesの上に federation control plane という形でレイヤを作成
  - Hybrid cloudも可能
  - 今のところLBやDNSのサービス連携が無いと辛いのでクラウド
    - DNSについては”CoreDNS”を使うことが出来る
- 複数のkuberentesクラスタを同期させる
- 以下の機能に対応
  - Cluster,ConfigMap, DaemonSet, Events, Ingress Namespaces, ReplicaSets, Secrets, Services
  - 他のクラスタのサービスに接続できる
  - クラスタ間で同期する(ConfigMap/Secrets)
  - 全クラスタでデーモンを展開(DaemonSets)
  - 全クラスタでレプリカ作成(ReplicaSets)

```
kubefed init fellowship ¥
    --host-cluster-context=rivendell ¥
    --dns-provider="google-clouddns" ¥
    --dns-zone-name="example.com."
```

AutoScale

FUJITSU

- https://kubernetes.io/docs/tasks/run-application/horizontal-pod-autoscale/
- まだAlpha機能
  - Horizontal Pod Autoscaler … Podを動的に増やす
  - Cloud AutoScaler ://GCE, AWS, Azure … VMを動的に増やす
- Pod AutoScalerの動作
  - Pod には必要なCPU/メモリ量を宣言させておく
  - #kubectl autoscale deployment php-apache –cpu-percent=50 –-min=1 –max=10
    - 何秒か毎に現在ターゲットにしているＰｏｄの使えているリソース量をチェック
    - デフォルトは３０秒に一回
    - カスタムメトリクスも利用できる （https://github.com/kubernetes/metrics）



shaping tomorrow with you

DockerCon ‘17

FUJITSU

- 2017.Apr.17-18開催
  - ブースは80以上。コンテナ管理、モニタリング、ネットワーク・ストレージ管理製品が中心
  - Docker Store では500以上の信頼されたコンテナイメージを配布
- キーノート
  - 参加者は5500人、14Mのdocker host, 900Kのdocker アプリケーション, 3300名のdocker 貢献者
  - (OSS版)Dockerの開発者
    - Docker(40%), Microsoft(7.7%), IBM(3.2%), Huawei(3.0%), Red Hat(3.0%)……
  - Dockerを利用するエンタープライズ企業
    - AT&T, Sprint, Verizon, AMGEN, HUDSON ALPHA, MERCK, OPTUM, Juniper, RSA, TIBCO, splulnk, GE, Anthem, Libery Mutual, MetLife等
- 新規トピック
  - Moby Project
  - Hyper-V Isolation Technologyにより、Windows上でLinuxコンテナが動作可能に

Moby

- Moby
  - Moby is an open framework created by Docker to assemble specialized container systems without reinventing the wheel. It provides a “lego set” of dozens of standard components and a framework for assembling them into custom platforms.“