

[MSK クラスターの作成]

MSK は Kinesis と異なり、各 AZ ごとにクラスターを作成する必要があります。クラスターの AZ 配置は自動化されますが、起動時に対応する AZ を指定する必要があります。東京やノースバージニアなど古い AZ を保有する AWS アカウントではエラーになる可能性がありますので、別の Region を使ってさい。

1. MSK のマネージメントコンソールにアクセスします
2. [クラスターの作成]をおします
3. クラスター名に[MSKTutorialCluster]と入力します
4. クイック作成では以下の通りデフォルト VPC と 3 つの Subnet が設定されています。サブネットとセキュリティグループの値をコピーしメモしておきます。その後[クラスターの作成]をおします

VPC	vpc-071ef70c7b103769f (デフォルト)	⊗ いいえ
サブネット	subnet-0e066afd4612c0887 🔗	⊗ いいえ
	subnet-0460b1510deaeb467 🔗	
	subnet-0a510030127e0c698 🔗	

デフォルト VPC 以外を指定する場合は（デフォルト VPC が存在しない場合は）、カスタム作成を行うか、リージョンを変更し作業を行ってください。

[クライアントの作成]

クラスターの作成は 30 分程度かかりますので、その間にクライアントを作成します。MSK クライアントは Kinesis と同様にデータを送出する Producer、データを受け取る Consumer というコンセプトがあります。このハンズオンでは、1 台の EC2 で両方を行いますが、商用環境では勿論、Producer と Consumer は別々の存在になります。また、Lambda はイベントソースとして MSK に対応していますので、Producer がメッセージを送出したイベントをもとに関数を起動することが可能です。

5. EC2 のマネージメントコンソールにアクセスします。（ブラウザの別タブを開きます）
6. [インスタンスを起動]をおします
7. AMI は” Amazon Linux 2 AMI (HVM) - Kernel 5.10, SSD Volume Type”を選択します

- インスタンスタイプは”t2.xlarge”を選び、[次のステップ]をおします
- ネットワークから先程 MSK クラスターを作成した VPC を選択し、パブリック IP 割り当てが有効になっていることを確認します。

ネットワーク ⓘ vpc-071ef70c7b103769f | default (デフォルト) ↓ [新しい VPC の作成](#)
 サブネット ⓘ 優先順位なし (アベイラビリティゾーンのデフォルト) ↓ [新しいサブネットの作成](#)
 自動割り当てパブリック IP ⓘ サブネット設定を使用 (有効) ↓

- [次のステップ]をおします
- ストレージのサイズに 30 を指定します

ボリュームタイプ ⓘ デバイス ⓘ スナップショット ⓘ サイズ (GiB) ⓘ ボリュームタイプ ⓘ IOPS ⓘ スループット (MB/秒) ⓘ
 ルート /dev/xvda snap-06a1c12b2297eeb5a 8 汎用 SSD (gp2) 100 / 3000 該当なし

- [確認と作成]をおし、次の画面で[起動]をおします
- キーペアは”なし”を選んで、[インスタンスの作成]をおします

この AMI は、Amazon Linux 2 のみサポートされています。
 キーペアなしで続行
☐ 私は、キーペアがない場合、EC2 Instance Connect を使用するか、AMI に組み込まれているパスワードを知っている場合にのみ、このインスタンスに接続できることを認識しています。
 EC2 Instance Connect は、Amazon Linux 2 および Ubuntu のみサポートされています。 [詳細はこちらをご覧ください。](#)

- インスタンスが起動した後、インスタンス名をクリックし詳細画面へ移動し、“セキュリティ”タブでセキュリティグループをコピーしメモしておきます。

詳細 **セキュリティ** ネットワーキング ストレージ ステータスチェック モニタリング タグ
 ▼ セキュリティの詳細

IAM ロール -	所有者 ID 294963776963	起動時刻 Wed Mar 16 2022 11:10:47 GMT+0900 (日本標準時)
セキュリティグループ sg-05f1e031dd536c58b (launch-wizard-7)		


 ▼ インバウンドルール

- 左ペインから“セキュリティグループ”をクリックし、先程 MSK クラスター起動時にコピーしたセキュリティグループを特定しクリックします。(EC2 用ではないので注意してください)
- [インバウンドルールの編集]をおします。MSK クラスターに対して入ってくる通信 (Producer である EC2 が出す通信=インバウンド通信) を受け入れる設定を行います。
- [ルールを追加]をおします
- EC2 のクライアントに設定されているセキュリティグループから出る全てのトラフィ

ックを受け付けるように設定します。以下の画面を参考にしてください。



19. [ルールを保存]をおします
20. EC2 マネージメントコンソールから、先程作成した EC2 にチェックを付けます。(インスタンス名はクリックしません)

<input checked="" type="checkbox"/>	MSKTutorialClient 	i-0f97c838f6bace434	 実行中		t2.xlarge
-------------------------------------	---	---------------------	---	---	-----------

21. [接続]ボタンをおします
22. [EC2 Instance Connect]のタブを選んで[接続]をおします



23. 以下のようにコンソールが起動します。

```

  _ |  ( _ | _ )
  _ |  ( _ | _ /
  _ | \ _ | _ |
                                Amazon Linux 2 AMI

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-172-31-8-210 ~]$
[ec2-user@ip-172-31-8-210 ~]$
```

24. (Chrome の場合)ブラウザのタブを右クリックし、グループを Producer として作成しておきます。



25. [sudo yum install java-1.8.0] を実行します。途中確認が求められますので”y”を入力してください
26. [wget https://archive.apache.org/dist/kafka/2.6.2/kafka_2.12-2.6.2.tgz]を実行し、Apache Kafka クライアントライブラリを入手します
27. [tar -xzf kafka_2.12-2.6.2.tgz]を実行し、ダウンロードしたライブラリを解凍します
28. 事前準備はこれで完了です。MSK クラスター作成が完了するまで待ちます。

[MSK の構成]

ブローカータ...	ゾーンあたりのブローカー	ゾーン
kafka.m5.large	1	3

起動時に指定した 3 つの AZ に対してそれぞれ 1 個ずつブローカーと言われるメッセージを取り扱うノードが作成されています。アプリケーションの構成に応じてこれらをスケールさせていく必要があります。本ハンズオンでは割愛しますが興味のある方は以下に挑戦してみてください。

<https://catalog.us-east-1.prod.workshops.aws/workshops/c2b72b6f-666b-4596-b8bc-bafa5dcca741/en-US>

以下の状態になれば MSK クラスターの作成が完了です

クラスター (1)					
<div> <div>Q クラスターを検索</div> <div> <div>アクション ▼</div> <div>クラスターの作成</div> </div> </div> <div> <div>< 1 ></div> </div>					
クラスター名 ▼	状態 ▼	認証 ▼	Apache... ▼	ブローカータ... ▼	ゾ...
<div>○</div> <div>MSKTutorialCluster</div>	<div>🟢</div> <div>アクティブ</div>	Unauthenticated, l...	2.6.2	kafka.m5.large	1

[メッセージのテスト]

ここから、先程設定した EC2 の Producer でメッセージを投入してみます

- MSK クラスターのクリックし、[クライアント情報の表示]をおし、以下 2 つの文字列をコピーしメモしておきます。非常に長い文字列ですが、改行が入らないことに注意してください

認証タイプ	プライベートエンドポイント	パブリックエンドポイント
プレーンテキスト	<div>📄</div> <div>b-2.msktutorialcluste.j73ll5.c4.kafka.a.ap-northeast-1.amazonaws.com:9092, b-3.msktutorialcluste.j73ll5.c4.kafka.ap-northeast-1.amazonaws.com:9092, b-1.msktutorialcluste.j73ll5.c4.kafka.ap-northeast-1.amazonaws.com:9092</div>	-

Apache ZooKeeper 接続

プロデューサーまたはコンシューマーの設定で使用される設定値です。Apache ZooKeeper 接続文字列を指定し、Apache ZooKeeper サーバーのホストおよびポートを一覧表示します。このプロパティ値を Kafka ブローカーの設定で使用してください。これは Apache ZooKeeper ノードへの接続を確立するためにブローカーで使用されるホスト/ポートのペアのリストです。

プレーンテキスト

📄 z-3.msktutorialcluste.j73ll5.c4.kafka.ap-northeast-1.amazonaws.com:2181,z-1.msktutorialcluste.j73ll5.c4.kafka.ap-northeast-1.amazonaws.com:2181,z-2.msktutorialcluste.j73ll5.c4.kafka.ap-northeast-1.amazonaws.com:2181

- EC2 上で、kafka_2.12-2.6.2 ディレクトリに移動します
- [bin/kafka-topics.sh --create --zookeeper **ZookeeperConnectString** --replication-factor 3 --partitions 1 --topic MSKTutorialTopic]
をコピーして実行します。その際”ZookeeperConnectString”を先程コピーしたものに置換してください
- “Created topic MSKTutorialTopic.”と表示されれば成功です。このトピックとは SNS と同じ概念です。複数のメッセージ種別を取り扱う際などのメッセージパイプライン識別子になります。トピックの詳細に興味がある方は”aws pub sub”などで検索してみてください。疎結合モデルの非常に重要なアーキテクチャ設計指針の 1 つです。
- [cd bin]を実行し移動します

34. [vi client.properties]を実行します
35. [security.protocol=PLAINTEXT]を 1 行目にペーストします。
36. 以下の順番にキーを入力します(ファイルを保存します。vi が使える方は気にせず保存してください)
[ESC][:] [w] [q]
37. [./kafka-console-producer.sh --broker-list **BootstrapBrokerString** --producer.config client.properties --topic MSKTutorialTopic]
を実行します。
BootstrapBrokerString は先ほどコピーした“認証タイプ：プレーンテキスト”の文字列置換します。
38. 以下のように、“>”が表示されたら接続ができていますので、メッセージを適当に 3 行ほど入力します。

```
"client.properties" 1L, 28B written
[ec2-user@ip-172-31-8-210 bin]$ ./kafka-console-producer.sh --broker-list b-2.msktutorialcluste.j73ll5.c4.kafka.ap-north
.com:9092,b-3.msktutorialcluste.j73ll5.c4.kafka.ap-northeast-1.amazonaws.com:9092,b-1.msktutorialcluste.j73ll5.c4.kafka.
amazonaws.com:9092 --producer.config client.properties --topic MSKTutorialTopic
>
```

[Consumer の起動]

今までの手順でブローカーにはメッセージが挿入されています。今度はこれを Consumer として呼び出します。

39. 同じ EC2 インスタンスに対してもう 1 回 Instance Connect を実行し、ブラウザのタブ機能でわかりやすい名前を付けておきます。



40. 以下のディレクトリに移動します

```
[ec2-user@ip-172-31-8-210 ~]$ cd kafka_2.12-2.6.2/
[ec2-user@ip-172-31-8-210 kafka_2.12-2.6.2]$ cd bin
```

41. [./kafka-console-consumer.sh --bootstrap-server **BootstrapBrokerStringTls** --consumer.config client.properties --topic MSKTutorialTopic --from-beginning]
を実行します。
BootstrapBrokerStringTls の値は先程と同じく認証：プレーンテキストとしてコピーした文字列に置換します (tls と書いてあるが気にしなくて大丈夫です)
42. 以下のようにメッセージが出てきたら成功です！

```
[ec2-user@ip-172-31-8-210 bin]$ ./kafka-console-consumer.sh --bootstrap-server b-2.msktutorialcluste.j73ll5.c4.kafka.ap-  
onaws.com:9092,b-3.msktutorialcluste.j73ll5.c4.kafka.ap-northeast-1.amazonaws.com:9092,b-1.msktutorialcluste.j73ll5.c4.k  
t-1.amazonaws.com:9092 --consumer.config client.properties --topic MSKTutorialTopic --from-beginning  
test  
hello  
kameda
```

43. 更なるシナリオは以下にありますので興味がある方は挑戦してみてください

<https://catalog.us-east-1.prod.workshops.aws/v2/workshops/c2b72b6f-666b-4596-b8bc-bafa5dcca741/>

おつかれさまでした！

以下を削除してください

- MSK クラスター
- EC2