報道各位

2020年7月6日
株式会社インプレスR&D
https://nextpublishing.jp/

実装を追って理解する！

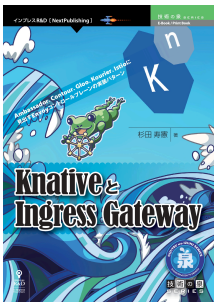
『Knativeと Ingress Gateway』発行

技術の泉シリーズ、7月の新刊

インプレスグループで電子出版事業を手がける株式会社インプレス R&D は、『Knative と Ingress Gateway』(著者:杉田 寿憲)を発行いたします。

最新の知見を発信する『技術の泉シリーズ』は、「技術書典」や「技術書同人誌博覧会」をはじめとした各種即売会や、勉強会・LT 会などで頒布された技術同人誌を底本とした商業書籍を刊行し、技術同人誌の普及と発展に貢献することを目指します。

『KnativeとIngress Gateway』
https://nextpublishing.jp/isbn/9784844378723



著者:杉田 寿憲
小売希望価格:電子書籍版 1600 円(税別)／印刷書籍版 2000 円(税別)
電子書籍版フォーマット:EPUB3／Kindle Format8
印刷書籍版仕様:B5 判／カラー／本文 80 ページ
ISBN:978-4-8443-7872-3
発行:インプレス R&D

＜＜発行主旨・内容紹介＞＞

Knative は Kubernetes 上で動くプラットフォームです。ステートレスなサービスのオートスケールの速度や管理を改善します。以前はルーティングやロードバランシングを行うための Ingress Gateway には Istio しか選択できませんでしたが、現在はさまざまなコンポーネントに代替可能です。

本書では、Knative が利用可能な Ingress Gateway である Ambassador、Contour、Gloo、Kourier、Istio に注目し、「なぜ代替可能なのか」を解明します。

(本書は、次世代出版メソッド「NextPublishing」を使用し、出版されています。)

Knative の概要を紹介

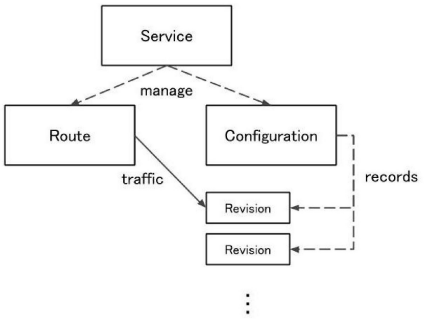
用者から隠蔽します。そのため、開発者は一層コアロジックに集中できます。

Servingは、4つのカスタムリソースとカスタムコントローラー[3]から構成されます。

- Revision: 利用するコンテナイメージと環境変数などの設定のスナップショット
- Configuration: サービス設定の理想状態。変更すると新しいRevisionが作成される
- Route: RevisionへのHTTPリクエストルーティング設定
- Service: RouteとConfigurationから構成されるサービス全体。KubernetesのServiceリソースとは別物

カスタムリソース同士はつぎの図のような関係にあります。

図1.1: Servingを構成するカスタムリソース同士の関係



1.3Eventing

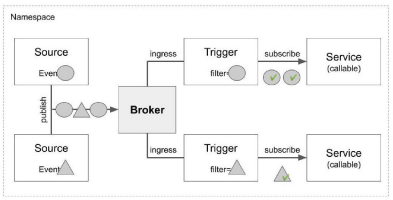
Eventingの責務はイベントのサブスクリプション、デリバリー、管理です。イベントの接続情報やデータの流れを宣言的に定義し、イベントドリブンなアプリケーションを構築できます。

つぎのカスタムリソースとカスタムコントローラーから構成されます。

- Sources: イベントソース。GitHubSourceやKafkaSourceなど、種類ごとに独立したカスタムリソースが用意されている
- Broker: イベントを受け取り、フィルタリングされたものをService（subscriber）に渡す
- Trigger: subscriberにわたすイベントのフィルター

Sourcesが発行するイベントとカスタムリソース同士は、つぎの図のような関係にあります。

図1.2: Eventingを構成するカスタムリソース



1.4まとめ

Knativeは、Kubernetes上でKubernetesのスケーリングやネットワークを抽象化するレイヤーを提供するコンポーネント群です。

Knativeのコンポーネントと、次の章で紹介するGatewayのコンポーネントをKubernetesにインストールして、プラットフォームを構築します。Knativeを利用して開発されたOSSや、Knative互換のAPIをベースに開発されたCloud Run[4]のようなマネージドサービスを利用することもできます。また、Knativeを利用して独自のプラットフォームを構築することもできます。

ここまでの話を図にまとめると、つぎのようになります。

3.https://kubernetes.io/docs/concepts/extend-kubernetes/api-extension/custom-resources/

4.https://cloud.google.com/run

Gateway の特徴を解説

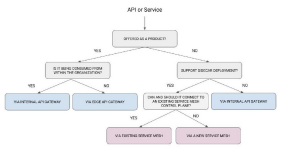
図2.1: API管理のライフサイクル（https://www.cncf.io/blog/2020/03/06/the-difference-between-api-gateways-and-service-mesh/ より）

サービスメッシュは、システム内のすべてのサービス間の接続性を改善します。信頼性が高く、安全で、観測性の高いL4/L7トラフィック制御をサイドカーデプロイモデルを利用して構築します。API Gatewayも、サービスメッシュ内のひとつのサービスになり得ます。

両者は、L7でサービス接続を制御する点で共通しています。しかし、サービスメッシュはL4も含めより広範なサービス接続の制御が可能な一方で、APIのライフサイクルを管理する機能まで提供しているわけではありません。

同記事では、つぎのフローチャートで両者を区別する方法も提案されています。

図2.2: フローチャート（https://istio.io/docs/examples/bookinfo/より）



それぞれが提供する機能は異なるため、両方を組み合わせてシステムを構築するパターンもあり得ます。たとえば、Ambassadorを API Gatewayに利用しつつ、Istioでサービスメッシュを構築することが可能です。一方を利用すると、もう一方が利用できなくなるということはありません。

10.https://www.cncf.io/blog/2020/03/06/the-difference-between-api-gateways-and-service-mesh/

2.3Envoyとコントロールプレーン

Envoyは、Kubernetesに限らずさまざまなプラットフォームでネットワークを抽象化するように実装されています。つぎのような機能を提供します。

- 動的なサービスディスカバリー
- ロードバランシング
- TLS終端
- HTTP/2とgPRCのプロキシ
- サーキットブレーカー
- ヘルスチェック
- パーセント単位のトラフィック分割を用いた段階的ロールアウト
- フォールトインジェクション
- 豊富なメトリクス

アプリケーションのサイドカープロキシとして実行されるEnvoyは、他のアプリケーションと通信するための接続情報を保持します。

通信に使う情報は、Envoyの世界でつぎのように整理されています。[11]クラスターという概念も登場しますが、あくまでもKubernetesの文脈とは別です。

- Host: ネットワーク通信が可能な単位。物理的なハードウェアは、それぞれが個別にアドレス指定できる限り、複数のホストをもつ。
- Downstream: Envoyへのリクエスト送信元。
- Upstream: Envoyのリクエスト送信先。
- Listener: ダウンストリームが接続できる名前付きのネットワークロケーション（ポート、UNIXドメインソケットなど）。Envoyは、ダウンストリームホストが接続するひとつ以上のリスナーを公開する。
- Cluster: Envoyが接続する、論理的にひとまとまりのアップストリームホストのグループ。

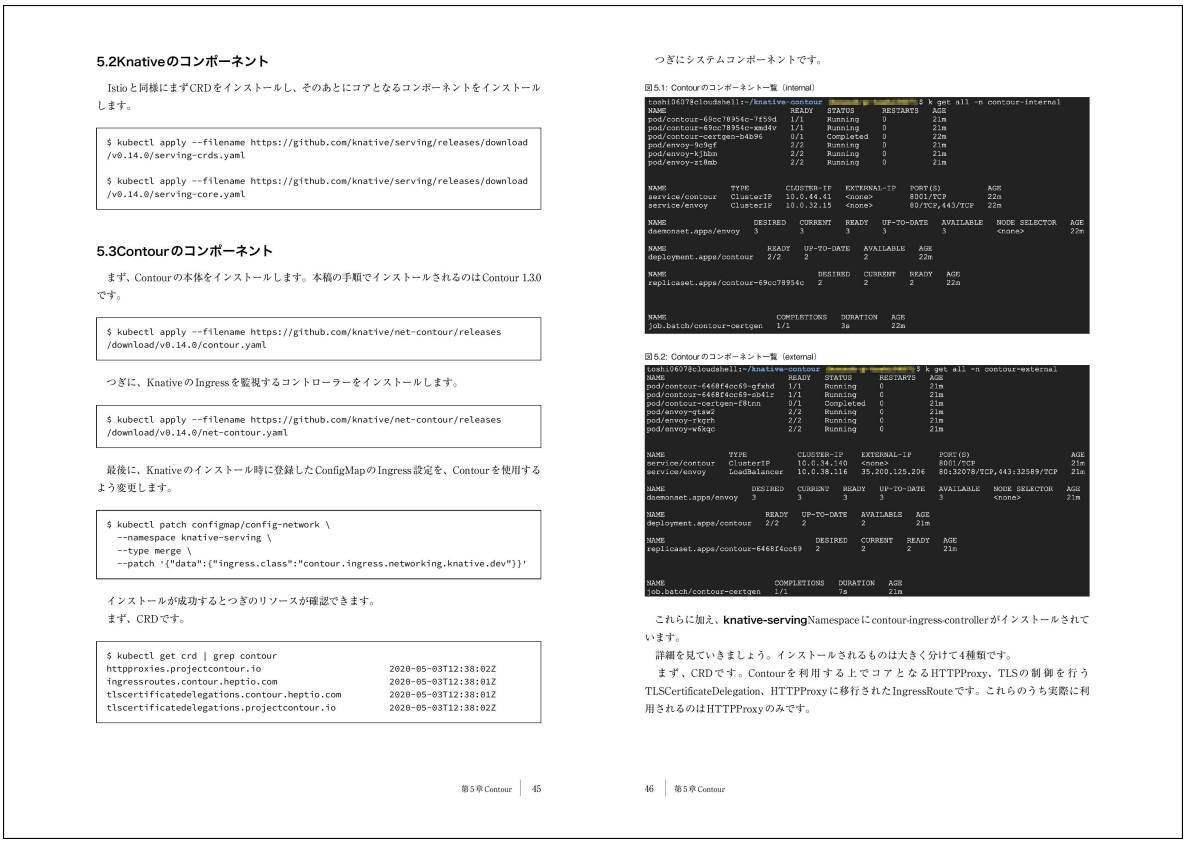
これらの概念は、Envoyを実行するための設定ファイルに現れます。たとえば、リスナーやクラスターはつぎのような設定を使用します。

リスト2.3: リスナーの設定例

```
listeners:
- name: listener_0
  address:
    socket_address: { address: 0.0.0.0, port_value: 10000 }
  filter_chains:
  - filters:
    - name: envoy.http_connection_manager
      typed_config:
        "@type": type.googleapis.com/envoy.config.filter.network.http_connection
_manager.v2.HttpConnectionManager
        stat_prefix: ingress_http
        codec_type: AUTO
        route_config:
```

11.https://www.envoyproxy.io/docs/envoy/latest/intro/arch_overview/intro/terminology

Ambassador、Contour、Gloo、Kourier、Istio から見出す実装パターンを解説



5.2Knativeのコンポーネント

Istioと同様にまずCRDをインストールし、そのあとにコアとなるコンポーネントをインストールします。

```
$ kubectl apply --filename https://github.com/knative/serving/releases/download
/v0.14.0/serving-crds.yaml

$ kubectl apply --filename https://github.com/knative/serving/releases/download
/v0.14.0/serving-core.yaml
```

5.3Contourのコンポーネント

まず、Contourの本体をインストールします。本稿の手順でインストールされるのはContour 1.3.0です。

```
$ kubectl apply --filename https://github.com/knative/net-contour/releases
/download/v0.14.0/contour.yaml
```

つぎに、Knativeの Ingressを監視するコントローラーをインストールします。

```
$ kubectl apply --filename https://github.com/knative/net-contour/releases
/download/v0.14.0/net-contour.yaml
```

最後に、Knativeのインストール時に登録したConfigMapの Ingress設定を、Contourを使用するよう変更します。

```
$ kubectl patch configmap/config-network \
  --namespace knative-serving \
  --type merge \
  --patch '{"data":{"ingress.class":"contour.ingress.networking.knative.dev"}}'
```

インストールが成功するとつぎのリソースが確認できます。
まず、CRDです。

```
$ kubectl get crd | grep contour
httpproxies.projectcontour.io                    2020-05-03T12:38:02Z
ingressroutes.contour.heptio.com                 2020-05-03T12:38:01Z
tlscertificatedelegations.contour.heptio.com     2020-05-03T12:38:01Z
tlscertificatedelegations.projectcontour.io      2020-05-03T12:38:02Z
```


つぎにシステムコンポーネントです。

図5.1: Contourのコンポーネント一覧（internal）

```
toshi06078@cloudshell:~/knative-contour                    $ k get all -n contour-internal
NAME                                READY   STATUS      RESTARTS   AGE
pod/contour-69cc78954c-7f55d        1/1     Running     0          21m
pod/contour-69cc78954c-xmd4v        1/1     Running     0          21m
pod/contour-certgen-b4b96           0/1     Completed   0          22m
pod/envoy-9c9gf                     2/2     Running     0          21m
pod/envoy-kjhbn                     2/2     Running     0          21m
pod/envoy-rtfmb                     2/2     Running     0          21m

NAME              TYPE        CLUSTER-IP    EXTERNAL-IP   PORT(S)          AGE
service/contour   ClusterIP   10.0.44.41    <none>        8001/TCP         22m
service/envoy     ClusterIP   10.0.32.15    <none>        80/TCP,443/TCP   22m

NAME                   DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR   AGE
daemonset.apps/envoy   3         3         3       3            3           <none>          22m

NAME                       READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/contour    2/2     2            2           22m

NAME                                  DESIRED   CURRENT   READY   AGE
replicaset.apps/contour-69cc78954c    2         2         2       22m


NAME                        COMPLETIONS   DURATION   AGE
job.batch/contour-certgen   1/1           3s         22m
```

図5.2: Contourのコンポーネント一覧（external）

```
toshi06078@cloudshell:~/knative-contour                    $ k get all -n contour-external
NAME                                READY   STATUS      RESTARTS   AGE
pod/contour-6468f4cc69-gfxhd        1/1     Running     0          21m
pod/contour-6468f4cc69-zb41r        1/1     Running     0          21m
pod/contour-certgen-f8tnn           0/1     Completed   0          21m
pod/envoy-gtswz                     2/2     Running     0          21m
pod/envoy-rkqrh                     2/2     Running     0          21m
pod/envoy-w6kqo                     2/2     Running     0          21m

NAME              TYPE           CLUSTER-IP      EXTERNAL-IP       PORT(S)                      AGE
service/contour   ClusterIP      10.0.34.140     <none>            8001/TCP                     21m
service/envoy     LoadBalancer   10.0.38.116     35.200.125.206    80:32078/TCP,443:32589/TCP   21m

NAME                   DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR   AGE
daemonset.apps/envoy   3         3         3       3            3           <none>          21m

NAME                       READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/contour    2/2     2            2           21m

NAME                                  DESIRED   CURRENT   READY   AGE
replicaset.apps/contour-6468f4cc69    2         2         2       21m


NAME                        COMPLETIONS   DURATION   AGE
job.batch/contour-certgen   1/1           7s         21m
```

これらに加え、**knative-serving**Namespaceにcontour-ingress-controllerがインストールされています。

詳細を見ていきましょう。インストールされるものは大きく分けて4種類です。

まず、CRDです。Contourを利用する上でコアとなるHTTPProxy、TLSの制御を行うTLSCertificateDelegation、HTTPProxyに移行されたIngressRouteです。これらのうち実際に利用されるのはHTTPProxyのみです。

＜＜目次＞＞

＜＜著者紹介＞＞

株式会社メルカリの YAML エンジニア。マイクロサービスを開発していたが、マイクロサービス全体の開発生産性を高めるべく、現在は Microservices Platform チーム。

＜＜販売ストア＞＞

電子書籍:

Amazon Kindle ストア、楽天 kobo イーブックストア、Apple Books、紀伊國屋書店 Kinoppy、Google Play Store、honto 電子書籍ストア、Sony Reader Store、BookLive!、BOOK☆WALKER

印刷書籍:

Amazon.co.jp、三省堂書店オンデマンド、honto ネットストア、楽天ブックス

※　各ストアでの販売は準備が整いしだい開始されます。

※　全国の一般書店からもご注文いただけます。

【インプレス R&D】 https://nextpublishing.jp/

株式会社インプレスR&D（本社：東京都千代田区、代表取締役社長：井芹昌信）は、デジタルファーストの次世代型電子出版プラットフォーム「NextPublishing」を運営する企業です。また自らも、NextPublishing を使った「インターネット白書」の出版など IT 関連メディア事業を展開しています。

※NextPublishing は、インプレス R&D が開発した電子出版プラットフォーム（またはメソッド）の名称です。電子書籍と印刷書籍の同時制作、プリント・オンデマンド（POD）による品切れ解消などの伝統的出版の課題を解決しています。これにより、伝統的出版では経済的に困難な多品種少部数の出版を可能にし、優秀な個人や組織が持つ多様な知の流通を目指しています。

【インプレスグループ】 https://www.impressholdings.com/

株式会社インプレスホールディングス（本社：東京都千代田区、代表取締役：松本大輔、証券コード：東証 1 部 9479）を持株会社とするメディアグループ。「IT」「音楽」「デザイン」「山岳・自然」「モバイルサービス」「学術・理工学」「旅・鉄道」を主要テーマに専門性の高いメディア&サービスおよびソリューション事業を展開しています。さらに、コンテンツビジネスのプラットフォーム開発・運営も手がけています。

【お問い合わせ先】

株式会社インプレス R&D NextPublishing センター
TEL 03-6837-4820
電子メール: np-info@impress.co.jp