Alibaba Cloud CSK 容器服#Kubernetes版

クイックスタート

Document Version20200226

目次

1	ワークフロー	1
2	基本操作	2
	2.1 Kubernetes クラスターのスピーディーな作成	2
	2.2 イメージを利用したデプロイアプリケーションの作成	7
	2.3 依存関係をベースにした WordPress アプリケーションのデプロイ	28
3	高度な操作	35
	3.1 Helm を使用したマイクロサービスアプリケーションのデプロイ	35
	3.2 プライベートイメージリポジトリを使用してアプリケーションを作成する	42

1 **ワークフロー**

このドキュメントでは、Container Service コンソールでアプリケーションを作成する方法の ワークフローについて説明します。このワークフローには、アプリケーションの作成に必要な手 順が含まれます。



アプリケーションを作成するには、次の手順に従います。

1. RAM ロールを使用して、対応するアクセス権限をユーザーアカウントに付与します。

詳細は、「#unique_2」をご参照ください。

2. Kubernetes クラスターを作成します。

ニーズに合わせてクラスタータイプを選択します。 詳細については、「#unique_3」、 「#unique_4」、「#unique_5」をご参照ください。

3. イメージまたはオーケストレーションテンプレートを使用して、アプリケーションを作成しま す。

既存のイメージまたはオーケストレーションテンプレートを使用するか、イメージまたはオー ケストレーションテンプレートを作成してアプリケーションを作成します。詳細は、「イメー ジを利用したデプロイアプリケーションの作成」と「#unique_7」をご参照ください。

注:

複数のイメージでサポートされるサービスを含むアプリケーションを作成する場合は、オー ケストレーションテンプレートを使用してアプリケーションを作成することを推奨します。

アプリケーションのステータス、およびアプリケーションのサービスとポッドを表示します。
 詳細は、「#unique_8」と「#unique_9」をご参照ください。

2 基本操作

2.1 Kubernetes クラスターのスピーディーな作成

以下のサービスを有効化する必要があります: Container Service、ROS (Resource Orchestration Service) および RAM (Resource Access Management)。より詳細な制限 および処理命令については、*#unique_12*をご参照ください。

Container Service コンソール、*ROS* コンソール、*RAM* コンソールにログインし、該当するサービスを有効化します。

この例では、スピーディーな Kubernetes クラスターを作成方法を紹介します。 いくつかにつ いては、デフォルトの設定、または最も簡単な設定を使います。

- 1. Container Service コンソールにログインします。
- 2. Kubernetes 上で、左側のナビゲーションウィンドウにある [クラスター] をクリックしま す。表示されたページの右上角にある [Kubernetes クラスターの作成] をクリックします。

3. クラスターパラメーターを設定します。

この例では、ほとんどの設定をデフォルトのままにします。 固有の設定については以下の図に 示します。

* Cluster Nam	e sis-cluster				
Region	China North 2 China North 3 China East 1 China East 2 China Sorth 2 (Beijing) (Zhangjiakou) (Hangzhou) (Shanghai) (Shanghai) Asia Pacific SOU 1 US East 1 US West 1 Middle East 1 EU Cert (Mumbai) (Virginia) (Silicon Valley) (Dubai) (Frank	nbers, Chinese characters, English letters and hypnens. Asia Pacific SE buth 1 Asia Pacific SE Asia Pacific SE 3 (Kuala Asia Pacific SE then) Hong Kong 1 (Singapore) 2 (Sydney) Lumpur) 5 (Jakarta) tral 1 furt)			
Zone	China East 1 Zone B *				
VPC	Auto Create Use Existing				
Node Type	Pay-As-You-Go Subscription				
MASTER Confi	guration				
Instance Type	4 Core(s) 8 G (ecs.n1.large) • VQuantBunit(s)				
System Disk	SSD Cloud Disk • 40 G/B ‡				
Worker Instan	c Create Add You can now convert a paid instance to an example of an annual sub-	cription through the ECS Management Console. View details			
WORKER Conf	iguration				
Instance Type	4 Core(s) 8 G (ecs.n1.large) - VQuanti 3 unit	st			
System Disk	SSD Cloud Disk - 40 GiB C				
Attach Data	Disk				
Login * Logon Passv * Confirm Pas	Key Pair Password wd	types of characters (uppercase/lowercase letters, numbers and special characters).			
Docker Versio	n 17.06.2-ce-3				
Kubernetes Vo	ersici0.4				
Configure SNA	AT Configure SNAT for VPC SNAT must be configured when automatically creating a VPC				
SSH Login Cable SSH access for Internet					
II you choose not to open it, prease refer to SSH access to Rubernetes cluster to manually enable SSH access.					
Installing a cloud monitoring plug-in on the node allows you to view the monitoring information of the created ECS instance in the CloudMonitor console					
設定		説明			
カニフス		クラスター名は1文字から63文字以内で数			

	字、漢字、英字およびハイフン (-) を含むこ とができます。
リージョンおよびゾーン	クラスターが設置されているリージョンと ゾーンです。

設定	説明
VPC	 [自動作成] または [既存を使う] を選択できます。 ・ 自動作成: クラスター作成時、システムにより自動的にお使いの VPC に対する NAT ゲートウェイが作成されます。 ・ 既存を使う: 選択した VPC に NAT ゲートウェイがある場合、Container Service ではその NAT ゲートウェイが使用されます。それ以外の場合、デフォルトでシステムにより自動的に NAT ゲートウェイが作成されます。システムによる NAT ゲートウェイの自動作成を希望しない場合は、[VPC 用 SNAT の設定] チェックボックスをオフにしてください。
	注: [VPC 用 SNAT の設定] チェックボック スをオフにした場合、セキュリティ保護 された VPC インターネット環境に実装 するためにユーザー側での NAT ゲート ウェイの設定、または手動で SNAT を設 定する必要があります。設定を行わない 場合、VPC 上のインスタンスが正常にイ ンターネットにアクセスできず、クラス ター作成が失敗します。
ノードタイプ	従量課金およびサブスクリプションがサポー トされます。
マスター設定	インスタンスタイプとシステムディスクを設 定します。 ・ インスタンスタイプ: 詳しくは『インスタ ンスタイプファミリー』をご参照くださ い。#unique_13 ・ システムディスク: SSD および Ultra Disk がサポートされます。

設定	説明
ワーカー設定	ワーカーノードの作成または既存の ECS イ ンスタンスの追加を選択できます。 インスタ ンスの追加を選択した場合、追加したインス タンスを以下のように設定できます。
	 ・ インスタンスタイプ: 詳しくは『インスタ ンスタイプファミリー』をご参照くださ い。#unique_13 ・ システムディスク: SSD および Ultra Disk がサポートされます。 ・ データディスクの接続: SSD、Ultra Disk
ログイン	およひ Basic Disk がサホートされます。 キーペアとパスワードがサポートされ ます。詳しくは『SSH キーペアによる Kubernetes クラスターへのアクセス』をご 参照ください。#unique_14
ポッドネットワーク CIDR および サービス CIDR (オプション)	これらのプランについての詳細 は#unique_15をご参照ください。
	注 このオプションは [既存を使う] から VPC を選択した場合に利用できます。
SNAT の設定	SNAT は [自動作成] を選択し VPC を作 成した場合、必ず設定する必要がありま す。 [既存を使う] により VPC を選択した場 合、SNAT ゲートウェイを自動的に設定する かどうかを選択できます。 SNAT 設定の自 動設定を選択しなかった場合、NAT ゲート ウェイを設定するか、SNAT を手動で設定し ます。
SSH ログイン	 インターネットからの SSH アクセスを 有効にした場合、SSH を利用してクラス ターにアクセスできます。 インターネットからの SSH アクセスを 有効にしなかった場合、クラスターへの SSH を利用したアクセスまたは kubectl を利用した接続ができません。SSH アク セスの有効化を手動で行えます。詳しく は、#unique_16をご参照ください。

設定	説明
モニタリングプラグイン	ECS ノードにクラウドモニタリングプラグ インをインストールできます。これにより、 CloudMonitor コンソール上で、作成した ECS インスタンスのモニタリング情報を参照 できます。
RDS ホワイトリスト (オプション)	RDS インスタンスホワイトリストに対する ECS インスタンスの IP アドレスを追加でき ます。
	注: このオプションは [既存を使う] から VPC を選択した場合に利用できます。
アドバンスコンフィグの表示	 ネットワークプラグイン: Flannel ネット ワークプラグインおよび Terway ネット ワークプラグインがサポートされます。 デフォルトでは、Flannel ネットワーク が使用されます。詳しくは、#unique_17を ご参照ください。 ノードに対するポッド数: 1 つのノードで 実行できるポッドの最大数です。 カスタムイメージ:カスタムイメージをイ ンストールするかどうかが表示されます。 カスタンスによりデフォルトで CentOS バージョンがインストールされます。 クラスター CA: カスタムクラスター CA を使用するかどうかが表示されます。

4. 右上角にある [クラスターの作成] をクリックします。

クラスターの作成が成功したあと、クラスターリスト上に作成したクラスターが表示されます。

•	Container Service - Kubernetes -	Cluster List	Y	ou can create up to 5 cluster	s and can add up to 40	nodes in each cl	uster. Refresh	Create Serverless Kubernetes C	Ouster Create Kubernetes Oluster -
I	Overview _	Help: ${\mathscr O}$ Create cluster ${\mathscr O}$ Scale cluster	Connect to K	lubernetes cluster via kubect	🖉 Manage applicatio	ons with comman	ts		
l	Clusters	Name 🔻							
h	Clusters	Cluster Name/ID	Cluster Type	Region (All) -	Network Type	Cluster Status	Time Created	Kubernetes Version	Action
	Volumes	sis-cluster	Kubernetes	China East 1 (Hangzhou)	VPC vpc-bp1kd7yn4qn	Running	08/14/2018,10:57:5	50 1.10.4	Manage View Logs Dashboard Scale Cluster More+

このようにスピーディーに Kubernetes クラスターを作成できます。

2.2 イメージを利用したデプロイアプリケーションの作成

イメージを利用して、インターネットに接続できる Nginx アプリケーションを作成できます。

Kubernetes クラスターを作成します。詳しくは、#unique_3をご参照ください。

- 1. Container Service コンソールにログインします。
- **2. Kubernetes** で、左側のナビゲーションウィンドウから [アプリケーション] > [デプロイ] の クリックの後、右上角の [イメージから作成] をクリックします。
- 名前、クラスター、名前空間、レプリカおよびタイプを設定します。設定したレプリカパラ メーターの値は、アプリケーションに含まれるポッドの数を示します。 [次へ] をクリックしま す。



この例では、[デプロイ] タイプを選択します。

[名前空間] を設定していない場合、デフォルトではシステムによりデフォルトの名前空間が使われます。

Basic	Information	Container	\geq	Advanced	\rightarrow	Done
Name:	nginx					
	The name should be 1-64 characte	ers long, and can contain numbers,	lower case English le	etters and hyphens, but cannot s	tart with a hyphen.	
Cluster:	k8s-test	v				
Namespace :	default	Ŧ				
Replicas:	2					
Гуре	Deployment	v				
Гуре	Deployment	T				

4. コンテナーを設定します。



アプリケーションのポッドに対して複数のコンテナーを設定できます。

- a) アプリケーションに関する一般設定を行います。
 - ・ イメージ名: [イメージの選択] をクリックして表示されたダイアログボックス内のイメージを選択し、[OK] をクリックします。この例では、Nginx イメージを選択します。

さらに、イメージの指定のため、プライベートレジストリを domainname/namespace / imagename:tag の形式で入力します。

- ・ イメージバージョン:バージョンを選択するため、[イメージバージョンの選択]をクリックします。
 イメージバージョンを選択しない場合、デフォルトでシステムにより最新のバージョンが使用されます。
- ・常にイメージをプルする: Container Service によりデプロイ効率を向上させるためイ メージがキャッシュされます。デプロイ中、新しく設定されたイメージのタグがキャッ シュのイメージのタグと一致した場合、Container Service により、同じイメージを 再度プルするよりも、キャッシュのイメージを再利用することが優先されます。その ため、上位層の業務の利便性のためにお使いのコードおよびイメージを変更した際にイ メージタグを編集しなかった場合、ローカルキャッシュ内の一番最初のイメージがアプ リケーションデプロイに使用されます。[常にイメージをプルする] チェックボックス がオンのとき、Container Service により、アプリケーションのデプロイの際、確実 に最新のイメージおよびコードを使用するため、キャッシュ内のイメージを無視し、イ メージが再度プルされます。
- ・リソース制限:リソース (CPU およびメモリー) に関する上限を設定します。これにより、アプリケーションによるリソースの過度の占有を防ぐことができます。 CPU はミリコア単位、1コアの1000分の1で計測されます。メモリーはバイト単位で計測され、Gi、Mi、または Ki となります。
- ・リソースリクエスト:アプリケーションに対して確保するリソース (CPU およびメモリー) 量を指定します。ここで指定するリソースはコンテナーに対して占有する量です。リソースが不足している場合は、他のサービスまたはプロセスとリソースとの競合

が発生します。 リソースリクエストを指定することで、リソース不足によりアプリケー ションが利用不可となることはありません。

 Init Container: このチェックボックスをオンにすると、便利なツールを含んだ Init Container が作成されます。詳細については、*https://kubernetes.io/docs/concepts/ workloads/pods/init-containers/*をご参照ください。

	Ba	isic Informati	on Cor	ntainer	Advanced
С	ontainer1	Add Contai	ner		
	Image I	Name:	nginx	Select image	
	Image	Version:	latest	Select image version	
-			Always pull image Image pull secret		
Genera	Resource	te Limit:	CPU eg : 500m Core Memory	eg : 128Mi MiB	
	Resourc	te Request:	CPU 500m Core Memory	eg : 128Mi MiB	
	Init Cor	ntainer			

b) オプション:環境変数を設定します。

キーとバリューのペアを利用したポッドの環境変数を設定できます。 環境変数は、環境ラ ベルの追加または、ポッドに関する設定を渡すために使われます。 詳しい情報は、『ポッ ド変数』をご参照ください。

c) オプション: ヘルスチェックを設定します。

ヘルスチェック機能には、Liveness プローブおよび Readiness プローブが含まれます。 Liveness プローブは、いつコンテナーの再起動を行うかを検出します。 Readiness プ ローブは、コンテナーがトラフィックの受信が可能な状態かを判定します。 ヘルスチェッ クに関する詳しい情報は、*https://kubernetes.io/docs/tasks/configure-pod-container/configure-liveness-readiness-probes*をご参照ください。

Liveness	🖉 Enable				
	нттр		ТСР	Command	~
	Protocol	HTTP	v		
	path				
	Port				
	Http Header	name			
		value			
	Initial Delay	3			
	Period	10			
	Timeout	1			
	Success Threshold	1			
	Failure Threshold	3			
n Checl					
Readiness	Enable				
	НТТР		ТСР	Command	~
	Protocol	нттр	v		
	path				
	Port				
	Http Header	name			
		value			
	Initial Delay	3			
	Period	10			
	Timeout	1			

リクエストメソッド	設定の説明
HTTP リクエスト	HTTP GET リクエストによりコンテナー へ送信されます。 以下のパラメーターがサ ポートされます。
	 プロトコル: HTTP または HTTPS です。 パス: アクセスする HTTP サーバーのパスです。 ポート: コンテナーにより開放されるポートの番号または名称です。ポート番号は1から 65335 である必要があります。 HTTP ヘッダー: HTTP リクエストでのカスタマイズされたヘッダーです。 HTTP は複数のヘッダーを同時に送信できます。キーとバリューの設定をサポートしています。 初期遅延(秒): "initialDelaySeconds"です。コンテナーが起動してから、最初のプローブの待機時間を秒単位で設定します。デフォルトでは3です。
	 期間(秒): "periodseconds" です。プローブの実行間隔です。デフォルト値は10です。最小値は1です。 タイムアウト(秒): "timeoutSeconds" です。プローブのタイムアウトまでの時間です。デフォルト値は1で、最小値は1です。 成功しきい値: 失敗したプローブの発生後、成功したとみなされる連続して成功したプローブの最少の数です。デフォルトは1で、最小値は1です。Livenessプローブでは1である必要があります。 失敗しきい値: 成功したプローブの発生後、失敗したとみなされる連続して失敗したとみなされる連続して失敗したプローブの最少の数です。デフォルト値は3です。最小値は1です。

リクエストメソッド	設定の説明
TCP 接続	TCP ソケットはコンテナーに送信されま す。kubelet は指定されたポートにより、 お使いのコンテナーへのソケットの作成を 試みます。接続が確立された場合、コンテ ナーは正常であるとみなされます。接続さ れなかった場合、失敗とみなされます。以 下のパラメーターがサポートされます。
	 ・ポート:コンテナーにより開放される ポートの番号または名称です。ポート番 号は1から 65335 である必要がありま す。 ・初期遅延 (秒): "initialDelaySeconds "です。コンテナーが起動してから、 最初の Liveness プローブまたは
	 Readiness プローブの待機時間を秒単位で設定します。デフォルトでは15です。 ・期間(秒): "periodseconds" です。プローブの実行間隔です。デフォルト値は10です。 鼻小値は1です
	・ タイムアウト (秒): "timeoutSeconds" です。 プローブのタイムアウトまでの時 間です。 デフォルト値は1で、最小値は 1です。
	 ・ 成功しきい値: 失敗したプローブの発生 後、成功したとみなされる連続して成功 したプローブの最少の数です。 デフォル トは1で、最小値は1です。Liveness プローブでは1である必要があります。 ・ 失敗しきい値: 成功したプローブの発生 後、失敗したとみなされる連続して失敗 したプローブの最少の数です。 デフォル ト値は3です。最小値は1です。

リクエストメソッド	設定の説明
コマンド	コンテナーにおいて、プローブ検出コマン ドを実行することで、コンテナーのヘルス ステータスを検出します。 以下のパラメー ターがサポートされます。
	 コマンド:コンテナーのヘルスのステー タスを検出するプローブコマンドです。 初期遅延(秒):"initialDelaySeconds "です。コンテナーが起動してから、 最初の Liveness プローブまたは Readiness プローブの待機時間を秒単 位で設定します。デフォルトは5です。 期間(秒): "periodseconds" です。プ ローブの実行間隔です。デフォルト値は 10です。最小値は1です。 タイムアウト(秒): "timeoutSeconds" です。プローブのタイムアウトまでの時 間です。デフォルト値は1で、最小値は 1です。 成功しきい値: 失敗したプローブの発生 後、成功したとみなされる連続して成功 したプローブの最少の数です。デフォル トは1で、最小値は1です。Liveness プローブでは1である必要があります。 失敗しきい値: 成功したプローブの発生 後、失敗したとみなされる連続して失敗
	ト値は3です。最小値は1です。

d) ライフサイクルルールを設定します。

コンテナーのライフサイクルに関する以下のパラメーターを設定できます:コンテナー設定 スタート、ポストスタートおよび プレストップ。詳しい情報は、『*https://kubernetes.io/docs /tasks/configure-pod-container/attach-handler-lifecycle-event/*』をご参照ください。

・コンテナー設定: [stdin] チェックボックスをオンにすると、コンテナーへの標準入力が有効になります。[tty] チェックボックスをオンにすると、コンテナーへの仮想ターミナルを割り当てます。これにより、コンテナーヘシグナルを送ることができます。通常、これら2つのオプションは併用されます。これは、ターミナル(tty)をコンテナー

標準入力 (stdin) にバインドすることを示します。 たとえば、対話型プログラムは、 ユーザーから標準入力を取得し、取得した標準入力をターミナルに表示します。

- ・スタート:コンテナーへの開始前コマンドおよびパラメーターを設定します。
- ・ポストスタート:コンテナーへの開始後のコマンドを設定します。
- ・プレストップ:コンテナーへの終了前コマンドを設定します。

	Container Config:	🗆 stdin 🗆 tty
fo ouclo	Start:	Command ["/bin/sh","-c","echo Hello > /usr/share/message"] Parameter
-	Post Start:	Command
	Pre Stop:	Command ["/usr/sbin/nginx","-s","quit"]

e) オプション: データボリュームを設定します。

ローカルストレージおよびクラウドストレージの設定ができます。

- ・ ローカルストレージ:ホストパス、コンフィグマップ、シークレットおよび一時ディレクトリをサポートします。ローカルデータボリュームはコンテナーパスの対応するマウントソースにマウントされます。詳しい情報は、『ボリューム』をご参照ください。
- クラウドストレージ: 3 つの種類のクラウドストレージをサポートします: クラウドディ スク、NAS (Network Attached Storage) および OSS (Object Storage Service) で す。

このページの例では、クラウドディスクをデータボリュームとして設定し、クラウドディス クをコンテナーパス /tmp にマウントします。 このパスで生成されたコンテナーデータは クラウドディスクに保存されます。

Data Volume:	 Add local storage 			
	Storage type	Mount source	Container Path	
	• Add cloud storage			
	Storage type	Mount source	Container Path	
	Disk •	pvc-disk	▼ /tmp	•

f) オプション: Log Service を設定します。 ログの収集方法および、このサービスへのカス タムタグを設定します。 🗎 注:

Kubernetes クラスターがデプロイされ、クラスターにログプラグインがインストールさ れていることを確認してください。

ログの収集方法の設定は以下のようになります:

- Log Store: 収集されたログを保存するために使用される Log Service で生成された Logstore を設定します。
- ・コンテナー上のログパス: stdout およびテキストログをサポートします。
 - stdout: コンテナーの標準出力ログを収集します。
 - テキストログ:コンテナー上の指定したパスでのログを収集します。このページの例では、パス "/var/log/nginx" にあるテキストログを収集します。 ワイルドカードも サポートされます。

カスタムタグの設定もできます。 カスタムタグはコンテナー出力ログへ収集されます。 カ スタムタグはコンテナーログのタグ付けに有用で、ログ統計やフィルタリングなどのログ 解析を効率的に行うことができます。

	Log Service:	Note: please ensure that	at cluster has deployed log plug-ins.	
		🕙 Configuration		
		Log Store	Log path in the container (can be set to stdout)	
uo		catalina	stdout	•
mfigurati		access	/var/log/nginx	•
Log Co		Custom Tag		
		Name Of Tag	Value Of Tag	
		арр	nginx	•

5. 設定が完了したら、[次へ] をクリックします。

- 6. 詳細設定を行います。
 - a) アクセス制御を設定します。

バックエンドポッドの公開方法を設定できます。[作成] をクリックします。 この例で

は、[Cluster IP サービス] および [Ingress] を選択し、インターネットにアクセス可能な nginx アプリケーションを作成します。



アプリケーションの通信要件を満たすため、ニーズに応じたアクセス制御を設定できます:

内部アプリケーション:クラスター内でのみ実行されるアプリケーションで、必要に応じて内部通信のためのクラスター IP またはノードポートに関するサービスを作成できます。

- ・外部アプリケーション:インターネットに公開される必要のあるアプリケーションで、 以下の方法のうち1つを使ってアクセス制御を設定できます。
 - Server Load Balancer サービスの作成: Alibaba Cloud による SLB (Server Load Balancer) サービスを使用します。これにより、アプリケーションによるイ ンターネットへのアクセスが可能にします。
 - ClusterIP または NodePort の作成および Ingress の作成: この方法では Ingress を通じてインターネットへのアクセスを提供します。詳細については、https:// kubernetes.io/docs/concepts/services-networking/ingress/をご参照ください。

	Basic Informa	ation	>	Container	Advanced	Done	
Control	Service(Service)	Create					
Access (Ingress(Ingress)	Create					
Scale	HPA	Enable					
	Node Affinity	Add					
Scheduling	Pod Affinity	Add					
57	Pod Anti Affinity	Add					
						Prev Crea	te

A. サービスの右側にある [作成] をクリックします。表示されたダイアログボックスでサービスを設定し、[作成] をクリックします。

Create Service		\times
Name: Type:	nginx-svc Server Load Balancer v public v	
Port Mapping:	 OAdd service port 80 80 TCP ▼ 	
annotation: Tag:	 Add Annotations for load balancer Add 	
	Create	ncel

- 名前:カスタマイズ名を入力できます。デフォルトでは、applicationname-svc となっています。
- ・タイプ:以下の3つのサービスタイプから1つを選びます。
 - **ClusterIP:** クラスターの内部 **IP** アドレスを使用してサービスを公開します。 このタイプを選択すると、サービスはクラスター内でのみアクセスできます。
 - NodePort: それぞれのノードで IP アドレスおよび静的ポート (NodePort) を使用してサービスを公開します。NodePort サービスは ClusterIP サービスヘルーティングされます。これは自動的に作成されます。 <NodeIP>: <NodePort> リクエストによりクラスター外部の NodePort サービスヘアクセスできます。
 - Server Load Balancer: Server Load Balancer サービスです。これは
 Alibaba Cloud により提供されます。このタイプのサービスを使用して、イン
 ターネットアクセスまたはイントラネットアクセスを設定できます。Server

Load Balancer は **NodePort** サービスおよび **ClusterIP** サービスヘルーティン グされます。

- ・ポートマッピング:サービスポートおよびコンテナーポートを追加します。タイプに [NodePort] を選択した場合、ポートの競合を避けるためノードポートを設定する必要があります。TCP プロトコルおよび UDP プロトコルがサポートされます。
- アノテーション:サービスへアノテーションを追加します。Server Load Balancer
 設定パラメーターがサポートされます。詳しくは、#unique_18をご参照ください。
- ・ ラベル:サービスヘラベルを追加し、サービスを識別できます。
- B. イングレスの右側にある [作成] をクリックします。表示されたダイアログボックスで、バックエンドポッドへのルートに関するルールを設定し、[作成] をクリックします。ルートの設定に関する詳しい情報は、#unique_19を参照してください。

注:

イメージを使用してアプリケーションを作成した際、1 つのサービスにのみ イングレス を作成できます。 このページの例では仮想ホスト名をテスト用のドメイン名として使 用します。ホストヘレコードの追加が必要です。実際の作業シナリオでは、登録され たドメイン名を使用します。

101.37.224.146 foo.bar.com # Ingress の IP アドレス

Create		\times
Name: Rule:	nginx-ingress	
	Domain foo.bar.com Select *.c62d7cbe628444321aca3ef92b478d193.cn-beijing.alicontainer.com or Custom path e.g./ Service • Add Name Port Weight nginx-svc 80 • 100 100.0% EnableTLS	
Grayscale release:	Add After the gray rule is set, the request meeting the rule will be routed to the new service. If you set a weight other than 100, the request to satisfy the gamma rule will continue to be routed to the new and old version services according to the weights.	
annotation:	 Add rewrite annotation 	
Tag:	Opp Space State St	
	Create Car	ncel

C. 作成されたサービスおよび Ingress がアクセス制御セクションに表示されます。 サー ビスおよび Ingress を再度設定するには、[更新] および [削除] をクリックします。

Crea	ate Application						
	Basic I	information	Container	>	Advanced		Done
	Service(Service)	Update Delete					
		service port	Containe	er Port		Protocol	
2		80	80			ТСР	
ccess Cont	Ingress(Ingress)	Update Delete					
×		Domain	path	Name		service port	
		foo.bar.com		nginx-svc		80	
Scale	HPA	Enable					
B	Node Affinity	Add					
chedulin	Pod Affinity	Add					
Ň	Pod Anti Affinity	Add					
							Prev Create

b) オプション: HPA (Horizontal Pod Autoscaling) を設定します。

HPA を有効化するかどうか選択することができます。 さまざまな負荷状況でアプリケー ションの要求を満たすために、Container Service によりコンテナーのオートスケーリン グがサポートされます。これにより、コンテナーの CPU およびメモリーの使用率に応じ て、コンテナーの数を自動的に調整します。

Metric: CPU Usage
Condition: Usage 70 %
Maximum Replicas: 10 Range : 2-100
Minimum Replicas: 1 Range : 1-100

注:

オートスケーリングを有効化するために、デプロイに関して必要なリソースを設定する必 要があります。 設定しない場合、コンテナーオートスケーリングは有効になりません。 コンテナーの基本設定をご参照ください。

・ 測定: CPU およびメモリーです。 必要に応じてリソースタイプを設定します。

- 条件:リソース使用率のパーセンテージでの値です。リソース使用率がこの値を超えた
 とき、コンテナーの拡張が始まります。
- ・レプリカの最大数:デプロイが拡張することができるレプリカの最大数です。
- ・ レプリカの最小数:デプロイが縮小させることができるレプリカの最少数です。
- c) オプション: スケジューリングアフィニティ を設定します。

ノードアフィニティ、ポッドアフィニティおよびポッドアンチアフィニティを設定できま す。詳しくは、『*https://kubernetes.io/docs/concepts/configuration/assign-pod-node/#affinity-andanti-affinity*』をご参照ください。



アフィニティスケジューリングは、タグおよびポッドタグよって行われます。 ビルトイン タグを使って、あらかじめ設定したノードおよびポッドへのタグと同様にスケジューリン グすることができます。

A. ノードタグの設定で、ノードアフィニティを設定します。

Create		\times
Required:	Add Rule Selector • Add	
	Tag Name Operator Tag Value kubernetes.io/hostname In In	
	kubernetes.io/hostname 🔻 In 👻 c	
Preferred:	Add Rule Weight 100	
	Selector • Add Tag Name Operator Tag Value	
	kubernetes.io/hostname 🔻 NotIn 🔻 c	
	ОК	Cancel

ノードスケジューリングは、必須ルールと優先ルール、および

"In"、"NotIn"、"Exists"、"DoesNotExist"、"GT" および **"LT"** のようなさまざまな 演算子のどちらもサポートしています。

- ・必須ルールが満たされ、requiredDuringSchedulingIgnoredDuringExe cutionに対応している必要があります。必須ルールは NodeSelector と同様の影 響を持ちます。このページの例では、ポッドが対応するタグがあるノードのみスケ ジューリングされます。複数の必須ルールを追加できますが、そのうち1つにのみ 合致する必要があります。
- ・優先ルールは必ずしも満たされなくてもよく、preferredDuringSched ulingIgnoredDuringExecutionに対応している必要があります。この例では、 スケジュールは、対応するタグがあるノードにはスケジューリングを行いません。

優先ルールには重み付けをすることができます。 条件を満たす複数のノードが存在 する場合、最も大きな重みが付けられたノードが優先されたスケージューリングが行 われます。 複数の優先ルールを定義できますが、全てのルールはスケジューリング の前に満たされている必要があります。

B. ポッドアフィニティを設定し、他のポッドと共にトポロジドメイン上のアプリケーションのノードをデプロイします。 たとえば、お互いに通信しあうサービスは、ポッドア

フィニティスケジューリングの設定によりサービス間のネットワークの遅延が低減され るため、同じトポロジドメイン (ホストと同様) ヘデプロイされます。

Create		×
Required:	Add Rule	
	Namespace	
	default	
	Topology Key	
	kubernetes.io/hostname	
2	Selector • Add View Application List	
	Tag Name Operator Tag Value	
	app In v nginx O	
Preferred:	Add Pula	
	Add Kale	
	Weight	
	100	
	Namespace	
	default v	
	Topology Key	
	kubernetes.io/hostname	
	Tag Name Operator Tag Value	
	app NotIn 🔻 wordpress 🗢	
		Cancol
	OK (cancer

ノードで実行中のポッドのタグに基づいて、ポッドのスケージューリングが行われま す。使用可能な式は、In、 NotIn、Exists、DoesNotExistです。

- 必須 ルールが満たされ、requiredDuringSchedulingIgnoredDuringExe
 cution に対応している必要があります。ポッドアフィニティスケジューリングは設
 定されたルールを合致する必要があります。
 - 名前空間:スケジューリングポリシーはポッドタグに基づいており、名前空間に制約を受けます。
 - トポロジキー:ノードタグを通じてスケジューリングするドメインを指定します。
 たとえば、kubernetes.io/hostnameをトポロジキーとして設定した場合、

ノードはトポロジの定義に使用されます。 beta.kubernetes.io/osをトポロジ として指定した場合、ノードのオペレーティングシステムがトポロジの定義に使 用されます。

- セレクター:セレクターの右にある [追加] ボタンをクリックすると、強い制約を 持つルールを追加できます。
- アプリケーションリストの表示: [アプリケーションリストの表示] をクリックする
 と、ダイアログボックスが表示されます。ダイアログボックスで、それぞれの名
 前空間のアプリケーションを参照でき、アプリケーションのタグをアフィニティ設
 定ダイアログボックスにエクスポートできます。
- 強い制約:既存のアプリケーション、演算子およびタグの値に対するタグを設定で
 きます。この例では、app: nginx タグのあるアプリケーションを実行するホス
 トに対しアプリケーションを作成するようスケジューリングします。
- 優先 ルール、つまり、緩い制約を持ったルールは、preferredDuringSched ulingIgnoredDuringExecution に対応しています。ポッドアフィニティスケ ジューリングは設定したルールにできるだけ早く満たされます。緩い制約を持った ルールに関しては、それぞれのルールに重み付けすることができます。他の設定要 件は、強い制約を持つルールと同様なものになります。

🧾 注:

Weight: 1 つの緩い制約を持つルールに対して 1 から 100 の範囲で weight を設定します。 設定された緩い制約を持つルールを満足するノードの weight はアルゴリズムによって算出され、最も大きな weight を持つノードに対してポッドがスケジューリングされます。

- C. ポッドアンチアフィニティを設定し、他のボッドを含むトポロジドメイン上でアプリケーションのポッドをデプロイします。ポッドアンチアフィニティスケジューリングを使用するシナリオ:
 - ・サービスのポッドを、異なるトポロジドメイン (ホストなど) へ分配し、サービスの 安定性を向上させます。
 - ・ポッドに対し、ノードへの排他的なアクセスを承認し、ノードのリソースを他のポッドが使用しないことを保証します。
 - ・ お互いに影響しあうサービスに関するポッドを異なるホストに分配します。



ポッドアンチアフィニティスケジューリングの設定方法は、ポッドアフィニティのスケ ジューリングと同様のものです。ただし、同じスケジューリングルールでもポッドア ンチアフィニティスケジューリングでは違う意味を持ちます。シナリオをもとに適切な スケジューリングルールを選択します。

- 7. [作成] をクリックします。
- アプリケーションの作成後、作成成功ページが表示され、デフォルトでは、アプリケーション に含まれるオブジェクトがリスト化されます。 [詳細の表示] をクリックし、デプロイの詳細を ご参照ください。

	Create	Success	
Create dep	loyment	nginx	Succeeded
Create s	ervice	nginx-svc	Succeeded
Create ir	ngress	nginx-ingress	Succeeded
	View detail	Create again	

デフォルトでは、"nginx-deployment"ページが表示されます。

Deployment nginx 🔹 Back to List									
Overview									
Name:	nginx								
Namespace:	default								
Time Created:	10/11/2018,10:26:08								
Label:	app:nginx								
annotation:	deployment.kubernetes.lo/revision:1								
Selector:	app:nginx								
Strategy:	RollingUpdate								
Status:	Updated:2 , Unavailable:0 , Replica:2								
Trigger 1. You can only have or	e of each trigger type.		Create Trigger						
No trigger is available at the moment. Pods Access Events	Click "Create Trigger" in the upper-right corner. Horizontal Pod Autoscaler								
Name	Name Status Image								
nginx-884c7fc54-rh26t	nginx-844/7054-fr38k Profeshing nginceleter								
nginx-884c7fc54-tc9rc	nginx-884/7/534:tShr oginx-iatest								

9. 左側のナビゲーションウィンドウから [アプリケーション] > [Ingress] をクリックする と、Ingress リストの下にルールが表示されます。

Container Service - Kubernetes 🗸	Ingress				Refresh	reate
Overview	Clusters k8s-test	 Namespace 	default 🔻			
Clusters	Name	Endpoint	Rule	Time Created		Action
▼ Application	nginx-ingress		foo.bar.com/ -> nginx-sv	rc 10/10/2018,22:12:43	Details Update View YAML	Delete
Deployment						
StatefulSet						
Pods 😇						
Service						Cont
Ingress						ad Us
Volumes Claim						

10.ブラウザから Ingress のテスト用ドメインヘアクセスすると、Nginx の "welcome" ページ

が表示されます。

foo.bar.com/?spm=5176.	2020520152.0.0.704061b1K4UgO
	Welcome to nginx!
	If you see this page, the nginx web server is successfully installed and working. Further configuration is required.
	For online documentation and support please refer to <u>nginx.org</u> . Commercial support is available at <u>nginx.com</u> .
	Thank you for using nginx.

2.3 **依存関係をベースにした** WordPress アプリケーションのデプロ イ

- Kubernetes の作成が必要です。詳しい情報は、Kubernetes クラスターのスピーディーな作 成をご参照ください。
- ストレージボリュームおよびストレージボリューム要求の作成が必要です。ストレージボ リュームの作成方法は#unique_21、#unique_22および#unique_23をご参照ください。ストレー ジボリューム要求の作成方法は、#unique_24を参照してください。ストレージボリュームとし て Alibaba Cloud ディスクを使用します。この例では、ストレージボリュームのマウント に PV/PVC を選択します。2つのストレージボリューム要求: "wordpress-pv-claim" およ び "wordpress-mysql-pv-claim" を作成します。これらは、それぞれ "wordpress" yaml

ファイルおよび "wordpress-mysql" yaml ファイルで対応するストレージボリュームのマウ ントに使用されます。

Volumes Claims						Refrest	Create
Clusters k8s-cluster • Namespace	default	۲					
Container Service will optimize the Please contact the main account in	e security policy ir n time to use the	the near future, prohibiting "Sub-account Authorization"	unauthorized si function to com	ub-accounts from accessing cluster r plete the cluster resource authorizat	esources. tion.		
Name	Capacity	Access Mode	Status	Storage Class Name	Relate Volume	Time Created	Action
wordpress-mysql-pv-claim	20Gi	ReadWriteOnce	Bound	disk	in the second	08/28/2018,16:58:13	Delete
wordpress-pv-claim	20Gi	ReadWriteOnce	Bound	disk		08/28/2018,16:58:00	Delete

この例では、オーケストレーションテンプレート上のテンプレートをカスタマイズし、依存関係 をベースにしたアプリケーションの作成方法を紹介します。

主要なコンポーネント:

- wordpress
- \cdot mysql

含まれるリソース:

- ・ ストレージボリューム
- ・シークレット
- ・サービス
- 1. Container Service コンソールにログインします。
- 事前に用意したストレージボリューム要求を使用します。2つのストレージボリューム要求、"wordpress-pv-claim" および "wordpress-mysql-pv-claim" を作成します。これらは、それぞれ "wordpress yaml" ファイルおよび "wordpress-mysql yaml" ファイルで対応するストレージボリュームのマウントに使用されます。

 左側のナビゲーションウィンドウから [アプリケーション] > [シークレット] をクリックします。クラスターおよび名前空間を選択し、右上角にある [作成] をクリックします。 作成過程 について詳しくは『シークレットの作成』をご参照ください。

Container Service - Kubernetes -		Secret					Refre	sh	(Create
Overview	Â	Clusters k8s-cluster 🔻 Namespace defau	t • 3							4
 Clusters 	L	Name	Туре	Namespace	Time Created					Action
Application	L	batchrelease-test-mia-svc.secret	Opaque	default	09/05/2018,14:53:06	Detail	I	Edit		Delete
Deployment	L	batchrelease-test-mia.35139.secret	Opaque	default	09/05/2018,14:54:24	Detail	I	Edit		Delete
Pods		batchrelease-test-mia.secret	Opaque	default	09/05/2018,14:53:06	Detail	I	Edit		Delete
Service	1	default-token-kgkh8	kubernetes.io/service-account-token	default	08/24/2018,10:03:28	Detail	I	Edit		Delete
Volumes Claim	L	mysql-pass	Opaque	default	08/28/2018,17:02:45	Detail	I	Edit		Delete
Helm	L									
Release										_
Config Maps										ē
Secret 2										

MySQL データベースの作成およびアクセスにユーザー名とパスワードが必要なため、ユー ザー名とパスワードを管理するためにシークレットを作成します。

シークレットを使用する前に、暗号化に必要なシークレットを作成します。 この例では、 MySQL のルートパスワードはシークレットとして作成され、シークレット名は "mysqlpass" とします。 このシークレットは "wordpress" yaml ファイルおよび "wordpressmysql" yaml ファイルで使用されます。

Namespace	default											
* Name	mysq	l-pass										
	Name must consist of lowercase alphanumeric characters, '' or '.'. Name cannot be empty.											
* Data	0	Name	Value									
	•	password-wordpress	WORDPRESS_D8_PASSWORD									
	•	password-mysql	MYSQL_ROOT_PASSWORD									
	Names	Names can only contain numbers, letters, ", "-" and "."										
	🗷 End	code data values using Base64										
			OK Cancel									

左側のナビゲーションウィンドウから [アプリケーション] > [デプロイ] をクリックし、右上角の [テンプレートで作成] をクリックします。

Container Service - Kubernetes 🗸	Deployment			Refresh Create by image	Create by template
Overview	Clusters k8s-cluster 🔻 Namespace default	• 2			3
 Clusters 	Name	Tag	PodsQuantity	Time Created	Action
Clusters Nodes Volumes	ack-springcloud-eureka-default-ack-springcloud- eureka-0	apprack-springcloud-eureka-default-ack-springcloud-eureka appInstance:ack-springcloud-eureka-default-ack-springcloud- eureka-0 chart:ack-springcloud-eureka-0.1.0 release:ack-springcloud-eureka-default hentage:Tiller	1/1	09/05/2018,17:36:54	Details Edit Scale Monitor More -
Application Deployment	adi:springdoud-eureka-default-adi-springdoud- eureka-1	apprack-springcloud-eureka-default-ack-springcloud-eureka appfinstance:ack-springcloud-eureka-default-ack-springcloud- eurelaa-1 chart:ack-springcloud-eureka-0.1.0 release:ack-springcloud-eureka-default hertage:Tiller	1/1	09/05/2018,17:36:54	Details Edit Scale Monitor More→

クラスターおよび名前空間を選択します。 WordPress のデプロイの作成に関する yaml ファ イルは以下のようになります。

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: wordpress
  labels:
    app: wordpress
spec:
  selector:
    matchLabels:
      app: wordpress
      tier: frontend
  strategy:
    type: Recreate
  template:
    metadata:
      labels:
        app: wordpress
        tier: frontend
   spec:
      containers:
      - image: wordpress:4
        name: wordpress
        env:
        - name: WORDPRESS DB HOST
          value: wordpress-mysql #アクセスする MySQL を指す名前を使用しま
    MySQL サーバー名に対応した名前です。
す。
        - name: WORDPRESS_DB_PASSWORD
          valueFrom:
            secretKeyRef:
              name: mysql-pass
              key: password-wordpress
        ports:
        - containerPort: 80
          name: wordpress
        volumeMounts:
        - name: wordpress-pvc
          mountPath: /var/www/html
      volumes:
      - name: wordpress-pvc
        persistentVolumeClaim:
```

```
claimName: wordpress-pv-claim
```

MySQL デプロイの作成のための yaml ファイルは以下のようになります。

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: wordpress-mysql
  labels:
    app: wordpress
spec:
  selector:
    matchLabels:
      app: wordpress
      tier: mysql
  strategy:
    type: Recreate
  template:
    metadata:
      labels:
        app: wordpress
        tier: mysql
    spec:
      containers:
      - image: mysql:5.6
        name: mysql
        env:
        - name: MYSQL_ROOT_PASSWORD
          valueFrom:
            secretKeyRef:
              name: mysql-pass
              key: password-mysql
        ports:
        - containerPort: 3306
          name: mysql
        volumeMounts:
        - name: wordpress-mysql-pvc
          mountPath: /var/lib/mysql
      volumes:
      - name: wordpress-mysql-pvc
        persistentVolumeClaim:
          claimName: wordpress-mysql-pv-claim
```

5. WordPress への外部からのアクセスを可能にするため、WordPress サービスにより公開されているアクセス方法を作成する必要があります。この例では、Container Service により、外部からのアクセスを提供する Alibaba Cloud Server Load Balancer の自動生成のため、LoadBalancer タイプの WordPress サービスを作成します。

WordPress mysql 上で作成された WordPress デプロイのアクセスを有効化するため、 WordPress mysql に対し サービス名を "Wordpress-mysql" としたサービスを作成しま す。 WordPress に関しては、内部からのみ MySQL が呼び出されるため、MySQL に対する LoadBalancer タイプのサービスの作成は必要ありません。

サービスの作成については、『サービスの作成』をご参照ください。

WordPress サービスおよび MySQL サービスの作成に使われる yaml ファイルは以下のよう になります。

```
apiVersion: v1
kind: Service
metadata:
  name: wordpress
  labels:
    app: wordpress
spec:
  ports:
    -port: 80
  selector:
    app: wordpress
    tier: frontend
  type: LoadBalancer
apiVersion: v1
kind: Service
metadata:
  name: wordpress-mysql
  labels:
    app: wordpress
spec:
  ports:
    - port: 3306
  selector:
    app: wordpress
    tier: mysql
        clusterIP: None
```

 デプロイ完了後、左側のナビゲーションウィンドウより [アプリケーション] > [サービス] をク リックします。WordPress サービスを検索し、『外部エンドポイント』をご参照ください。

Container Service - Kubernetes 👻		Servio	e List							Refresh	Create
Overview	•	Clusters	k8s-cluster 🔻	Namespace defa	ult 🔻 3						
 Clusters 		0	Container Servic Please contact t	e will optimize the sec he main account in tim	urity policy in the near future, pro e to use the "Sub-account Authori	hibiting unauthorized s zation" function to com	ub-accounts from accessing cluster resou plete the cluster resource authorization.	rces.			
Clusters		Name		Туре	Time Created	ClustersIP	InternalEndpoint	ExternalEndpoint			Action
Volumes	L	kubern	etes	ClusterIP	08/24/2018,10:02:37	172.21.0.1	kubernetes:443 TCP		Details Update	View YAML	Delete
Namespace	L	wordp	ess	LoadBalancer	08/28/2018,17:09:06	172.21.8.237	wordpress:80 TCP wordpress:30465 TCP	:80	Details Update	View YAML	Delete
Application	L	wordp	ess-mysql	ClusterIP	08/28/2018,17:09:06	None	wordpress-mysql:3306 TCP		Details Update	View YAML	Delete
Deployment	Ľ										
Pods											
Service 2	-										

 ブラウザーから WordPress サービスの外部エンドポイントにアクセスします。Server Load Balancer により提供された IP アドレスから WordPress アプリケーションにアクセ スすることができます。



WordPress アプリケーションの設定中は、シークレットで設定したパスワードによりアプリ ケーションにログインできます。 さらに、WordPress アプリケーションの属するコンテナーに より生成されたデータはデータストレージボリュームに保存されます。

3 高度な操作

3.1 Helm を使用したマイクロサービスアプリケーションのデプロ イ

このドキュメントでは、Alibaba Cloud Container Service for Kubernetes (ACK) によって 作成された Kubernetes クラスターに、マイクロサービスアプリケーションをデプロイするため に Helm を使用できる 2 つの方法について説明します。 ここでは、PiggyMetrics アプリケー ションをマイクロサービスアプリケーションとしてデプロイします。

デプロイ方法の概要

Helm を使用してマイクロサービスアプリケーションをデプロイするには、次の2つの方法を使用できます。

- 方法 1: Kubernetes クラスターに、マイクロサービスアプリケーションと必要な基本コン ポーネントをデプロイする
- ・方法 2: 基本的な Spring Cloud コンポーネントを含む Kubernetes クラスターに、マイク ロサービスアプリケーションをデプロイする

方法 1: Kubernetes クラスターに、マイクロサービスアプリケーションと必要な基本コンポーネントを デプロイする

前提条件

- Kubernetes クラスターが ACK で作成されている。詳細は、「#unique_3」をご参照ください。
- kubectl を使用して Kubernetes クラスターにアクセスできる。詳細は、「#unique_27」を ご参照ください。
- Kompose と Helm がローカルホストにインストールされている。詳細については、「
 Kompose」、「*Helm*」をご参照ください。

概要

PiggyMetrics アプリケーションは、次の 2 つの **Docker compose** ファイルで定義されてい ます。docker-compose.yml、および docker-compose.dev.yml。 したがって、このよ うなアプリケーションを ACK によって作成された **Kubernetes** クラスターにデプロイするに は、**Kompose** を使用して 2 つの **Docker compose** ファイルを **Kubernetes** 設定ファイルに 変換する必要があります。 *docker-compose、*および *docker-compose.dev* で、**PiggyMetrics** アプリケーションの **Docker compose** ファイルを取得できます。

手順

- 1. PiggyMetrics アプリケーションの Docker compose ファイルを変更します。
 - a. これら2つの Docker compose ファイルのバージョンを2.1から2に変更します。

```
🗎 注:
```

Kompose では、V2.1 の Docker compose ファイルを変換できません。

b. *docker-compose.yml* ファイルで、**Kompose** でサポートされていない次のフィールド を検索します。

```
depends_on:
    config:
        condition: service_healthy #condition is not supported
```

c. docker-compose.yml ファイルで、前の手順のフィールドを以下の手順に置き換えま

す。

```
depends_on:
    - config
labels:
    kompose.service.type: loadbalancer
```

d. *docker-compose.dev.yml* ファイルで、**PiggyMetrics** アプリケーションで使用される 4 つの **MongoDB** データベースの外部ポートを 27017 に変更します。

```
注:
このドキュメントの PiggyMetrics アプリケーションには、4 つの MongoDB データ
ベースが含まれています。これらは、次の4 つの対応するフィールドによって定義され
ています。auth-mongodb、account-mongodb、statistics-mongodb、および
notification-mongodb。
```

たとえば、auth-mongodb フィールドは、次のように変更する必要があります。

```
auth-mongodb:
    build: mongodb
    ports:
```

- 27017:27017

- **2. Kompose** を使用して **Kubernetes** 設定ファイルを生成し、**PiggyMetrics** アプリケーショ ンをデプロイします。
 - a. 次のコマンドを実行して、アプリケーションをデプロイするために必要な環境を設定しま す。

export NOTIFICATION_SERVICE_PASSWORD=passw0rd export CONFIG_SERVICE_PASSWORD=passw0rd export STATISTICS_SERVICE_PASSWORD=passw0rd export ACCOUNT_SERVICE_PASSWORD=passw0rd export MONGODB_PASSWORD=passw0rd

b. 次のコマンドを実行して、**Docker compose** ファイルを **Kubernetes** 設定ファイルに変換します。

kompose convert -f docker-compose.yml -f docker-compose.dev.yml -o
piggymetrics -c

3. helm install コマンドを実行して、対象となる **Kubernetes** クラスターに **PiggyMetrics** アプリケーションをデプロイします。

たとえば、次のコマンドを実行して、piggy という名前のアプリケーションを pm 名前空間に デプロイします。

helm install --namespace pm --name piggy piggymetrics/

4. ローカル Helm クライアントのバージョンがリモート Helm サーバーのバージョンと同じで あることを確認します。

ローカル Helm クライアントとリモート Helm サーバーのバージョンが一致しない場合は、 次のエラーメッセージが表示されます。

Error: incompatible versions client[v2.14.1] server[v2.11.0]

このエラーメッセージが表示されない場合は、この手順を無視できます。

MAC OS を使用する場合は、次のコマンドを実行して、Helm クライアントのバージョン
 2.11.0 を取得します。

brew unlink kubernetes-helm

brew install https://raw.githubusercontent.com/Homebrew/homebrewcore/ee94af74778e48ae103a9fb080e26a6a2f62d32c/Formula/kuberneteshelm.rb

- Windows または Linux OS を使用する場合は、対応するバージョンの Helm クライアン トをインストールします。
- 5. gateway サービスの IP アドレスを使用して、デプロイされたアプリケーションにアクセスします。

en l

注:

gateway サービスは、ターゲットアプリケーションによって提供されるサービスの1つで す。

6. helm delete --purge コマンドを使用して、デプロイされたアプリケーションを削除しま す。

ここでは、 piggy という名前の PiggyMetrics アプリケーションを削除するために、以下の コマンドを実行します。

helm delete --purge piggy

方法 2: 基本的な Spring Cloud コンポーネントを含む Kubernetes クラスターにアプリケーションをデ プロイする

シナリオ1の手順では、すべての基本コンポーネント (Eureka、Zuul、ConfigServer

、Hystrix Dashboard)、およびサービスアプリケーション (gateway、notification、 statistics) を 1 つの helm チャートとしてデプロイする方法を解説しています。 実際には、よ り一般的な状況では、クラスター上に既に Eureka などの基本コンポーネントが存在していま す。 必要なのは、業務アプリケーションのデプロイ、アップグレードおよび管理のみです。

Alibaba Cloud Container Service for Kubernetes で Spring Cloud コンポーネントを使 用するには、[ストア] > [アプリのカタログ] を選択します。

App Catalog			
Øo	Øo	Øo	Øo
ack-hyperledger-fabric 1.1.0 incubator	ack-istio 1.0.0 incubator	ack-istio-remote 1.0.0 incubator	ack-openmpi 3.1.0 incubator
₿ <mark>0</mark>	Øo	Øo	Øo
ack-springcloud-configserver 1.5.13.RELEASE incubator	ack-springcloud-eureka 1.5.13.RELEASE incubator	ack-springcloud-hystrix 1.5.13.RELEASE incubator	ack-springcloud-turbine 1.5.13.RELEASE incubator
₿ <mark>0</mark>	Øo	Øo	Øo
ack-springcloud-zipkin 1.5.13.RELEASE incubator	ack-springcloud-zuul 1.5.13.RELEASE incubator	ack-tensorflow-dev 1.5.0 incubator	ack-tensorflow-serving 1.4.0 incubator

アプリのカタログで Eureka サービスをデプロイします。 ack-springcloud-eureka コンポー

ネントをクリックします。

ack-springcloud-eureka Incubator Spring Cloud Eureka Helm chart for Kubernetes on Alibaba Cloud Container Service	
Readme Values	Danlau
Spring Cloud Notflix Euroka	Берюу
Spring Cloud Nethix Edleka	Only Kubernetes versions 1.8.4 and above are supported.
Eureka	For clusters of version 1.8.1, you can perform "upgrade cluster" operation in the cluster list
	Clusters
Eureka is service discovery server in Spring Cloud Netflix .	k8s-test 🔻
Introduction	Namespace
	default 🔻
This chart bootstraps a two node Eureka deployment on a Kubernetes cluster using the Helm package manager.	Release Name
Installing the Chart	ack-springcloud-eureka-default
To install the chart with the release name myeureka :	DEPLOY
<pre>\$ helm installname myeureka incubator/ack-springcloud-eureka</pre>	

設定の参照または変更には、[値] をクリックします。

Ack-springcloud-eureka Incubator Spring Cloud Eureka Helm chart for Kubernetes on Alibaba Cloud Container Service	
Readme Values 1 repolicaCount: 2 2 inage: 3 repolicaCount: 2 5 pullPolicy: nlways 6 service: 7 enabled: true 8 type: LoadBalancer 9 externalPort: 8761 10 internalPort: 8761 11 sanagement: 12 endpointsEnabled: true	Deploy Only Kubernetes versions 1.8.4 and above are supported. For clusters of version 1.8.1, you can perform "upgrade cluster" operation in the cluster list Clusters kBs-test Namespace default V
	Release Name ack-springcloud-eureka-default DEPLOY

[デプロイ]をクリックします。

左側のナビゲーションウィンドウで、[ディスカバリとロードバランシング] > [サービス] を選 択します。次に、対象となるクラスターと名前空間を選択します。 **EurekaServer** には **2** つ の例があることがわかります。 公開されているサービスのアドレスは、 ack-springcloudeureka-default-ack-springcloud-eureka-svc です。

Container Service - Kubernetes +		Service List						Refresh	Create
Overview	1	Clusters k8s-test • Namespace de	efault 🔹	3					
 Clusters 	Ľ	Name	Туре	Time Created	ClustersIP	InternalEndpoint	ExternalEndpoint		Action
Clusters Nodes	l	ack-springcloud-eureka-default-ack- springcloud-eureka-svc	LoadBalancer	09/04/2018,14:01:16		ack-springcloud-eureka-default-ack- springcloud-eureka-svc:8761 TCP ack-springcloud-eureka-default-ack- springcloud-eureka-svc:30689 TCP		Details View YAML	Update Delete
Volumes Namespace	12	ack-springcloud-eureka-default-ack- springcloud-eureka-svc-0	ClusterIP	09/04/2018,14:01:16	1000.00	ack-springcloud-eureka-default-ack- springcloud-eureka-svc-0:8761 TCP		Details View YAML	Update Delete
Authorization		ack-springcloud-eureka-default-ack- springcloud-eureka-svc-1	ClusterIP	09/04/2018,14:01:16	10,000,000	ack-springcloud-eureka-default-ack- springcloud-eureka-svc-1:8761 TCP	-	Details View YAML	Update Delete
Deployment		batchrelease-01-batch-svc	LoadBalancer	09/03/2018,16:00:56	10.00	batchrelease-01-batch-svc:80 TCP batchrelease-01-batch-svc:32226 TCP	1.1.10.104	Details View YAML	Update Delete
Pods Service 2		kubernetes	ClusterIP	08/22/2018,17:28:51	10.000	kubernetes:443 TCP		Details View YAML	Update Delete
Ingress									

PiggyMetrics において、起動時にすべてのコンテナーが自動的にアクセスする EUREKA サー ビスを registry と呼びます。一般的に、EUREKA サービス名は、イメージ内でパラメーター として渡されます。 このシナリオでは、コードやイメージの変更を行いません。 そのためもう 1 つの方法である、**"registry"** と呼ばれるサービスで Eureka を公開します。

Container Service を使用して、以下の yaml ファイルをデプロイします。

```
    注:
    アプリのカタログデプロイの間にリリース名を変更した場合、同様の変更を以下の例でも行います。
    apiVersion: v1
kind: Service
metadata:
name: registry
spec:
```

```
type: LoadBalancer
ports:
    - port: 8761
    targetPort: 8761
selector:
    app: ack-springcloud-eureka-default-ack-springcloud-eureka
```

release: ack-springcloud-eureka-default

kubectl コマンドラインを使ってサービスを作成できます。

```
$ kubectl apply -f registry-svc.yml
```

コンソールインターフェイスから以下のように実行することもできます。

Clusters	k8s-test v	
Namespace	default 🔻	
Resource Type	Custom	
Template	<pre>1 apiVersion: v1 2 kind: Service 3 metadata: 4 name: registry 5 spec: 6 type: LoadBalancer 7 ports: 8</pre>	Add Deployment Deploy with exist template
	Save Template DEPLOY	

左側のナビゲーションウィンドウで、[ディスカバリとロードバランシング] > [サービス] を選択 し、レジストリが作成されていることを確認します。

Service	List								Refresh
Clusters	k8s-test 🔻	Namespace	default	٣					
Name				Туре	Time Created	ClustersIP	InternalEndpoint	ExternalEndpoint	
ack-spri springcl	ingcloud-eureka oud-eureka-svc	-default-ack-		LoadBalancer	09/04/2018,14:01:16	10.00	ack-springcloud-eureka-default-ack-springcloud- eureka-svc:8761 TCP ack-springcloud-eureka-default-ack-springcloud- eureka-svc:30689 TCP	****	Details View YAML
ack-spri springd	ingcloud-eureka oud-eureka-svc	-default-ack- :-0		ClusterIP	09/04/2018,14:01:16	10.000	ack-springcloud-eureka-default-ack-springcloud- eureka-svc-0:8761 TCP	-	Details View YAML
ack-spri springd	ngcloud-eureka oud-eureka-svc	-default-ack- -1		ClusterIP	09/04/2018,14:01:16		ack-springcloud-eureka-default-ack-springcloud- eureka-svc-1:8761 TCP		Details View YAML
batchre	lease-01-batch-	svc		LoadBalancer	09/03/2018,16:00:56	10.000	batchrelease-01-batch-svc:80 TCP batchrelease-01-batch-svc:32226 TCP	-	Details View YAML
kuberne	ites			ClusterIP	08/22/2018,17:28:51		kubernetes:443 TCP		Details View YAML
registry]			LoadBalancer	09/04/2018,19:46:48		registry:8761 TCP registry:32394 TCP	10.000	Details View YAML

PiggyMetrics に関する helm チャートのディレクトリを、新しいディレクトリ "piggymetri cs-no-eureka" にコピーします。以下の 2 つのファイルを削除します。

```
templates/registry-deployment.yaml
templates/registry-service.yaml
```

これら 2 つのファイルはそれぞれ Eureka デプロイおよび svc のデプロイに使用されます。 ア プリのカタログから SpringCloud の基本コンポーネントとして新しいレジストリサービスのデ プロイが成功した場合、デプロイを繰り返す必要はありません。

再度 PiggyMetrics をデプロイするには helm コマンドを実行します。

\$ helm install -n piggymetrics piggymetrics-no-eureka/

すべてのサービスが開始された後、レジストリサービスにアクセスすると、すべての

PiggyMetrics サービスが EurekaServer により適切に登録されていることを確認できます。

DS Replicas

ack-springcloud-eureka-default-ack-springcloud-eureka-headless-svc-1.default.svc.cluster.localinglices and the second s

Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
ACCOUNT-SERVICE	n/a (1)	(1)	UP (1) - account-service-7fd4976bfc-4rmmj:account-service:6000
AUTH-SERVICE	n/a (1)	(1)	UP (1) - auth-service-7bdb99b5dc-kfnsv:auth-service:5000
GATEWAY	n/a (1)	(1)	UP (1) - gateway-77857d9c49-dgz6j:gateway:4000
NOTIFICATION-SERVICE	n/a (1)	(1)	UP (1) - notification-service-5d5859d7-sc6wb:notification-service:8000
STATISTICS-SERVICE	n/a (1)	(1)	UP (1) - statistics-service-685fb8dc9f-9kfxh:statistics-service:7000

PiggyMetrics アプリケーションは **EurekaServer** 環境に対しデプロイされています。 GATEWAY にアクセスすると、一般的なログインインターフェイスが表示されます。

3.2 プライベートイメージリポジトリを使用してアプリケーション を作成する

このトピックでは、Container Registry コンソールでプライベートイメージリポジトリを作成 し、Container Service コンソールでこのリポジトリのイメージを使用してアプリケーション を作成する方法について説明します。

手順1:プライベートイメージリポジトリの作成

Container Registry コンソールを初めて使用する場合、まず **Containter Registry** の有効化 を求められます。 [今すぐ有効化] をクリックして、**Container Registry** ログインパスワードを 設定します。 左側のナビゲーションペインで [イメージリポジトリ] をクリックします。 イメージリポジト リを作成するリージョンを選択し、[イメージリポジトリを作成] をクリックします。

Home 📔 China (Ha	ngzhou) 🔻 2				Search	Q Message ⁷⁶ Billing I	Management	Enterprise	More	🛱 Englis	sh 🧕
Container Regi	Repositories						Reset Docker	Login Passwor	d	Create Rep	oository
Repositories 1	All Namespaces $$							Reposito	ry Name	3	٩
Namespace	Repository Name	Namespace	Status	Repository Type	Permissions	Repository Address	Created On				Actions

表示されるダイアログボックスで、名前空間、名前、ダイジェスト、およびタイプを設定します。この例では、リポジトリのタイプをプライベートに設定します。[次へ]をクリックします。

Create Repositor	ry	\times
	12	
F	Repository Info Code Source	
Region	China East 1 (Hangzh 🗸	
* Namespace	\sim	
* Repository	the second particular second sec	
Name	Repository name length: 2-64 characters. The name can contain lowercase English letters numbers and the separators and . (separators cannot be the first or last character)	
* Summary	tomcat	
	Max. 100 characters	
Description		
	Supports Markdown Format	
Repository Type	O Public () Private	
	Next	Cancel

3. [ローカルリポジトリ] を選択し、[イメージリポジトリを作成] をクリックします。

Create Repository							\times
Re	epository Inf	0		Cod	2 le Source		
Code Source	Code You can image re	GitHub use the com pository.	Bitbucket mand line to pu	Private GitLab	Local R	$\langle \rangle$	
			Previo	us Create	Repository	Cano	cel

[イメージリポジトリ]ページで、リポジトリを作成したリージョンと名前空間を選択し、新しく作成されたイメージリポジトリの [操作] 列に表示される [管理] をクリックします。

Home 🏾 🏙 China (Ha	ngzhou) 🔻 🙎				Search	ା ଦ Message ⁷⁶ Billing	Management	Enterprise	More	🗑 Engli	sh 🧕
Container Regi	Repositories						Reset Docker	Login Passwo	rd	Create Rep	ository
Repositories 1	All Namespaces 🗸							Reposit	ory Name		Q
Namespace	Repository Name	Namespace	Status	Repository Type	Permissions	Repository Address	Created On			3	Actions
 Image Hub Search 	tomcat-private	dev-testcs	Normal	Private	Manage	L)	09/05/2018,	14:49:32		Manage	Delete

5. [一般設定] ページには、プライベートイメージリポジトリの使用方法が表示されます。

<	tomcat-private-01 China (Hangzhou) Private Local • Normal			Deploy Application
Details	Details			Modify Settings
Authorization Trigger Tags Sync	Details Repository Name tomcat-private-01 Repository Region China (Hangzhou) Repository Type Private Code Repository None Guide Description 1. Log in to Alibaba Cloud Docker Registry \$ sudo docker loginusername=fangguoliang@containerdoc registry.cm:hangzhou aliyuncs.com Use your Alibaba Cloud account to log in to the registry. Your password is the password set when you subsc You can reset the docker login password on homepage. 2. Pull image from the registry \$ sudo docker pull registry.com:hangzhou.aliyuncs.com/kubernetes=java/toncat-private=01:[tag]	Internet VPC Intranet Summary	registry.cn-hangzhou.aliyuncs.com/kubernetes-ji Copy registry-vpc.cn-hangzhou.aliyuncs.com/kubernet Copy registry-internal.cn-hangzhou.aliyuncs.com/kube Copy tomcat	Modify Settings
	 rush mage u ne registry sudo docker loginusername=fangguliang@containerdoc registry.cn=hangzhou aliyuncs.com sudo docker tag [InageId] registry.cn=hangzhou aliyuncs.com/huberneteriyav/ionestrprivet=0 sudo docker pub registry cn=hangzhou aliyuncs.chuberneteriyav/ionestrprivet=00 sudo docker pub registry cn=hangzhou aliyuncs.com/huberneteriyav/ionestrprivet=00 	1:[tag]		
	Please replace the [Imageld] and [tag] parameters based on your image.			

6. Linux サーバーからイメージリポジトリにログインし、次のコマンドを実行してローカルイ メージをリポジトリにアップロードします。

```
$ sudo docker login --username=abc@aliyun.com
Password
## the password for accessing the image
repository
```

Login	Succeed					
\$ docke	er images	# >	tomaat imaga is	, ucod		
REPOST	ORY	# d	concat finage is	s useu		
SIZE	TAG		IMAGE ID		CREATED	
tomcat 463MB	latest		2d43521f2b1a		6 days ago	
ś cudo	dockor tog	[ImagoId]	rogistry on-hon	azhou	alivunce com/	vvv /

\$ sudo docker tag [ImageId] registry.cn-hangzhou.aliyuncs.com/XXX/ tomcat-private:[Image Version Number] \$ sudo docker push registry.cn-hangzhou.aliyuncs.com/XXX/tomcatprivate:[Image Version Number]

出力結果は以下の通りです。

The push refers to a repository [registry.cn-hangzhou.aliyuncs.com/
XXX/tomcat-private]
9072c7b03a1b: Pushed
f9701cf47c58: Pushed
365c8156ff79: Pushed
2de08d97c2ed: Pushed
6b09c39b2b33: Pushed
4172ffa172a6: Pushed
1dccf0da88f3: Pushed
d2070b14033b: Pushed
63dcf81c7ca7: Pushed
ce6466f43b11: Pushed
719d45669b35: Pushed
3b10514a95be: Pushed
V1: digest: sha256:cded14cf64697961078aedfdf870e704a5227018
8c8194b6f70c778a8289**** size: 2836

7. イメージリポジトリの詳細ページに移動します。 左側のナビゲーションペインで [イメージ バージョン] を選択し、ローカルイメージがリポジトリにアップロードされたことを確認しま

す。

<	tomcat-private China East 1 (Hangzhou) F	Private Local • Normal					Deploy Application
Details	Tags						Refresh
Authorization	Version	Image ID 🚳	Status	Digest 🚳	Image Size 📵	Last Updated	Actions
Webhook Tags	V1	690cb3b9c7d1	 Normal 	-	185.708 MB	09/05/2018, 15:19:20	Security Scan Layers Sync Delete

プライベートリポジトリのログインパスワードを作成する

1. 左側のナビゲーションペインで [設定] > [シークレット] をクリックします。

2. 対象となるクラスターおよび名前空間を選択します。 画面右上の [作成] をクリックします。

Container Service - Kubernetes 👻		Secrets					Refre	sh	Create
Pods	• [Clusters k8s-test v Namespaces default	· 3						4
Volume Claims		Name	Туре	Namespaces	Time Created				Action
Releases		acr-credential-be5ac8be6a88c42ac1d831b85135a585	kubernetes.io/dockerconfigjson	default	04/28/2019,14:17:24	Details		Edit	Delete
 Service Mesh 		acr-credential-c6c63a12436031a13361512e545490fd	kubernetes.io/dockerconfigjson	default	04/28/2019,14:17:24	Details		Edit	Delete
Virtual Services		default-token-8pptm	kubernetes.io/service-account-token	default	04/28/2019,14:09:20	Details		Edit	Delete
 Discovery and Load B 		istio.default	istio.io/key-and-cert	default	04/28/2019,16:58:00	Details		Edit	Delete
Services	4								
Ingresses									
Configuration	L								
Config Maps	L								
Secrets 2									
▼ Store									

3. シークレットを設定し、[OK] をクリックします。

Create Secret	
Namespaces	default
* Name	The second second
	Name must consist of lowercase alphanumeric characters, '-' or '.'. Name cannot be empty.
* Type	Opaque Private Repository Logon Password TLS Certificate
* Docker registry URL	
* Username	
* Password	
	OK Cancel

4. 新しく作成されたログインパスワードは、[シークレット]ページで参照できます。

Container Service - Kubernetes 🔻	Secrets											
Overview	Clusters k8s-test v Namespaces default v											
Clusters	Name	Туре	Namespaces	Time Created				Action				
 Applications 	acr-credential-be5ac8be6a88c42ac1d831b85135a585	kubernetes.io/dockerconfigjson	default	04/28/2019,14:17:24	Details	l Edit		Delete				
 Service Mesh 	acr-credential-c6c63a12436031a13361512e545490fd	kubernetes.io/dockerconfigjson	default	04/28/2019,14:17:24	Details	l Edit		Delete				
 Discovery and Load B 	default-token-8pptm	kubernetes.io/service-account-token	default	04/28/2019,14:09:20	Details	l Edit		Delete				
Configuration Config Maps	image-repository-secret	kubernetes.io/dockerconfigjson	default	05/10/2019,18:12:59	Details	l Edit		Delete				
Secrets	istio.default	istio.io/key-and-cert	default	04/28/2019,16:58:00	Details	Edit		Delete				
▼ Store								S				

また、*#unique_27*によりプライベートリポジトリのログインパスワードを作成することもできます。

プライベートイメージリポジトリを利用してアプリケーションを作成する

- 左側のナビゲーションペインで [アプリケーション管理] > [デプロイメント] を選択し、 [デ プロイメント] ページへ移動します。
- 2. 対象となるクラスターと名前空間を選択し、右上の [テンプレートから作成] をクリックしま す。

Container Service - Kubernetes 🕶		Deployments									efresh Create from Image Create from Template				
Overview	*	S How t	to use private images	🔗 Create	applications	🔗 Schedule a P	od to a specific node	Create a layer	-4 Ingress 🔗 Create a laye	er-7 Ingress 🔗 Configu	ure Pod au	uto scaling	8 Monite	or containe	rs
 Clusters 		& Blue-g	green release	•	Namespace	default	v					Course by			0
Clusters			100 000			derduit						Search by	name		4
Nodes		Nar	ne			Label	PodsQuantity	Image	Created At						Actions
Persistent Volumes		🔲 ngi	nx-deployment			app:nginx	2/2	nginx:1.7.9	07/09/2019,11:28:51		Details	l Edit	Scale	Monitor	More 🗸
Namespace	<u>.</u>		Batch Delete												-
Authorizations															Con
 Application 															tact Us
Deployment															
StatefulSet															

∐注:

[イメージから作成] クリックしてアプリケーションを作成することもできます。 詳細につい ては、「#unique_29」をご参照ください.

3. [サンプルテンプレート] ドロップダウンリストで [カスタム] を選択します。 次のコードをコ ピーして [テンプレート] エリアにペーストし、 [作成] をクリックします。

```
apiVersion: apps/v1beta2 # for versions before 1.8.0 use apps/
v1beta1
kind: Deployment
metadata:
  name: private-image
  nameSpace: default
 labels:
    app: private-image
spec:
  replicas: 1
  selector:
    matchLabels:
      app: private-image
 template:
    metadata:
      labels:
        app: private-image
    spec:
                     - name: private-image
      containers:
        image: registry.cn-hangzhou.aliyuncs.com/xxx/tomcat-private:
latest
        ports:
        - containerPort: 8080
      imagePullSecrets:
```

- name: regsecret

詳細については、「Use a private registry」をご参照ください。