IT Automation

オールインワン型HA構成(AWS)

インストールマニュアル

※本書では「Exastro IT Automation」を「ITA」として記載します。

ExastroIT Automation Version 1.9

Exastro developer

はじめに

1. 本資料について

・ 本資料は、ITAオールインワンインストール済EC2サーバ2台とOSS製品群を組み合わせて、HAクラスタを構成するのに
必要な情報を記述したものです。ITAサーバをHAクラスタ化することにより、
単一Linuxサーバでは実現できない可用性・信頼性の高いITAサーバを構築することができます。

・ 本資料にもとづいて構成したHAクラスタは、以下の機能を提供します。

  ・ 常にどちらか1台のサーバ上でITAサーバが稼働し、ITAのWeb/AP、Backyard、Ansibleドライバなどの全機能を提供します。
  ・ サービスを提供しているサーバがダウンすると、約2-3分前後で他のサーバ上でITAサーバが起動し、サービスが引き継がれます。
  この時、データベースのデータは失われません。

・ デプロイ例は「5. システム構成例」の図を参照してください。
ユーザ環境に合わせて、IPアドレス、ホスト名、ディレクトリ名、ファイル名を読み替えて構築してください。

・ 本資料は、以下の環境で構築することを前提に作成しています。

主要ソフトウェア(ITAを除く)	用途	検証バージョン
Red Had with Linux 8 High Availability	OS（EC2のイメージ）	8.4
Pacemaker(※)	クラスタ制御	2.0.5-9
Corosync(※)	ハートビート	3.1.0
pcs(※)	Pacemakerコマンド	0.10.8

※この3つは上記のEC2のイメージに同梱されています

2. 責任範囲

・ 本資料は、ITAサーバをクラスタ化する為の注意点や設定例を参考情報として示しています。

はじめに

3. 構築順序

・構築順序は下記のとおりです。

項目	サービス	1号機(Master)	2号機(Slave)
0-1. 事前準備(AWS設定)	VPC	-	-
	サブネット	-	-
	ルートテーブル	-	-
	セキュリティグループ	-	-
	IAM	-	-
	RDS	-	-
	EFS	-	-
	Route53	-	-
	EC2	EC2作成	EC2作成
	ElasticIP	ElasticIP関連付け	-
1. ITAインストール	ITA1.7.2オンラインインストール	ITAオンラインインストール	ITAオンラインインストール
2. Apacheリソース設定	Apache	Apacheステータス確認リソース作成	Apacheステータス確認リソース作成
		Apacheサービス停止・無効化	Apacheサービス停止・無効化
3. ITAサービス停止設定	-	ITAサービス停止・無効化	ITAサービス停止・無効化
4. EFS設定	EFS	EFS使用準備	EFS使用準備
		共有ディレクトリ設定	-
5. RDS設定	RDS	RDSデータ設定	-
		DB接続設定	DB接続設定
		MariaDB停止	MariaDB停止
6. Pacemaker使用準備	-	awscliインストール	awscliインストール
		hostsの設定	hostsの設定
7. Pacemaker設定	Pacemaker	クラスター設定	クラスター設定
		リソース設定	リソース設定
		Pacemaker起動順序設定	Pacemaker起動順序設定
		MariaDBアンインストール	MariaDBアンインストール

はじめに

4. Linux-HAクラスタスタックについて

- Linux-HAクラスタスタックは、サーバシステムの可用性を向上したり、ストレージ故障によるデータ喪失を防止したりすることを目的としたソフトウェア群です。次の2つのソフトウェアおよび関連パッケージによって構成されています。

  Corosync
  　HAクラスタを構成するサーバの正常稼働を相互に監視するソフトウェア。

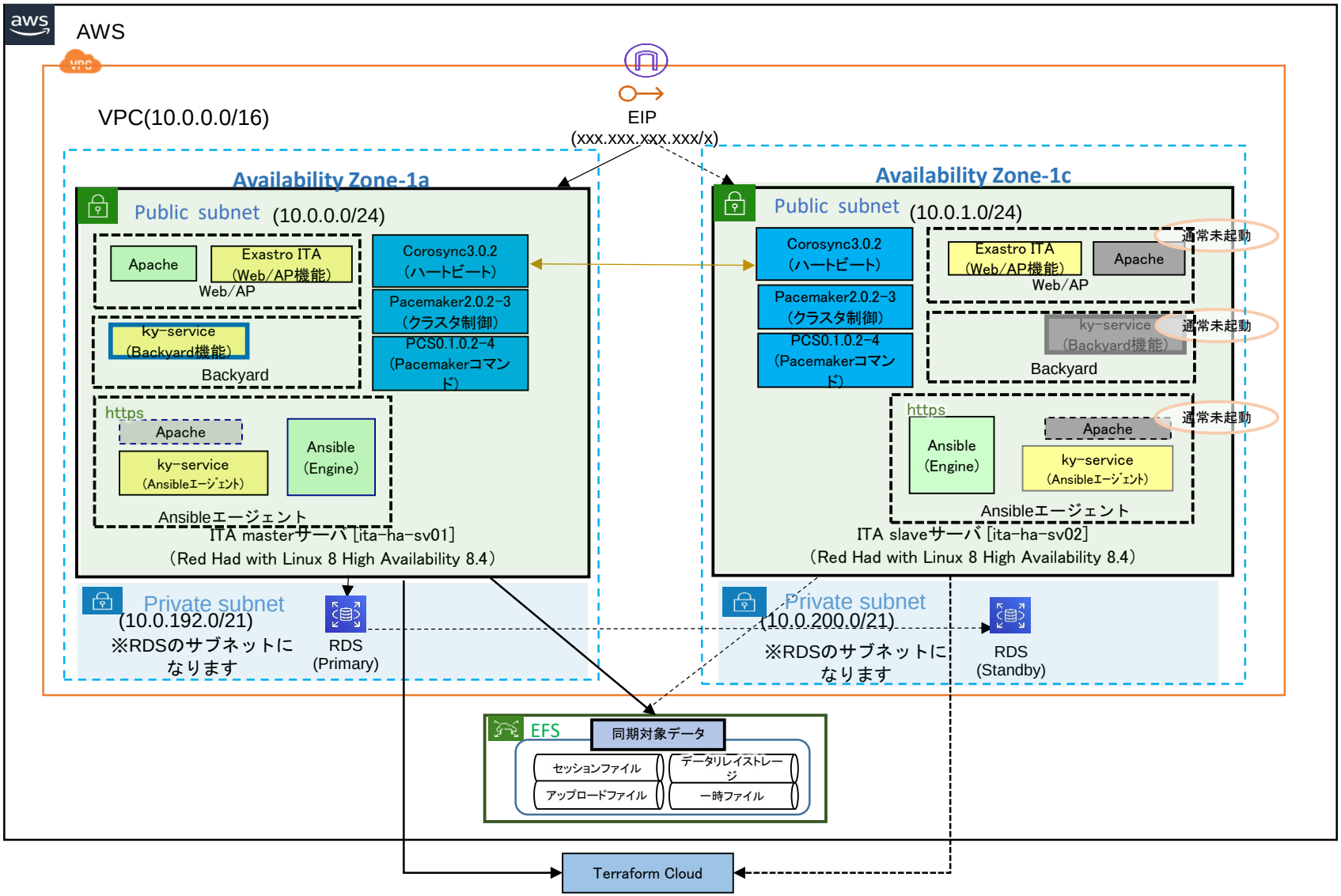
  Pacemaker
  　HAクラスタが提供するサービスを監視し、サービスを提供していたサーバがダウンした場合に他のサーバでサービスを継続させるなどして、クラスタ全体の可用性を保ちます。

  参照URL:
  　http://linux-ha.osdn.jp/wp/

はじめに

5. システム構成例

・ ITAクラスタを構成する2台のサーバのディスクやネットワーク構成、関連プログラムは、下図のようになります。



はじめに

- ネットワーク条件

  - 通常、各ネットワークインタフェースは、それぞれ固有のIPアドレスを持つが、ITAサーバのHAクラスタ構成を実現する為に追加の
    IPアドレス(ElasticIPアドレス)が必要になります。用意したElasticIPアドレスは、構築後、Pacemakerにより自動的にmasterサーバ側に割り当てられます

  - 本資料のネットワーク設定例

ネットワークIF	用途	masterサーバ	slaveサーバ
ElasticIP	ITA接続	自動的に発行されたグローバルIP	
eth0	ローカルIP	10.0.0.204	10.0.1.61

※masterサーバのローカルIPアドレスは、ElasticIPを関連付ける際に設定するローカルIPアドレスになります。

  - 利用クライアントからサーバへ以下の通信ポートが利用可能なこと。

通信種別	ポート番号
http	80/tcp
https	443/tcp

  - クラスタ構成サーバを同一ネットワーク上に構成し、以下の通信ポートが利用可能であること。

通信種別	ポート番号
ssh(scp)	22/tcp
MYSQL/Aurora(RDS)	3306/tcp
NFS	2409/tcp
Pacemaker	2224/tcp
Corosync	5405/udp

※Pacemaker、Corosyncポート参考URL:
https://access.redhat.com/documentation/ja-jp/red_hat_enterprise_linux/7/html/high_availability_add-on_reference/s1-firewalls-haar

はじめに

6. 作業前提

- 各種ソフトウェアのインストールはオンライン環境での手順を記載してます。
  オフラインで実行する場合はソフトウェアの依存関係を構築サーバと合致させた上で、
  事前のライブラリ収集を行ってください。

- 構築手順の前に、以下の基本的なOS設定を実施して下さい。なお、特に断らない限り、OS基本設定は両方のサーバで同一に揃える必要があります。

  - SELinuxの無効化
    SELinuxが有効な状態でHAクラスタを正常に運用するには、SELinuxに関する、極めて高度な知識と経験が必要になります。
    本手順ではSELinuxを無効での検証となっておりますが、無効化はユーザの自己責任で対応ください。

- ITAのインストールについて

  - ITAの動作要件については、以下のドキュメントを参照してください。
    https://exastro-suite.github.io/it-automation-docs/asset/Documents_ja/Exastro-ITA_システム構成／環境構築ガイド_基本編.pdf
    IT Automation BASE システム構成／環境構築ガイド 基本編
    1.1 サーバ動作要件

    https://exastro-suite.github.io/it-automation-docs/asset/Documents_ja/Exastro-ITA_システム構成／環境構築ガイド_Ansible-driver編.pdf
    Ansible-driver システム構成／環境構築ガイド Ansible-driver編
    3. システム要件

はじめに

・ インストール要件

以下URLのオールインワン構成のインストールマニュアル参照のこと。
ITAはオンライン／オフラインのどちらのインストール手順でも問題ありません。
https://exastro-suite.github.io/it-automation-docs/learn_ja.html#deploy

本資料では標準インストールの機能有効化を前提としてます。

ita_answers.txtの機能設定例
ita_base:yes
createparam:yes
hostgroup:yes
ansible_driver:yes
cobbler_driver:no
terraform_driver:yes
cicd_for_iac:no

ITAの起動サービスは、ITAのバージョン、インストールする機能によって変動します。
本資料は、ITA ver1.7.xの標準インストールを想定している為、
上記の想定と異なるバージョンの利用と機能のインストールを行う場合は、
本手順の下記の箇所にの対象サービス内容について、適宜修正が必要となります。
・ 構築手順「3.ITAサービス停止設定」の対象サービス

AWS設定手順書

ここでは事前準備として、AWSの設定手順を説明していきます。
IPアドレスや設定値は任意で変更をお願いいたします。

今回の利用するサービスは下記のとおりです。

利用サービス一覧
1.VPC
2.サブネット
3.インターネットゲートウェイ
4.ルートテーブル
5.セキュリティグループ
6.IAM
7.RDS
8.EFS
9.Route53
10.EC2
11.ElasticIP

AWS設定手順書

1.VPC

1.コンソールにログインし、左上の「サービス」から「VPC」をクリックする

2.「VPC」をクリックする

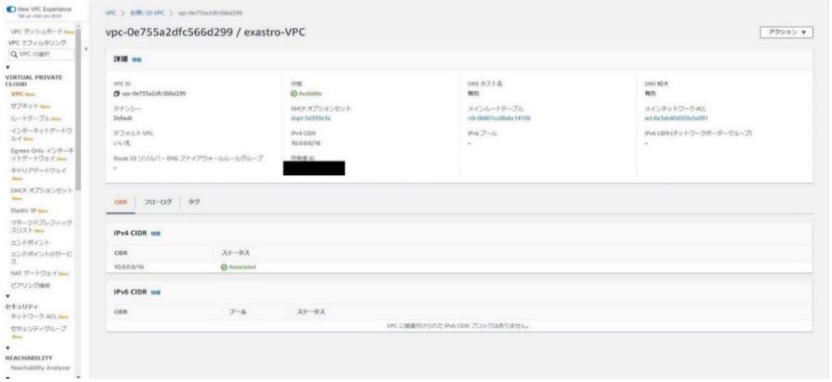
3.「VPCを作成」をクリックする

4.VPCの情報を入力する

入力例）

名前タグ	IPv4 CIDRブロック
exastro-VPC	10.0.0.0/16

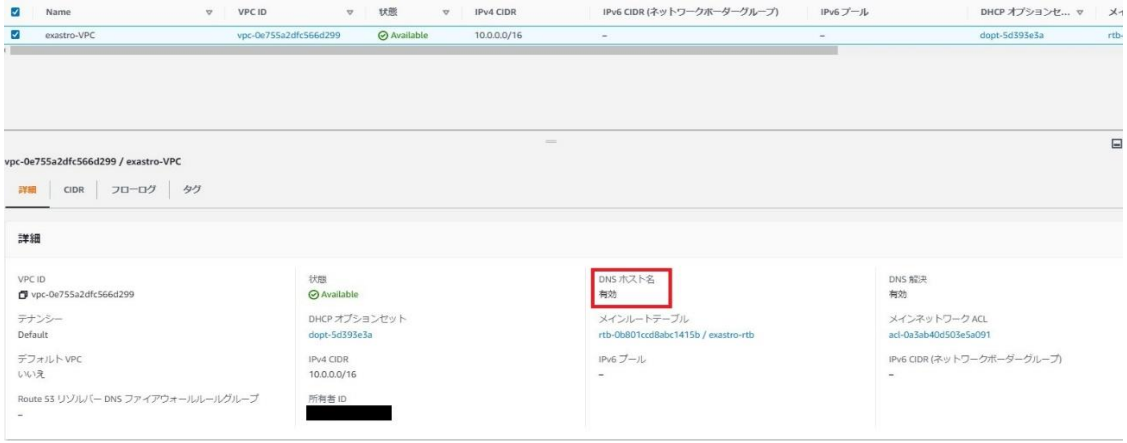
5.VPCが作成できたことを確認する



6.2の画面に戻りアクションから「DNSホスト名を編集」をクリックする

7.DNSホスト名を有効化し、「変更を保存」をクリックする

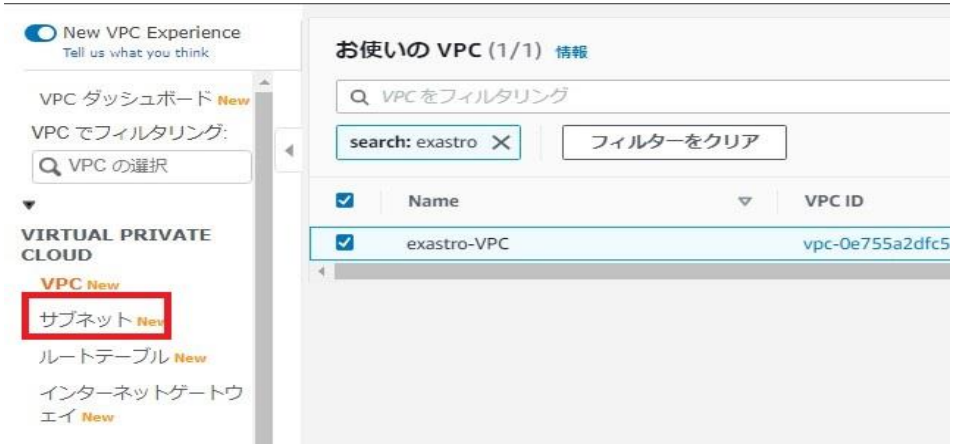
8.DNSホスト名が有効になっていることを確認する



AWS設定手順書

2.サブネット

1.左側のメニューから「サブネット」をクリックする

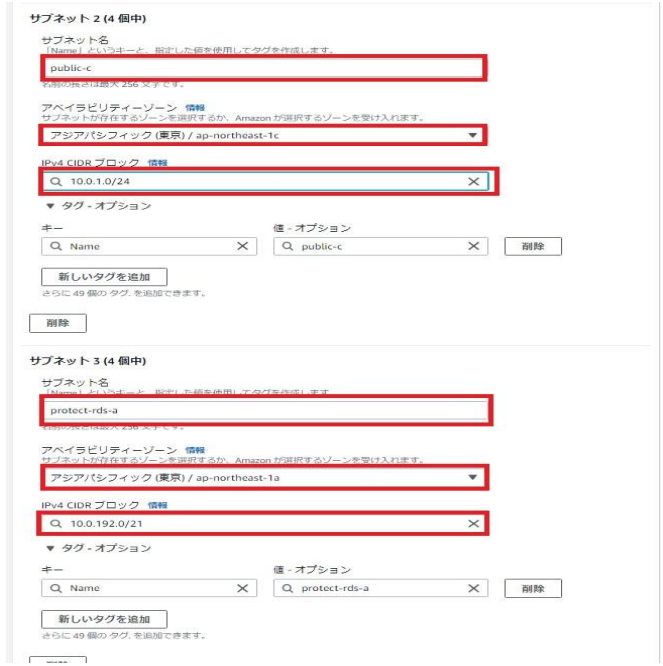


2.「サブネットを作成」をクリックする

3.先ほど作成したVPCをクリックし、サブネットの情報を入力する

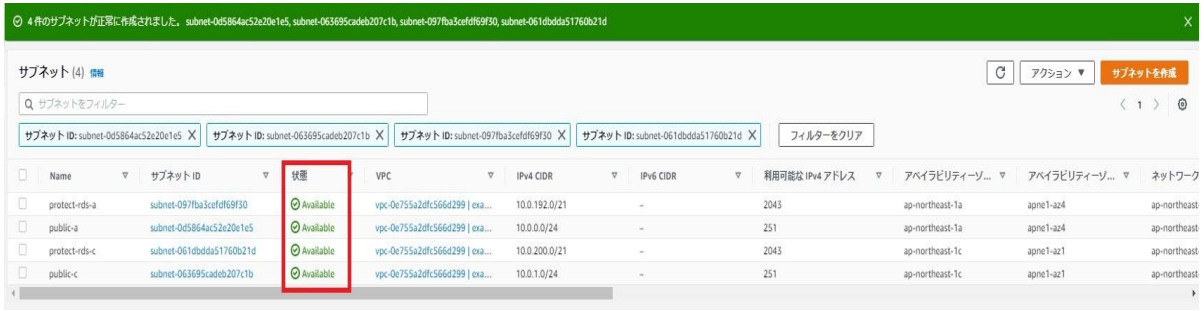
入力例）

サブネット名	アベイラリティーゾーン	Ipv4 CIDRブロック
public-a	アジアパシフィック (東京)/ap-northeast-1	10.0.0.0/24
public-c	アジアパシフィック (東京)/ap-northeast-1	10.0.1.0/24
protect-rds-a	アジアパシフィック (東京)/ap-northeast-1	10.0.192.0/21
protect-rds-c	アジアパシフィック (東京)/ap-northeast-1	10.0.200.0/21


サブネット 2 (4 個中)
サブネット名
public-c
アベイラビリティーゾーン 情報
アジアパシフィック (東京) / ap-northeast-1c
IPv4 CIDR ブロック 情報
10.0.1.0/24
タグ - オプション
キー
Name
値 - オプション
public-c
削除
新しいタグを追加
削除
サブネット 3 (4 個中)
サブネット名
protect-rds-a
アベイラビリティーゾーン 情報
アジアパシフィック (東京) / ap-northeast-1a
IPv4 CIDR ブロック 情報
10.0.192.0/21
タグ - オプション
キー
Name
値 - オプション
protect-rds-a
削除
新しいタグを追加


サブネット 4 (4 個中)
サブネット名
protect-rds-c
アベイラビリティーゾーン 情報
アジアパシフィック (東京) / ap-northeast-1c
IPv4 CIDR ブロック 情報
10.0.200.0/21
タグ - オプション
キー
Name
値 - オプション
protect-rds-c
削除
新しいタグを追加
削除
新しいサブネットを追加
キャンセル
サブネットを作成

4.作成後、状態がAvailableになっていることを確認する



3.インターネットゲートウェイ

1.左側のメニューから「インターネットゲートウェイ」をクリックする

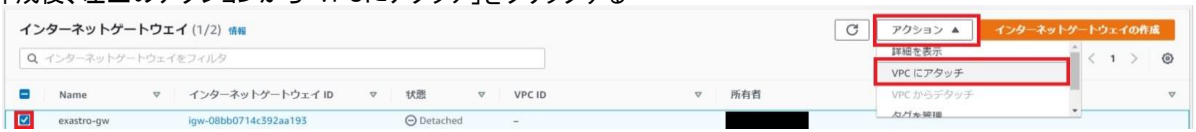
2.「インターネットゲートウェイの作成」をクリックする

3.インターネットゲートウェイの情報を入力する

入力例）

名前タグ	exastro-gw

4.作成後、左上のアクションから「VPCにアタッチ」をクリックする



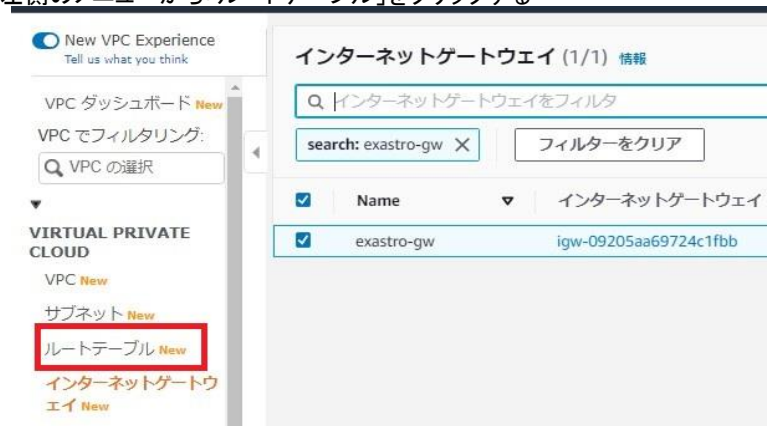
5.作成したVPCを選択し、「インターネットゲートウェイのアタッチ」をクリックする

6.状態がAvailableになっていることを確認する

AWS設定手順書

4.ルートテーブル

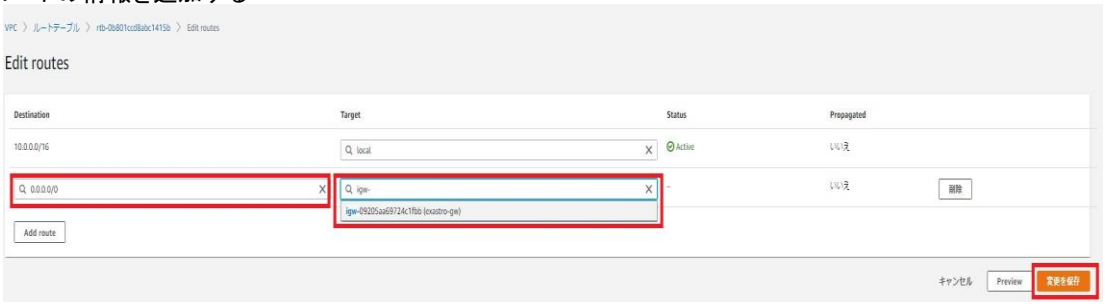
1.左側のメニューから「ルートテーブル」をクリックする



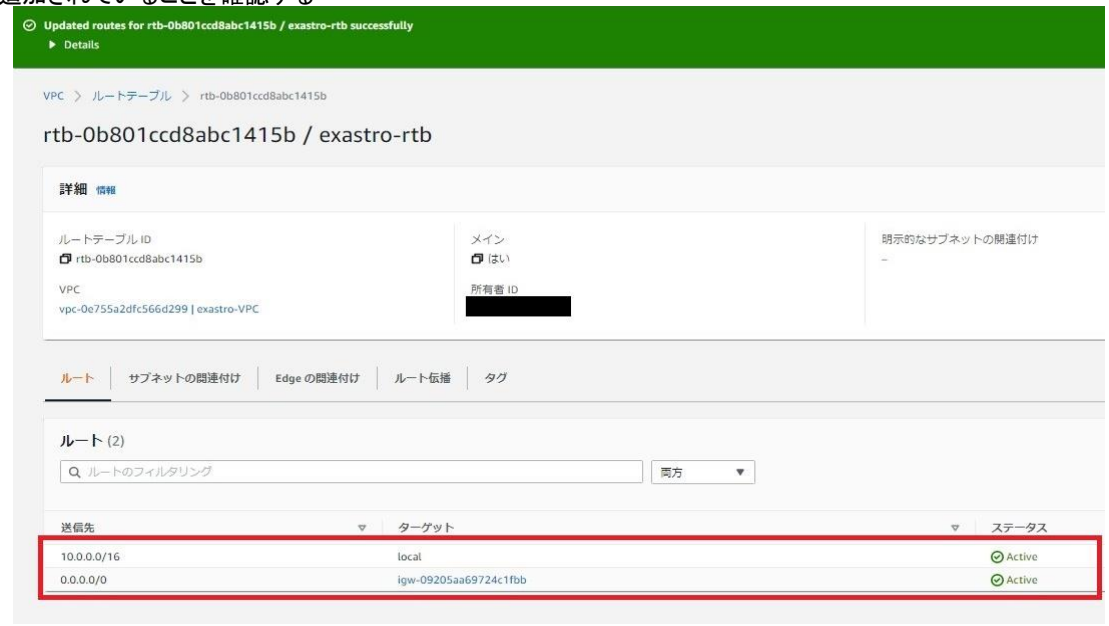
2.作成したVPCのIDで検索ボックスから検索する

3.対象のルートテーブルの下の「ルート」タブをクリックする

4.ルートの情報を追加する



5.追加されていることを確認する



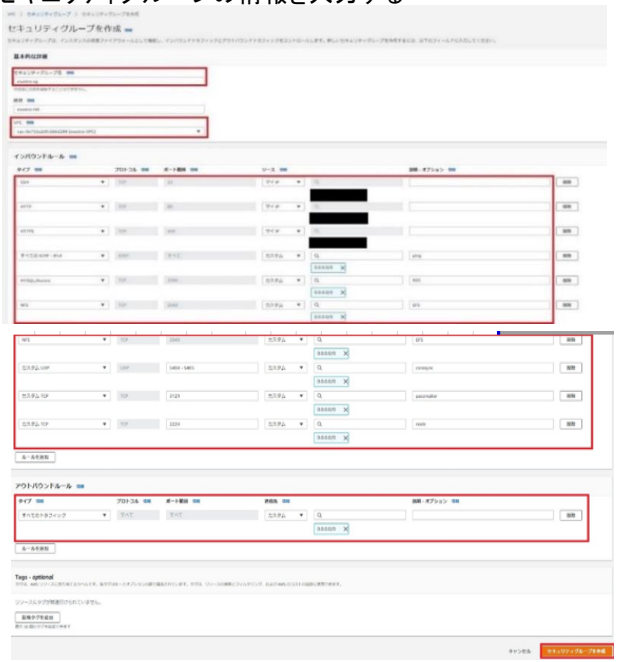
AWS設定手順書

5.セキュリティグループ

1.左側のメニューから「セキュリティグループ」をクリックする

2.「セキュリティグループを作成」をクリックする

3.セキュリティグループの情報を入力する



入力例）

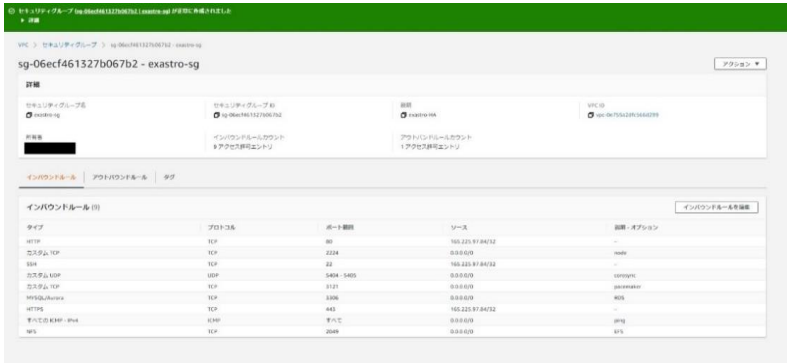
セキュリティグループ名	exastro-sg
VPC	exastro-vpc

インバウンドルール			
タイプ	プロトコル	ポート範囲	ソース
SSH	TCP	22	マイIP
HTTP	TCP	80	マイIP
HTTPS	TCP	443	マイIP
すべてのICMP-IPv4	IPv4	すべて	10.0.0.0/16 ※
カスタムUDP	UDP	5404-5405	10.0.0.0/16 ※
カスタムTCP	TCP	3121	10.0.0.0/16 ※
カスタムTCP	TCP	2224	10.0.0.0/16 ※

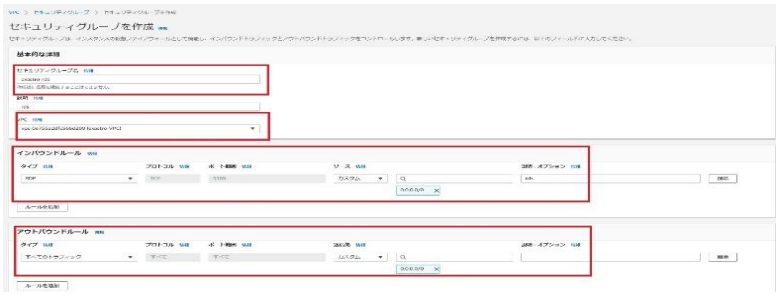
※ VPCのNWアドレス

アウトバウンドルール			
タイプ	プロトコル	ポート範囲	ソース
すべてのトラフィック	すべて	すべて	0.0.0.0/0

4.作成されていることを確認する



5.EFS、RDS用のセキュリティグループを作成する



入力例）

セキュリティグループ名	exastro-efs
VPC	exastro-vpc

インバウンドルール			
タイプ	プロトコル	ポート範囲	ソース
NFS	TCP	2049	10.0.0.0/16 ※

※ VPCのNWアドレス

アウトバウンドルール			
タイプ	プロトコル	ポート範囲	ソース
すべてのトラフィック	すべて	すべて	0.0.0.0/0

入力例）

セキュリティグループ名	exastro-rds
VPC	exastro-vpc

インバウンドルール			
タイプ	プロトコル	ポート範囲	ソース
MYSQL/Aurora	TCP	3306	10.0.0.0/16 ※

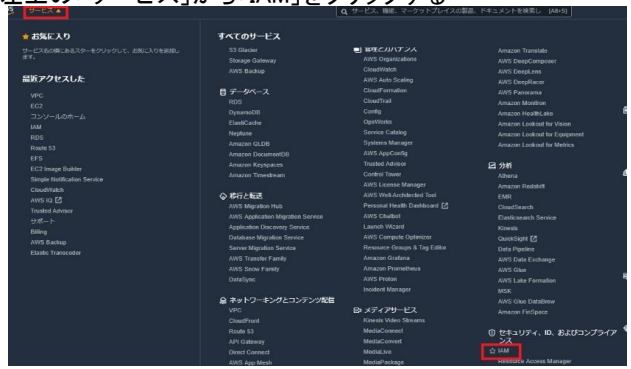
※ VPCのNWアドレス

アウトバウンドルール			
タイプ	プロトコル	ポート範囲	ソース
すべてのトラフィック	すべて	すべて	0.0.0.0/0

AWS設定手順書

6.IAM

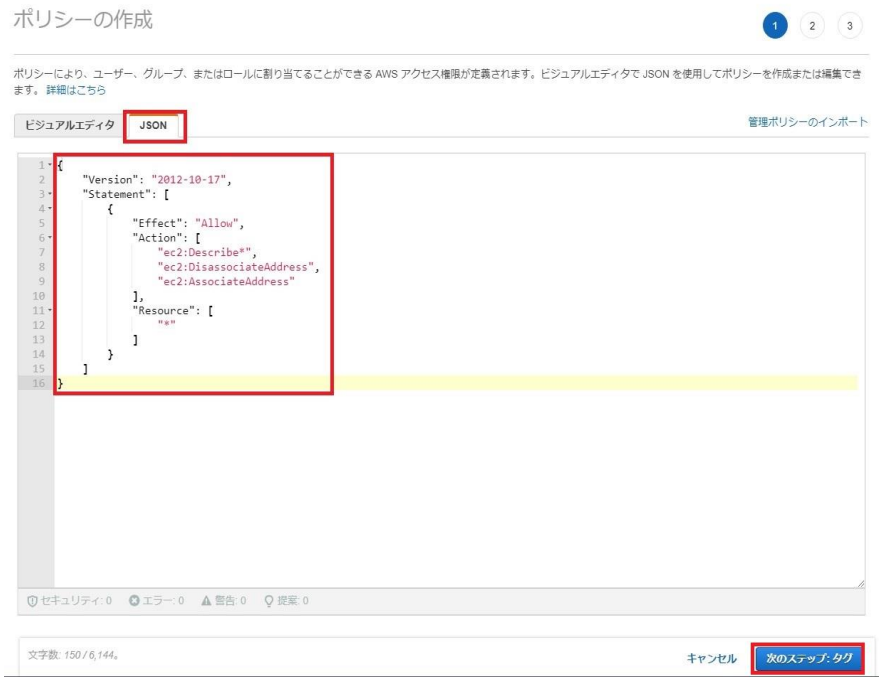
1.左上の「サービス」から「IAM」をクリックする



2.左側のメニューの「ポリシー」をクリックする

3.「ポリシーの作成」をクリックする

4.JSONタブを開いて、ポリシーを入力する



入力例）

ポリシー内容

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:Describe*",
                "ec2:DisassociateAddress",
                "ec2:AssociateAddress"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

5.「次のステップ：確認」をクリックする

6.ポリシー名を入力し、「ポリシーの作成」をクリックする

入力例）

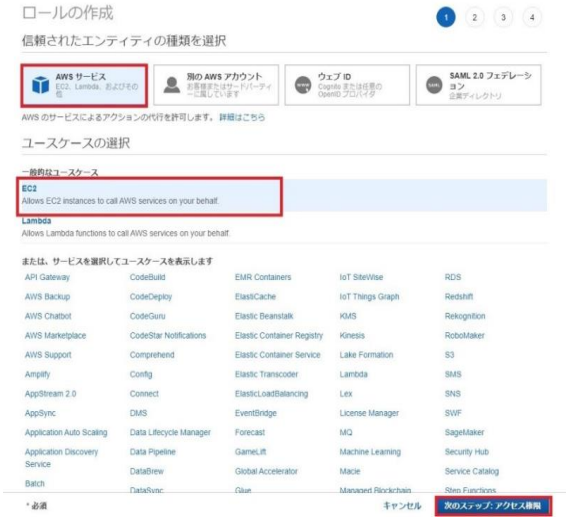
名前
exastro-policy

7.作成されたことを確認する

8.左側のメニューの「ロール」をクリックする

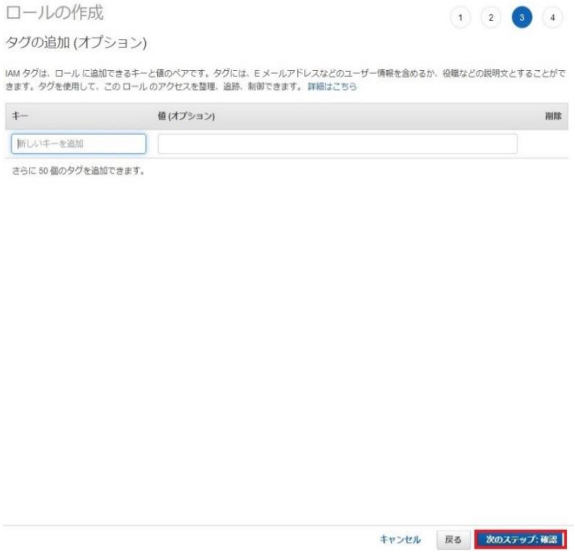
9.「ロールの作成」をクリックする

10.「AWSサービス」の「EC2」を選択し「次のステップ：アクセス権限」をクリックする



11.先ほど作成したポリシーにチェックをし「次のステップ：タグ」をクリックする

12.「次のステップ：確認」をクリックする



13 ロール名を入力し、先ほど選択したポリシーになっていることを確認し「ロールの作成」をクリックする

入力例）

ロール名	exastro-role
ポリシー	exastro-policy

14.ロールができていることを確認する

15.左側のメニューの「ユーザー」をクリックする

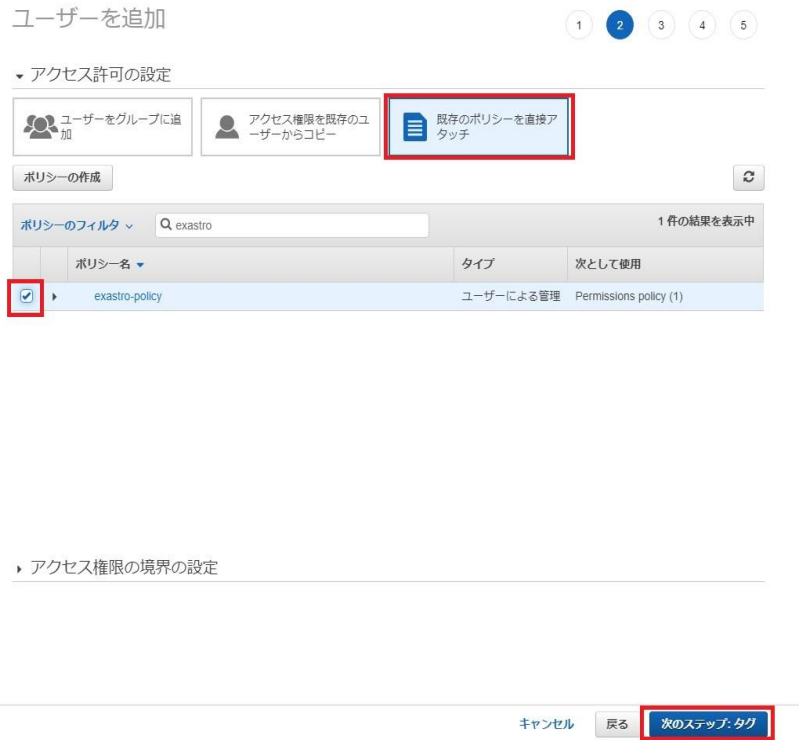
16.「ユーザーの追加」をクリックする

17.ユーザー名とプログラムによるアクセスにチェックを入れ、「次のステップ：アクセス権限」をクリックする

入力例）

ユーザー名	exastro-user
アクセスの種類	プログラムによるアクセス

18.既存のポリシーを直接アタッチを選択し、先ほど作成したポリシーをチェックして「次のステップ：タグ」をクリックする



19.「次のステップ：確認」をクリックする

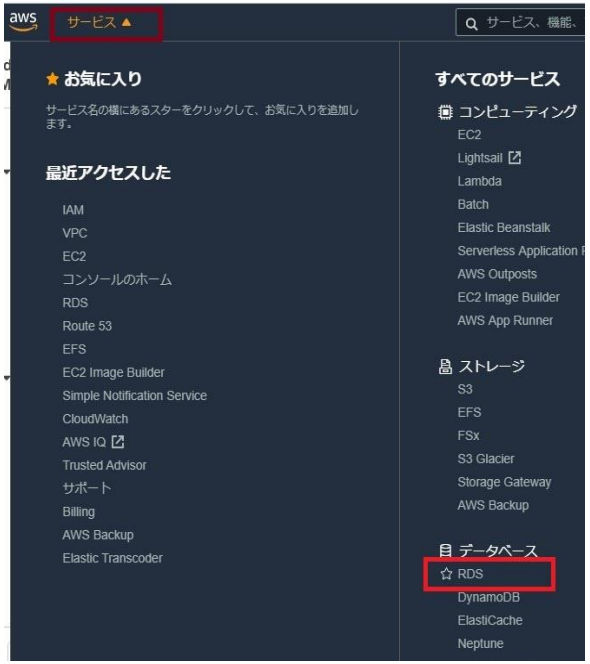
20.内容を確認し「ユーザーの作成」をクリックする

21.csvのダウンロードを実施する ※サーバ設定時に使用する

AWS設定手順書

7.RDS

1.左上のサービスからRDSをクリックする



2.左側のメニューから「サブネットグループ」をクリックする

3.「DBサブネットグループ作成」をクリックする

4.DBサブネットグループの情報を入力し、「作成」をクリックする

入力例）

名前	exatrodb-subnet
説明	rds-subnet
VPC	exastro-VPC
アベイラビリティゾーン	ap-northeast-1a
	ap-northeast-1c
サブネット	subnet-097fba3cefdf69f30（10.0.192.0/21）
	subnet-061dbdda51760b21d（10.0.200.0/21）

5.左側のパラメータグループをクリックし、「パラメータグループの作成」をクリックする

6.パラメーターグループファミリーを選択し、グループ名を記載して「作成」をクリックする ※説明は任意です

入力例）

パラメーターグループファミリー	mariadb10.5
グループ名	exastro-param

7.作成されたことを確認する

8.作成したパラメーターグループをクリックして、パラメーターの設定を実施する ※一つ設定するにつき「変更の保存」を必ずクリックすること


RDS > パラメータグループ > exastro-param
exastro-param
パラメータ
innodb_buffer_pool_size
編集のキャンセル
変更のプレビュー
元に戻す
変更の保存
名前
値
許可された値
変更可能
適用タイプ
データ型
説明
innodb_buffer_pool_size
536870912
5242880-18446744073709551615
true
system
dynamic
integer
The size in bytes of the memory buffer innodb uses to cache data and indexes of its tables


RDS > パラメータグループ > exastro-param
exastro-param
パラメータ
min_examined_row_limit
編集のキャンセル
変更のプレビュー
元に戻す
変更の保存
名前
値
許可された値
変更可能
送信元
適用タイプ
データ型
説明
min_examined_row_limit
100
0-18446744073709547520
true
engine-default
dynamic
integer
Can be used to cause queries which examine fewer than the stated number of rows not to be logged.


RDS > パラメータグループ > exastro-param
exastro-param
パラメータ
innodb_log_buffer_size
編集のキャンセル
変更のプレビュー
元に戻す
変更の保存
名前
値
許可された値
変更可能
送信元
適用タイプ
データ型
説明
innodb_log_buffer_size
67108864
262144-4294967295
true
system
static
integer
The size in bytes of the buffer that innodb uses to write to the log files on disk


パラメータ
query_cache_size
編集のキャンセル
変更のプレビュー
元に戻す
変更の保存
名前
値
許可された値
変更可能
送信元
適用タイプ
データ型
説明
query_cache_size
536870912
0-9223372036854774784
true
engine-default
dynamic
integer
The amount of memory allocated for caching query results.


exastro-param
パラメータ
join_buffer_size
編集のキャンセル
変更のプレビュー
元に戻す
変更の保存
名前
値
許可された値
変更可能
送信元
適用タイプ
データ型
説明
join_buffer_size
134217728
128-18446744073709547520
true
engine-default
dynamic
integer
Increase the value of join_buffer_size to get a faster full join when adding indexes is not possible.


パラメータ
query_cache_type
編集のキャンセル
変更のプレビュー
元に戻す
変更の保存
名前
値
許可された値
変更可能
送信元
適用タイプ
データ型
説明
query_cache_type
1
0-2
true
engine-default
static
integer
For query results either don't cache (=OFF), cache except for NO_CACHE (=ON), or only CACHE (=DEMAND)


パラメータ
tmp_table_size
編集のキャンセル
変更のプレビュー
元に戻す
変更の保存
名前
値
許可された値
変更可能
送信元
適用タイプ
データ型
説明
tmp_table_size
67108864
0-4294967295
true
engine-default
dynamic
integer
If an in-memory temporary table exceeds the limit, MySQL automatically converts it to an on-disk MyISAM table. Increased value can improve perf for many advanced GROUP BY queries.

パラメーター	値
explicit_defaults_for_timestamp	1
tx_isolation	READ-COMMITTED
innodb_buffer_pool_size	536870912
innodb_log_buffer_size	67108864
min_examined_row_limit	100
join_buffer_size	134217728
query_cache_size	536870912
query_cache_type	1
max_heap_table_size	67108864
tmp_table_size	67108864
mrr_buffer_size	67108864
max_connections	256
character_set_server	utf8

9.保存後、左側メニューのダッシュボードから「データベースの作成」をクリックする

10.RDSを作成する

※入力例の記載が無いものはデフォルト値です(パスワードは任意です)

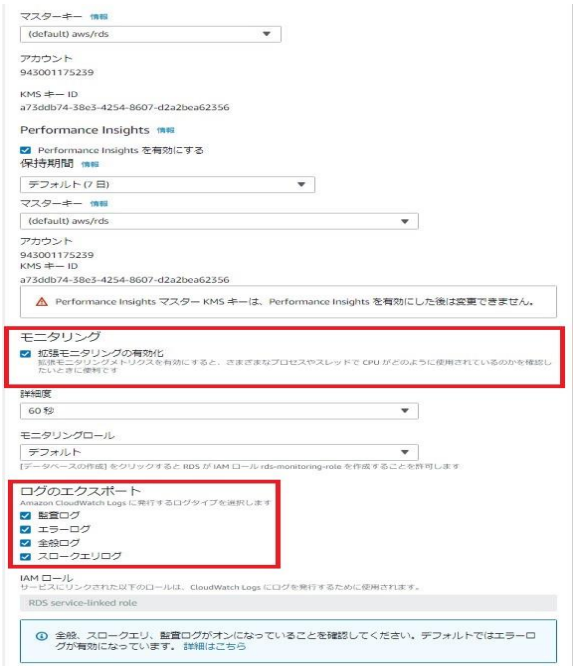
入力例)

作成方法	標準作成
エンジンのオプション	MariaDB
バージョン	MariaDB 10.5.8
テンプレート	本番稼働用
DBインスタンス識別子	exastro-rds
マスターユーザー名	admin
可用性と耐久性	スタンバイインスタンスを作成する
VPC	exastro-VPC
サブネットグループ	exastrodb-subnet
パブリックアクセス可能	なし
VPCセキュリティグループ	既存の選択
	exastro-rds
データベースポート	3306
DBパラメータグループ	exastro-param
オプショングループ	defaultmariadb-10.5
バックアップ	自動バックアップの有効化
モニタリング	拡張モニタリングの有効化
ログのエクスポート	監視ログ
	エラーログ
	全般ログ
	スロークエリログ
選択ウィンドウ	土曜日 03:00 0.5時間


設定
DB インスタンス識別子 情報
DB インスタンスの名前を入力します。この名前は、AWS アカウントが現在の AWS リージョンで所有しているすべての DB インスタンスにおいて一意である必要があります。
exastro-rds
DB インスタンス識別子は大文字と小文字を区別しませんが、すべて小文字で保存されます (例: "mydbinstance")。制約として、使用できるのは 1 ～ 60 文字以内で英数字またはハイフンのみです。1 字目は英文字でなければなりません。また、ハイフンを連続で 2 つ使ったり、最後の文字をハイフンにしたりすることはできません。
認証情報の設定
マスターユーザー名 情報
DB インスタンスのマスターユーザーのログイン ID を入力します。
admin
1～16 文字の英数字。1 字目は文字である必要があります
パスワードの自動生成
Amazon RDS がパスワードを生成するか、お客様がご自身でパスワードを指定することができます。
マスターパスワード 情報
制約事項: 表示可能な ASCII 文字で 8 文字以上で入力してください次の文字を含めることはできません: / (スラッシュ)、'(単一引用符)、" (二重引用符)、および @ (アットマーク)。
パスワードを確認 情報
DB インスタンスサイズ
DB インスタンスクラス 情報
処理能力とメモリの要件に合った DB インスタンスクラスを選択します。以下の DB インスタンスクラスオプションは、上記で選択したエンジンでサポートされているものに制限されます。
標準クラス (m クラスを含む)
メモリ最適化クラス (r クラスと x クラスを含む)
バースト可能クラス (t クラスを含む)
db.m6g.large
2 vCPUs 8 GiB RAM Network: 4,750 Mbps
以前の世代のクラスを含める
ストレージ
ストレージタイプ 情報
プロビジョンド IOPS (SSD)
ストレージ割り当て
100
GiB
最小: 100 GiB、最大: 65,536 GiB
プロビジョンド IOPS 情報
3000
IOPS
Minimum: 1,000 IOPS, Maximum: 80,000 IOPS
ストレージの自動スケーリング 情報
アプリケーションのニーズに基づいて、データベースのストレージに対する動的なスケーリングのサポートを提供します。
ストレージの自動スケーリングを有効にする
この機能を有効にすると、指定したしきい値を超えた場合にストレージを増やすことができます。
最大ストレージしきい値 情報
データベースが指定されたしきい値に自動スケールされると、料金が適用されます。
1000
GiB
最小: 101 GiB、最大: 65,536 GiB
可用性と耐久性
マルチ AZ 配置 情報
スタンバイインスタンスを作成する (本稼働環境向けに推奨)
データの冗長性を提供し、I/O のフリーズを防ぎ、システムバックアップの間のレイテンシーの急上昇を最小限に抑えるために、別のアベイラビリティーゾーン (AZ) にスタンバイを作成します。
スタンバイインスタンスを作成しないでください


接続
Virtual Private Cloud (VPC) 情報
exastro-VPC (vpc-0e755a2dfc566d299)
データベースの作成後に、VPC の選択を変更することはできません。
サブネットグループ 情報
exastrodb-subnet
パブリックアクセス可能 情報
あり
なし
VPC セキュリティグループ
既存の選択
新規作成
既存の VPC セキュリティグループ
VPC セキュリティグループを選択します
exastro-rds
追加の接続設定
データベースポート 情報
3306


追加設定
データベースの選択肢
最初のデータベース名 情報
DB パラメータグループ 情報
exastro-param
オプショングループ 情報
default:mariadb-10-5
バックアップ
自動バックアップの有効化
バックアップ保持期間 情報
7 日間
バックアップウィンドウ 情報
選択ウィンドウ
設定なし
開始時間
05
00
UTC
期間
0.5
時間
スナップショットにタグをコピー
暗号化
暗号を有効化

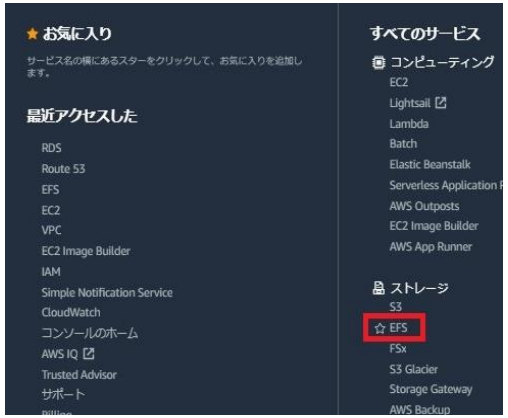


11.RDSが作成され、利用可能になることを確認する　※少し時間がかかります

AWS設定手順書

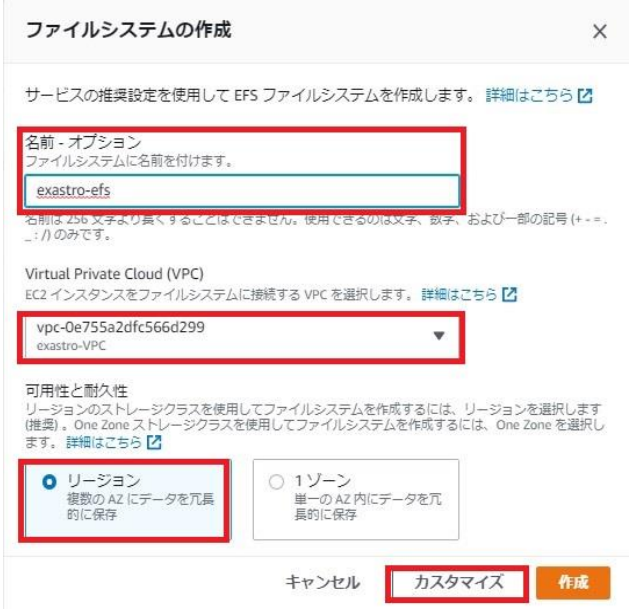
8.EFS

1.左上のサービスからEFSをクリックする



2.「ファイルシステムの作成」をクリックする

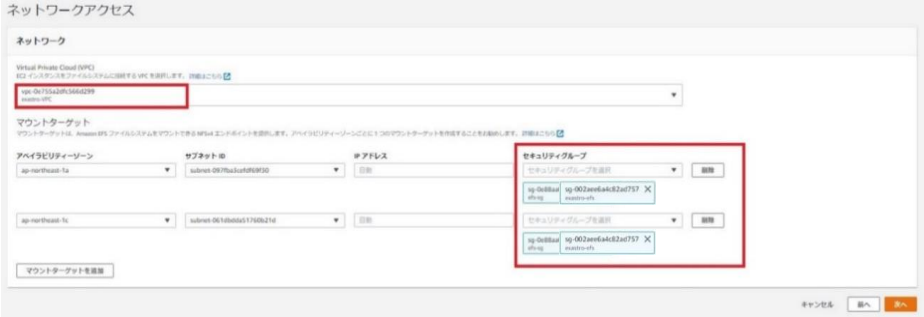
3.名前を入力し、作成したVPCを選択、リージョンを選択し「カスタマイズ」をクリックする



入力例）

名前-オプション	exastro-efs
VPC	exastro-VPC
可用性と耐久性	リージョン

4.自動バックアップを選択し、ネットワークは先ほどのVPCから選択し、EFS用に作成したセキュリティグループを追加し「次へ」をクリックする

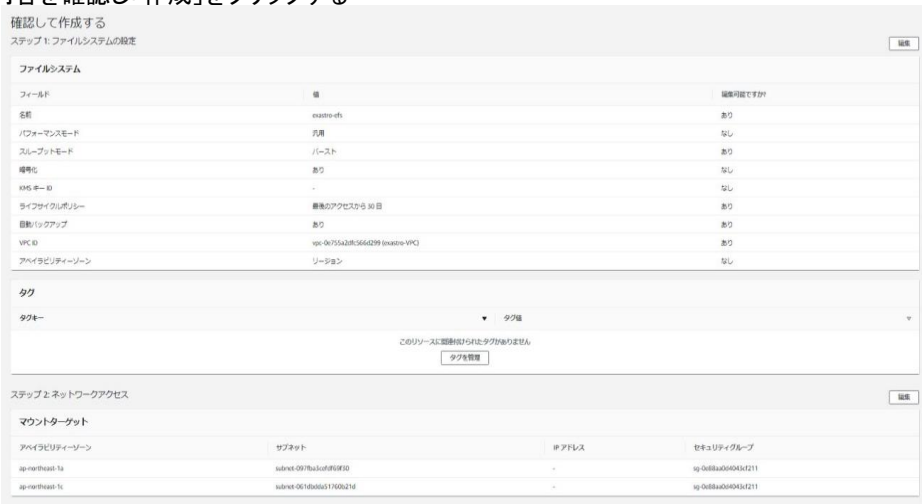


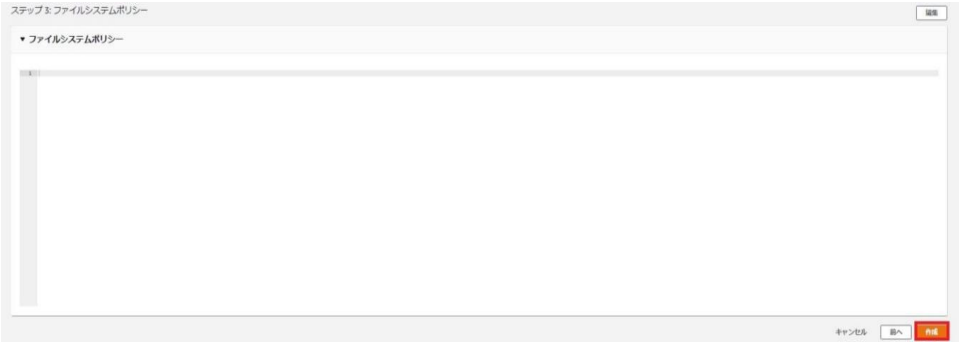
入力例）

ネットワーク	exastro-VPC
セキュリティグループ	exastro-efs

5.「次へ」をクリックする

6.内容を確認し「作成」をクリックする

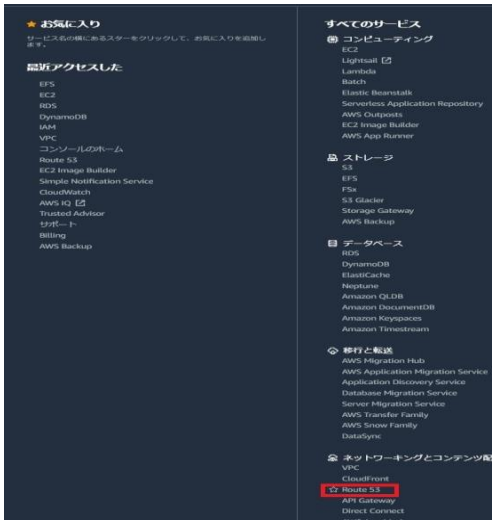




AWS設定手順書

9.Route53

1.左上のサービスからRoute53をクリックする



2.「ホストゾーンの作成」をクリックする

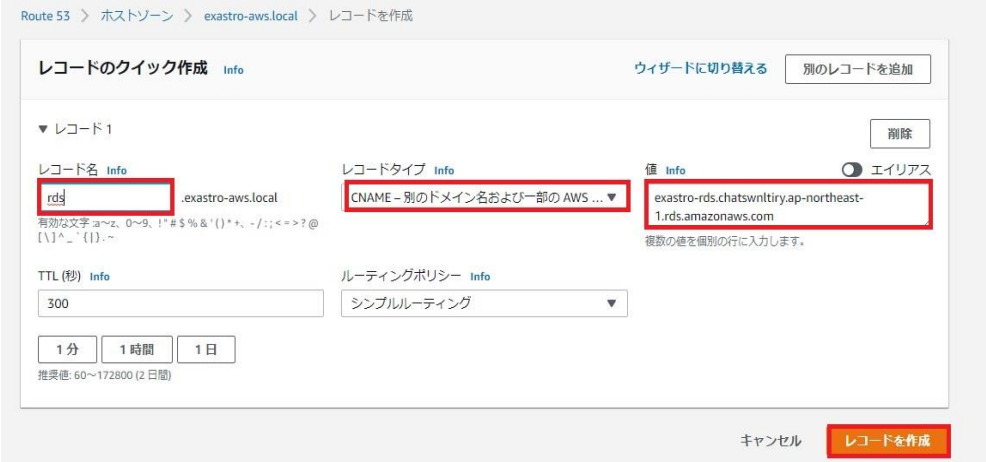
3.ドメイン名を入力し、プライベートホストゾーンにチェックし、東京リージョンとVPCを選択し「ホストゾーンの作成」をクリックする

入力例）

ドメイン名	exastro-aws.local
タイプ	プライベートホストゾーン
リージョン	アジアパシフィック(東京)[ap-northeast-1]
VPC ID	vpc-0e755a2dfc566d299

4.「レコードの作成」をクリックする

5.レコード名、CNAMEを選択しRDSのエンドポイントを値に入力し、「レコードを作成」をクリックする



入力例）

レコード名	rds
レコードタイプ	CNAME
値	RDSのエンドポイント

6.レコード名、CNAMEを選択しEFSのエンドポイントを値に入力し、「レコードを作成」をクリックする

入力例）

レコード名	efs
レコードタイプ	CNAME
値	EFSのエンドポイント

AWS設定手順書

10.EC2

1.左上のサービスからEC2をクリックする

2.「インスタンスを起動」をクリックする

3.Red Had with Linux 8 High Availabilityと検索ボックスに入力し、選択する　※Pacemaker、Corosync、pcsが既存でインストールされているイメージです

4.サイズを選択する　※Exastroに必要な最低限のスペックにて実施

5.作成したVPC、publicなサブネット、自動割り当てパブリックIP、作成したIAMロールを選択し「次のステップ：ストレージの追加」をクリックする

入力例）

ita-ha-sv01	
ネットワーク	exastro-VPC
サブネット	public-a
自動割り当てパブリックIP	有効
IAMロール	exastro-role
セキュリティグループ	exastro-sg

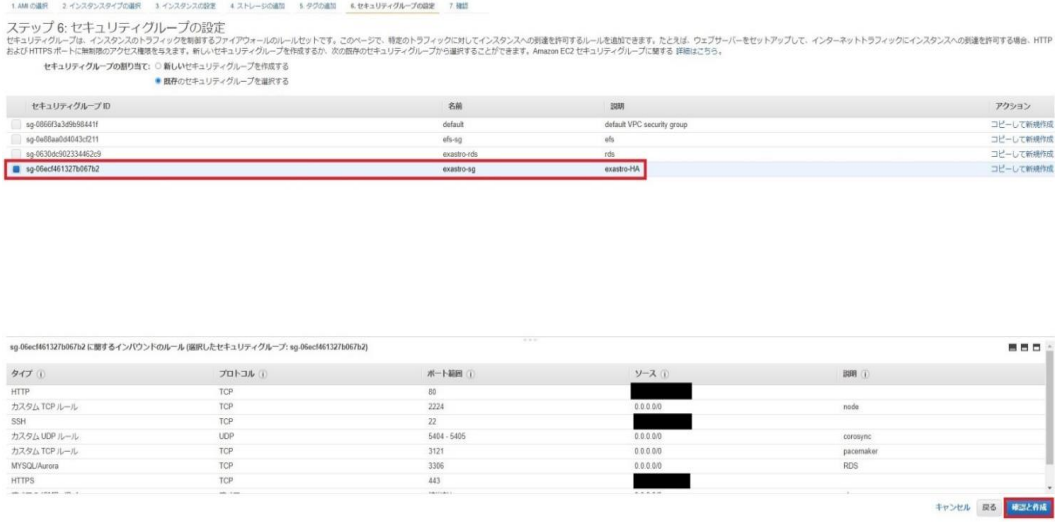
入力例）

ita-ha-sv02	
ネットワーク	exastro-VPC
サブネット	public-c
自動割り当てパブリックIP	有効
IAMロール	exastro-role
セキュリティグループ	exastro-sg

6.「次のステップ：タグの追加」をクリックする

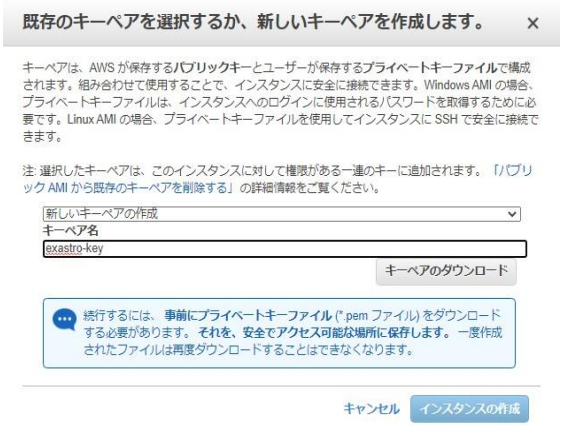
7.「次のステップ：セキュリティグループ」をクリックする

8.作成したセキュリティグループを選択し、「確認と作成」をクリックする



9.確認し、問題なければ「起動」をクリックする　※public-a、public-cに一台ずつ作成する

10.SSHに必要な鍵を作成する場合はこちらで作成する



11.作成後アクション→ネットワーキング→ソース宛先/変更をクリックする

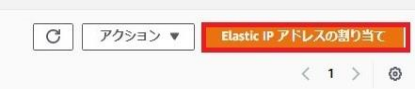
12.停止にチェックを入れ、「保存」をクリックする　※2台分実施すること

AWS設定手順書

11.Elastic IP

1.左側のメニューのElastic IPをクリックする

2.「IPアドレスの割り当て」をクリックする



3.ネットワークボーダーグループが東京リージョンになっていることを確認し、「割り当て」をクリックする

4.IPが生成されたらアクション→Elastic IPアドレスの関連付けをクリックする

5.インスタンスを選択し（public-aで動作しているインスタンス）、プライベートIPアドレスを入力し（候補に出るのでそれを選択）「関連付ける」をクリックする

これで全てのAWS上の設定は完了しました。
次は構築手順になります。

構築手順

実行ユーザ／接続条件
rootユーザで実行すること rootユーザがsshログイン可能であること

No.	作業項目	サーバ		設定内容 （※master、slaveの両方が対象の場合は、並行して設定してください。）	備考
		master	slave		
1.ITAのインストール					
1-1	ITAをインストールする	●	●	以下URLのオールインワン構成のインストールマニュアル参照 https://exastro-suite.github.io/it-automation-docs/learn_ja.html#deploy	
1-2	ITAインストールディレクトリの設定をする	●	●	find /(インストール資材展開先)/it-automation-1.8.0 -type f | xargs -I{} sed -i -e "s:%%%%%ITA_DIRECTORY%%%%%:/(ITAインストール先ディレクトリ):g" {}	赤字は、環境に合わせて読み替えること
2.Apacheリソース設定					
2-1	apacheのserver statusを作成する	●	●	cat > /etc/httpd/conf.d/server_status.conf << STAT ExtendedStatus On <Location /server-status> SetHandler server-status Order deny,allow Deny from all Allow from localhost </Location> STAT	コマンドをまとめて実施する
2-2	Apacheサービス停止・無効化する	●	●	systemctl disable httpd systemctl stop httpd	
2-3	httpsサーバ証明書／秘密鍵をmasterからslaveへコピーする	●	●	scp -r -i [鍵].pem /etc/pki/tls/certs/(httpsサーバ証明書名).crt <slaveサーバのローカルIPアドレス>:/etc/pki/tls/certs/ scp -r -i [鍵].pem /etc/pki/tls/certs/(httpsサーバ秘密鍵名).key <slaveサーバのローカルIPアドレス>:/etc/pki/tls/certs/	赤字は、環境に合わせて読み替えること (httpsサーバ証明書／秘密鍵は、ITAサーバインストール時に作成した、ファイル名に読み替えること)

構築手順

実行ユーザ／接続条件
rootユーザで実行すること rootユーザがsshログイン可能であること

No.	作業項目	サーバ		設定内容 （※master、slaveの両方が対象の場合は、並行して設定してください。）	備考
		master	slave		
3.ITAサービス停止設定					
3-1	ky-serviceを停止する	●	●	systemctl stop ky_activedirectory_roleuser_replication-workflow.service systemctl stop ky_ansible_execute-workflow.service systemctl stop ky_ansible_towermasterSync-workflow.service systemctl stop ky_bulk_excel-workflow.service systemctl stop ky_change_col_to_row.service systemctl stop ky_cmdbmenuanalysis-workflow.service systemctl stop ky_create_er-workflow.service systemctl stop ky_create_param_menu_execute.service systemctl stop ky_data_portability_execute-workflow.service systemctl stop ky_hostgroup_check_loop.service systemctl stop ky_hostgroup_split.service systemctl stop ky_legacy_role_valautostup-workflow.service systemctl stop ky_legacy_role_varsautolistup-workflow.service systemctl stop ky_legacy_valautostup-workflow.service systemctl stop ky_legacy_varsautolistup-workflow.service systemctl stop ky_mail.service systemctl stop ky_pioneer_valautostup-workflow.service systemctl stop ky_pioneer_varsautolistup-workflow.service systemctl stop ky_std_checkcondition-linklist.service systemctl stop ky_std_synchronize-Collector.service systemctl stop ky_std_synchronize-Conductor.service systemctl stop ky_std_synchronize-regularly2.service systemctl stop ky_std_synchronize-regularly.service systemctl stop ky_std_synchronize-symphony.service systemctl stop ky_terraform_execute-workflow.service systemctl stop ky_terraform_checkcondition-workflow.service systemctl stop ky_terraform_varsautolistup-workflow.service systemctl stop ky_terraform_valautosetup-workflow.service	ITAのバージョンによって、存在しないサービスがあります。 その場合は、サービス停止不要です。
3-2	ky-service 自動起動設定を無効にする	●	●	systemctl disable ky_activedirectory_roleuser_replication-workflow.service systemctl disable ky_ansible_execute-workflow.service systemctl disable ky_ansible_towermasterSync-workflow.service systemctl disable ky_bulk_excel-workflow.service systemctl disable ky_change_col_to_row.service systemctl disable ky_cmdbmenuanalysis-workflow.service systemctl disable ky_create_er-workflow.service systemctl disable ky_create_param_menu_execute.service systemctl disable ky_data_portability_execute-workflow.service systemctl disable ky_hostgroup_check_loop.service systemctl disable ky_hostgroup_split.service systemctl disable ky_legacy_role_valautostup-workflow.service systemctl disable ky_legacy_role_varsautolistup-workflow.service systemctl disable ky_legacy_valautostup-workflow.service systemctl disable ky_legacy_varsautolistup-workflow.service systemctl disable ky_mail.service systemctl disable ky_pioneer_valautostup-workflow.service systemctl disable ky_pioneer_varsautolistup-workflow.service systemctl disable ky_std_checkcondition-linklist.service systemctl disable ky_std_synchronize-Collector.service systemctl disable ky_std_synchronize-Conductor.service systemctl disable ky_std_synchronize-regularly2.service systemctl disable ky_std_synchronize-regularly.service systemctl disable ky_std_synchronize-symphony.service systemctl disable ky_terraform_execute-workflow.service systemctl disable ky_terraform_checkcondition-workflow.service systemctl disable ky_terraform_varsautolistup-workflow.service systemctl disable ky_terraform_valautosetup-workflow.service	ITAのバージョンによって、存在しないサービスがあります。 その場合は、設定の無効は不要です。

構築手順

実行ユーザ／接続条件
rootユーザで実行すること rootユーザがsshログイン可能であること

No.	作業項目	サーバ		設定内容 （※master、slaveの両方が対象の場合は、並行して設定してください。）	備考
		master	slave		
4.EFS設定					
前提	・EFSが作成されていること ・ITAがインストール済みであること				
4-1	東京に時刻設定をする	●	●	timedatectl set-timezone Asia/Tokyo	
4-2	EFSの使用準備をする	●	●	cd /tmp yum -y install git git clone https://github.com/aws/efs-utils yum -y install make rpm-build cd efs-utils make rpm yum -y install ./build/amazon-efs-utils*rpm	
4-3	EFSファイル共有設定をする	●		mount -t efs -o tls [Route53に登録したEFSの名前解決設定の情報]:/ /mnt cd /mnt mkdir -p ./(ITAのインストールパス)/data_relay_storage/symphony mkdir -p ./(ITAのインストールパス)/data_relay_storage/conductor mkdir -p ./(ITAのインストールパス)/data_relay_storage/ansible_driver mkdir -p ./(ITAのインストールパス)/ita_sessions mkdir -p ./(ITAのインストールパス)/ita-root/webconfs/sheets mkdir -p ./(ITAのインストールパス)/ita-root/webconfs/users mkdir -p ./(ITAのインストールパス)/ita-root/temp mkdir -p ./(ITAのインストールパス)/ita-root/uploadfiles mkdir -p ./(ITAのインストールパス)/ita-root/webroot/uploadfiles mkdir -p ./(ITAのインストールパス)/ita-root/webroot/menus/sheets mkdir -p ./(ITAのインストールパス)/ita-root/webroot/menus/users chmod 777 ./(ITAのインストールパス)/data_relay_storage/symphony chmod 777 ./(ITAのインストールパス)/data_relay_storage/conductor chmod 777 ./(ITAのインストールパス)/data_relay_storage/ansible_driver chmod 777 ./(ITAのインストールパス)/ita_sessions chmod 755 ./(ITAのインストールパス)/ita-root/webconfs/sheets chmod 755 ./(ITAのインストールパス)/ita-root/webconfs/users chmod 777 ./(ITAのインストールパス)/ita-root/temp chmod 777 ./(ITAのインストールパス)/ita-root/uploadfiles chmod 777 ./(ITAのインストールパス)/ita-root/webroot/uploadfiles chmod 755 ./(ITAのインストールパス)/ita-root/webroot/menus/sheets chmod 755 ./(ITAのインストールパス)/ita-root/webroot/menus/users	赤字は、環境に合わせて読み替えること AWSコンソールからEFSを作成してから本手順を実施する

構築手順

実行ユーザ／接続条件
rootユーザで実行すること rootユーザがsshログイン可能であること

No.	作業項目	サーバ		設定内容 （※master、slaveの両方が対象の場合は、並行して設定してください。）	備考
		master	slave		
4-4	共有ファイルのコピーを実施する	●		cp -rp /(ITAインストールパス)/data_relay_storage/symphony /mnt/(ITAインストールパス)/data_relay_storage/ cp -rp /(ITAインストールパス)/data_relay_storage/conductor /mnt/(ITAインストールパス)/data_relay_storage/ cp -rp /(ITAインストールパス)/data_relay_storage/ansible_driver /mnt/(ITAインストールパス)/data_relay_storage/ cp -rp /(ITAインストールパス)/ita_sessions /mnt/(ITAインストールパス)/ cp -rp /(ITAインストールパス)/ita-root/webconfs/sheets /mnt/(ITAインストールパス)/ita-root/webconfs/ cp -rp /(ITAインストールパス)/ita-root/webconfs/users /mnt/(ITAインストールパス)/ita-root/webconfs/ cp -rp /(ITAインストールパス)/ita-root/temp /mnt/(ITAインストールパス)/ita-root/ cp -rp /(ITAインストールパス)/ita-root/uploadfiles /mnt/(ITAインストールパス)/ita-root/ cp -rp /(ITAインストールパス)/ita-root/webroot/uploadfiles /mnt/(ITAインストールパス)/ita-root/webroot/ cp -rp /(ITAインストールパス)/ita-root/webroot/menus/sheets /mnt/(ITAインストールパス)/ita-root/webroot/menus/ cp -rp /(ITAインストールパス)/ita-root/webroot/menus/users /mnt/(ITAインストールパス)/ita-root/webroot/menus/ cd /tmp umount /mnt	
4-5	複数ディレクトリをマウントする設定をする				
	設定ファイルを作成する	●	●	vi /etc/systemd/system/mount-nfs-sequentially.service [Unit] Description=Workaround for mounting NFS file systems sequentially at boot time After=remote-fs.target [Service] Type=oneshot ExecStart=/bin/mount -avt nfs4 RemainAfterExit=yes [Install] WantedBy=multi-user.target	
	上記設定を反映させる	●	●	systemctl daemon-reload systemctl enable mount-nfs-sequentially.service	

構築手順

実行ユーザ／接続条件
rootユーザで実行すること rootユーザがsshログイン可能であること

No.	作業項目	サーバ		設定内容 （※master、slaveの両方が対象の場合は、並行して設定してください。）	備考
		master	slave		
5.RDS設定					
前提	・RDSが作成されていること ・ITAがインストール済みであること				
5-1	RDSに接続する	●		mysql -h [DBのホスト名（Route53にRDSの名前解決設定の情報）] -P 3306 -u admin -p<adminパスワード> 例）mysql -h rds.exastro-aws.local -P 3306 -u admin -p********	赤字は、環境に合わせて読み替えること
5-2	ユーザとDBを作成する	●		CREATE USER '<ユーザ名>' IDENTIFIED BY '<パスワード>'; CREATE USER '<ユーザ名>'@'localhost' IDENTIFIED BY '<パスワード>'; CREATE DATABASE [DB名] CHARACTER SET utf8; GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, REFERENCES, INDEX, ALTER, CREATE TEMPORARY TABLES, LOCK TABLES, EXECUTE, CREATE VIEW, SHOW VIEW, CREATE ROUTINE, ALTER ROUTINE, EVENT, TRIGGER ON [DB名].* TO '<ユーザ名>'@'%' WITH GRANT OPTION; GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, REFERENCES, INDEX, ALTER, CREATE TEMPORARY TABLES, LOCK TABLES, EXECUTE, CREATE VIEW, SHOW VIEW, CREATE ROUTINE, ALTER ROUTINE, EVENT, TRIGGER ON [DB名].* TO '<ユーザ名>'@'localhost' WITH GRANT OPTION; exit	赤字は、環境に合わせて読み替えること
5-3	DBへのデータ流し込みを実施する	●		mysqldump --default-character-set=utf8 -u <ita_answer.txtで設定したDBユーザー名> -p<ita_answer.txtで設定したDBパスワード> [ita_answer.txtで設定したDB名] > ITA_DB.sql sed -i 's/DEFINER=[^*]*¥*/¥*/g' ITA_DB.sql mysql --default-character-set=utf8 -h [DBのホスト名（Route53にEFSの名前解決設定の情報）] -P 3306-u <RDSのマスターユーザ名> -p<RDSのマスターパスワード> [DB名] < ITA_DB.sql 例） mysql --default-character-set=utf8 -h rds.exastro-aws.local -P 3306 -u admin -p******* ITA_DB < ITA_DB.sql	赤字は、環境に合わせて読み替えること

構築手順

実行ユーザ／接続条件
rootユーザで実行すること rootユーザがsshログイン可能であること

No.	作業項目	サーバ		設定内容 （※master、slaveの両方が対象の場合は、並行して設定してください。）	備考
		master	slave		
5-4	DBの接続設定を実施する				
	MariaDB接続情報設定	●	●	①以下の文字列をbase64エンコード、rot13暗号を行った文字列を作成してください。 echo -ne "mysql:dbname=[DB名];host=[DBのホスト名（Route53にRDSの名前解決設定の情報）]" | base64 | tr '[A-Za-z]' '[N-ZA-Mn-za-m]' ②作成した文字列を以下のファイルに書き込んでください。※記載済みの値は削除してください vi /(ITAインストール先ディレクトリ)/ita-root/confs/commonconfs/db_connection_string.txt	例： ITAのDB名が「ITA_DB」、Route53に登録しているDNS名がrds.exastro-aws.localの場合、以下のコマンドで base64エンコード、rot13暗号を行った文字列を取得できます。 echo -ne "mysql:dbname=ITA_DB;host=rds.exastro-aws.local" | base64 | tr '[A-Za-z]' '[N-ZA-Mn-za-m]' 赤字は、環境に合わせて読み替えること
	MariaDBのユーザ情報設定	●	●	①MariaDBのユーザ名をbase64エンコード、rot13暗号を行った文字列を作成してください。 echo -ne "<ユーザ名>" | base64 | tr '[A-Za-z]' '[N-ZA-Mn-za-m]' ②作成した文字列を以下のファイルに書き込んでください。※記載済みの値は削除してください vi /(ITAインストール先ディレクトリ)/ita-root/confs/commonconfs/db_username.txt	例： MariaDBのユーザ名が「ITA_USER」の場合、以下のコマンドで base64エンコード、rot13暗号を行った文字列を取得できます。 echo -ne "ITA_USER" | base64 | tr '[A-Za-z]' '[N-ZA-Mn-za-m]' 赤字は、環境に合わせて読み替えること
	MariaDBのパスワード情報設定	●	●	①MariaDBのパスワードをbase64エンコード、rot13暗号を行った文字列を作成してください。 echo -ne "<パスワード>" | base64 | tr '[A-Za-z]' '[N-ZA-Mn-za-m]' ②作成した文字列を以下のファイルに書き込んでください。※記載済みの値は削除してください vi /(ITAインストール先ディレクトリ)/ita-root/confs/commonconfs/db_password.txt	例： MariaDBのパスワードが「ITA_PASSWD」の場合、 以下のコマンドでbase64エンコード、rot13暗号を行った 文字列を取得できます。 echo -ne "ITA_PASSWD" | base64 | tr '[A-Za-z]' '[N-ZA-Mn-za-m]' 赤字は、環境に合わせて読み替えること
5-5	MariaDBを停止する	●	●	systemctl disable mariadb systemctl stop mariadb	
6.Pacemaker使用準備					
6-1	awscliをインストールする	●	●	yum -y install awscli	
6-2	ユーザー作成で作成したcredentials.csvを確認する	●	●	aws configure AWS Access Key ID [None]: ********** AWS Secret Access Key [None]: ********** Default region name [None]: ap-northeast-1 Default output format [None]: json	赤字は、環境に合わせて読み替えること IAM作成時にダウンロードしたcsvを参照すること
6-3	ホスト名とIPをhostsに登録する	●	●	uname -n vi /etc/hosts ローカルIP ホスト名 ローカルIP ホスト名 例） 10.0.0.204 <ita-ha-sv01のホスト名> 10.0.1.61 <ita-ha-sv02のホスト名>	赤字は、環境に合わせて読み替えること

構築手順

実行ユーザ／接続条件
rootユーザで実行すること rootユーザがsshログイン可能であること

No.	作業項目	サーバ		設定内容 （※master、slaveの両方が対象の場合は、並行して設定してください。）	備考
		master	slave		
7.Pacemaker設定					
7-1	pcs の管理アカウントとなるユーザーID「hacluster」 のパスワードを設定する	●	●	passwd hacluster 新しいパスワード:xxxxxx 新しいパスワードを再入力してください:xxxxxx passwd: すべての認証トークンが正しく更新できました。	赤字は、環境に合わせて読み替えること
7-2	pcsdサービスを起動し、システムの起動時に pcsd が有効になるよう設定する	●	●	systemctl enable pcsd systemctl start pcsd systemctl status pcsd	
7-3	クラスターを認証する	●		pcs host auth <ita-ha-sv01のホスト名> <ita-ha-sv02のホスト名> Username: hacluster Password:passwd haclusterで設定したパスワード	赤字は、環境に合わせて読み替えること
7-4	クラスターを指定してスタートする	●		pcs cluster setup aws-cluster --start <ita-ha-sv01のホスト名> <ita-ha-sv02のホスト名> --force	
7-5	クラスターを自動起動するように設定をする	●		pcs cluster enable --all	
7-6	STONITH (Shoot The Other Node In The Head) オプションを無効にする	●		pcs property set stonith-enabled=false	
7-7	クラスターが必要最低限の数に達していない場合でも特別な動作は行わず全ノードの管理を続	●		pcs property set no-quorum-policy=ignore	
7-8	属性待ち時間を設定する	●		pcs property set transition-delay="0s"	
7-9	自動フェイルバックを無効にする	●		pcs resource defaults resource-stickiness="INFINITY" migration-threshold="1"	
7-10	corosync設定をする	●	●	cp -p /usr/lib/systemd/system/corosync.service /etc/systemd/system/ sed -i 's/^#Restart=on.*/Restart=on-failure/' /etc/systemd/system/corosync.service sed -i 's/^#RestartSec=.*/RestartSec=70/' /etc/systemd/system/corosync.service	
7-11	Pacemakerの内部プロセスが異常になった場合もノード故障として取り扱う	●	●	sed -i 's/^#¥ PCMK¥_fail¥_fast.*/PCMK_fail_fast=yes/' /etc/sysconfig/pacemaker	
7-12	corosync設定をする	●	●	cp -p /usr/lib/systemd/system/pacemaker.service /etc/systemd/system sed -i "s/^#¥ ExecStopPost=¥/bin¥/sh.*/ExecStopPost=¥/bin¥/sh -c 'pidof crmd ¥|¥| killall -TERM corosync'/" /etc/systemd/system/pacemaker.service	

7-13	IP付け替えの設定をする	●	●	vi /etc/sysconfig/pacemaker
				
				/etc/sysconfig/pacemakerの中身
				# Set the options to pass to valgrind, when valgrind is enabled. See
				# valgrind(1) man page for details. "--vgdb=no" is specified because
				# pacemaker-execd can lower privileges when executing commands, which would
				# otherwise leave a bunch of unremovable files in /tmp.
				VALGRIND_OPTS="--leak-check=full --trace-children=no --vgdb=no --num-callers=25 --log-file=/var/lib/pacemaker/valgrind-%p --suppressions=/usr/share/pacemaker/tests/valgrind-pcmk.suppressions --gen-suppressions=all"
				AWS_DEFAULT_REGION=ap-northeast-1　←記載部分
					赤字は、環境に合わせて読み替えること
					(リージョンが東京の場合は、末尾に
					AWS_DEFAULT_REGION=ap-northeast-1 と記載)

構築手順

実行ユーザ／接続条件
rootユーザで実行すること rootユーザがsshログイン可能であること

No.	作業項目	サーバ		設定内容 （※master、slaveの両方が対象の場合は、並行して設定してください。）	備考
		master	slave		
7-14	httpd設定をする	●		pcs resource create httpd systemd:httpd ¥ op monitor interval=60 timeout=100 ¥ op start interval=0s timeout=100 ¥ op stop interval=0s timeout=100	
7-15	eip設定をする	●		pcs resource create eip ocf:heartbeat:awseip ¥ elastic_ip="ElasticIPのIPアドレス" awscli="$(which aws)" ¥ allocation_id=ElasticIPの割り当てID ¥ op start timeout="120s" interval="0s" on-fail="restart" ¥ op monitor timeout="20s" interval="30s" on-fail="restart" ¥ op stop timeout="120s" interval="0s" on-fail="block"	赤字は、環境に合わせて読み替えること
7-16	httpd、eipリソースグループ設定をする	●		pcs resource group add exastro httpd eip	
7-17	EFSリソース設定をする	●		pcs resource create symphony Filesystem device='[Route53に登録したEFSの名前解決設定の情報]:/(ITAインストール先ディレクトリ)/data_relay_storage/symphony' directory='/(ITAインストール先ディレクトリ)/data_relay_storage/symphony' fstype='nfs' pcs resource create conductor Filesystem device='[Route53に登録したEFSの名前解決設定の情報]:/(ITAインストール先ディレクトリ)/data_relay_storage/conductor' directory='/(ITAインストール先ディレクトリ)/data_relay_storage/conductor' fstype='nfs' pcs resource create ansible_driver Filesystem device='[Route53に登録したEFSの名前解決設定の情報]:/(ITAインストール先ディレクトリ)/data_relay_storage/ansible_driver' directory='/(ITAインストール先ディレクトリ)リ)/data_relay_storage/ansible_driver' fstype='nfs' pcs resource create ita_sessions Filesystem device='[Route53に登録したEFSの名前解決設定の情報]:/(ITAインストール先ディレクトリ)/ita_sessions' directory='/(ITAインストール先ディレクトリ)/ita_sessions' fstype='nfs' pcs resource create webconfs_sheets Filesystem device='[Route53に登録したEFSの名前解決設定の情報]:/(ITAインストール先ディレクトリ)/ita-root/webconfs/sheets' directory='/(ITAインストール先ディレクトリ)/ita-root/webconfs/sheets' fstype='nfs' pcs resource create webconfs_users Filesystem device='[Route53に登録したEFSの名前解決設定の情報]:/(ITAインストール先ディレクトリ)/ita-root/webconfs/users' directory='/(ITAインストール先ディレクトリ)/ita-root/webconfs/users' fstype='nfs' pcs resource create temp Filesystem device='[Route53に登録したEFSの名前解決設定の情報]:/(ITAインストール先ディレクトリ)/ita-root/temp' directory='/(ITAインストール先ディレクトリ)/ita-root/temp' fstype='nfs' pcs resource create uploadfiles Filesystem device='[Route53に登録したEFSの名前解決設定の情報]:/(ITAインストール先ディレクトリ)/ita-root/uploadfiles' directory='/(ITAインストール先ディレクトリ)/ita-root/uploadfiles' fstype='nfs' pcs resource create webroot_uploadfiles Filesystem device='[Route53に登録したEFSの名前解決設定の情報]:/(ITAインストール先ディレクトリ)/ita-root/webroot/uploadfiles' directory='/(ITAインストール先ディレクトリ)/ita-root/webroot/uploadfiles' fstype='nfs' pcs resource create menus_sheets Filesystem device='[Route53に登録したEFSの名前解決設定の情報]:/(ITAインストール先ディレクトリ)/ita-root/webroot/menus/sheets' directory='/(ITAインストール先ディレクトリ)/ita-root/webroot/menus/sheets' fstype='nfs' pcs resource create menus_users Filesystem device='[Route53に登録したEFSの名前解決設定の情報]:/(ITAインストール先ディレクトリ)/ita-root/webroot/menus/users' directory='/(ITAインストール先ディレクトリ)/ita-root/webroot/menus/users' fstype='nfs'	赤字は、環境に合わせて読み替えること

構築手順

実行ユーザ／接続条件
rootユーザで実行すること rootユーザがsshログイン可能であること

No.	作業項目	サーバ		設定内容 （※master、slaveの両方が対象の場合は、並行して設定してください。）	備考
		master	slave		
7-18	EFSリソースグループ設定をする	●		pcs resource group add efsmount symphony conductor ansible_driver ita_sessions webconfs_sheets webconfs_users temp uploadfiles webroot_uploadfiles menus_sheets menus_users	
7-19	各kyファイルリソース設定をする	●		pcs resource create ky_activedirectory_roleuser_replication-workflow systemd:ky_ansible_execute-workflow ¥ op monitor interval=60 timeout=100 ¥ op start interval=0s timeout=100 ¥ op stop interval=0s timeout=100 pcs resource create ky_ansible_execute-workflow systemd:ky_ansible_execute-workflow ¥ op monitor interval=30 timeout=60 ¥ op start interval=0s timeout=60 ¥ op stop interval=0s timeout=60 pcs resource create ky_ansible_towermasterSync-workflow systemd:ky_ansible_towermasterSync-workflow ¥ op monitor interval=30 timeout=60 ¥ op start interval=0s timeout=60 ¥ op stop interval=0s timeout=60 pcs resource create ky_bulk_excel-workflow systemd:ky_bulk_excel-workflow ¥ op monitor interval=30 timeout=60 ¥ op start interval=0s timeout=60 ¥ op stop interval=0s timeout=60 pcs resource create ky_change_col_to_row systemd:ky_change_col_to_row ¥ op monitor interval=30 timeout=60 ¥ op start interval=0s timeout=60 ¥ op stop interval=0s timeout=60 pcs resource create ky_cmdbmenuanalysis-workflow systemd:ky_cmdbmenuanalysis-workflow ¥ op monitor interval=30 timeout=60 ¥ op start interval=0s timeout=60 ¥ op stop interval=0s timeout=60 pcs resource create ky_create_er-workflow systemd:ky_create_er-workflow ¥ op monitor interval=30 timeout=60 ¥ op start interval=0s timeout=60 ¥ op stop interval=0s timeout=60 pcs resource create ky_create_param_menu_execute systemd:ky_create_param_menu_execute ¥ op monitor interval=30 timeout=60 ¥ op start interval=0s timeout=60 ¥ op stop interval=0s timeout=60 pcs resource create ky_data_portability_execute-workflow systemd:ky_data_portability_execute-workflow ¥ op monitor interval=30 timeout=60 ¥ op start interval=0s timeout=60 ¥ op stop interval=0s timeout=60 pcs resource create ky_hostgroup_check_loop systemd:ky_hostgroup_check_loop ¥ op monitor interval=30 timeout=60 ¥ op start interval=0s timeout=60 ¥ op stop interval=0s timeout=60 pcs resource create ky_hostgroup_split systemd:ky_hostgroup_split ¥ op monitor interval=30 timeout=60 ¥ op start interval=0s timeout=60 ¥ op stop interval=0s timeout=60	

構築手順

実行ユーザ／接続条件
rootユーザで実行すること rootユーザがsshログイン可能であること

No.	作業項目	サーバ		設定内容 （※master、slaveの両方が対象の場合は、並行して設定してください。）	備考
		master	slave		
7-19	各kyファイルリソース設定をする （前頁の続き）			pcs resource create ky_legacy_role_valautostup-workflow systemd:ky_legacy_role_valautostup-workflow ¥ op monitor interval=30 timeout=60 ¥ op start interval=0s timeout=60 ¥ op stop interval=0s timeout=60 pcs resource create ky_legacy_role_varsautolistup-workflow systemd:ky_legacy_role_varsautolistup-workflow ¥ op monitor interval=30 timeout=60 ¥ op start interval=0s timeout=60 ¥ op stop interval=0s timeout=60 pcs resource create ky_legacy_valautostup-workflow systemd:ky_legacy_valautostup-workflow ¥ op monitor interval=30 timeout=60 ¥ op start interval=0s timeout=60 ¥ op stop interval=0s timeout=60 pcs resource create ky_legacy_varsautolistup-workflow systemd:ky_legacy_varsautolistup-workflow ¥ op monitor interval=30 timeout=60 ¥ op start interval=0s timeout=60 ¥ op stop interval=0s timeout=60 pcs resource create ky_pioneer_valautostup-workflow systemd:ky_pioneer_valautostup-workflow ¥ op monitor interval=30 timeout=60 ¥ op start interval=0s timeout=60 ¥ op stop interval=0s timeout=60 pcs resource create ky_pioneer_varsautolistup-workflow systemd:ky_pioneer_varsautolistup-workflow ¥ op monitor interval=30 timeout=60 ¥ op start interval=0s timeout=60 ¥ op stop interval=0s timeout=60 pcs resource create ky_std_checkcondition-linklist systemd:ky_std_checkcondition-linklist ¥ op monitor interval=30 timeout=60 ¥ op start interval=0s timeout=60 ¥ op stop interval=0s timeout=60 pcs resource create ky_std_synchronize-Conductor systemd:ky_std_synchronize-Conductor ¥ op monitor interval=30 timeout=60 ¥ op start interval=0s timeout=60 ¥ op stop interval=0s timeout=60 pcs resource create ky_std_synchronize-regularly systemd:ky_std_synchronize-regularly ¥ op monitor interval=30 timeout=60 ¥ op start interval=0s timeout=60 ¥ op stop interval=0s timeout=60 pcs resource create ky_std_synchronize-regularly2 systemd:ky_std_synchronize-regularly2 ¥ op monitor interval=30 timeout=60 ¥ op start interval=0s timeout=60 ¥ op stop interval=0s timeout=60 pcs resource create ky_std_synchronize-symphony systemd:ky_std_synchronize-symphony ¥ op monitor interval=30 timeout=60 ¥ op start interval=0s timeout=60 ¥ op stop interval=0s timeout=60 pcs resource create ky_terraform_execute-workflow systemd:ky_terraform_execute-workflow ¥ op monitor interval=30 timeout=60 ¥ op start interval=0s timeout=60 ¥ op stop interval=0s timeout=60	

構築手順

実行ユーザ／接続条件
rootユーザで実行すること rootユーザがsshログイン可能であること

No.	作業項目	サーバ		設定内容 （※master、slaveの両方が対象の場合は、並行して設定してください。）	備考
		master	slave		
7-19	各kyファイルリソース設定をする （前頁の続き）			pcs resource create ky_terraform_checkcondition-workflow systemd:ky_terraform_checkcondition-workflow ¥ op monitor interval=30 timeout=60 ¥ op start interval=0s timeout=60 ¥ op stop interval=0s timeout=60 pcs resource create ky_terraform_varsautolistup-workflow systemd:ky_terraform_varsautolistup-workflow ¥ op monitor interval=30 timeout=60 ¥ op start interval=0s timeout=60 ¥ op stop interval=0s timeout=60 pcs resource create ky_terraform_valautosetup-workflow systemd:ky_terraform_valautosetup-workflow ¥ op monitor interval=30 timeout=60 ¥ op start interval=0s timeout=60 ¥ op stop interval=0s timeout=60	
7-20	kyリソースのグループ化を実施する	●		pcs resource group add ky_services ky_activedirectory_roleuser_replication-workflow ky_ansible_execute-workflow ky_ansible_towermasterSync-workflow ky_bulk_excel-workflow ky_change_col_to_row ky_cmdbmenuanalysis-workflow ky_create_er-workflow ky_create_param_menu_execute ky_data_portability_execute-workflow ky_hostgroup_check_loop ky_hostgroup_split ky_legacy_role_valautostup-workflow ky_legacy_role_varsautolistup-workflow ky_legacy_valautostup-workflow ky_legacy_varsautolistup-workflow ky_pioneer_valautostup-workflow ky_pioneer_varsautolistup-workflow ky_std_checkcondition-linklist ky_std_synchronize-Conductor ky_std_synchronize-regularly ky_std_synchronize-regularly2 ky_std_synchronize-symphony ky_terraform_execute-workflow ky_terraform_checkcondition-workflow ky_terraform_varsautolistup-workflow ky_terraform_valautosetup-workflow	
7-21	これまでに作成したリソースを指定し、リソースグループ起動/停止の順序を指定する	●		pcs constraint order set efsmount ky_services exastro	
7-22	これまでに設定したリソースが同ノードで起動する制限を指定する	●		pcs constraint colocation set efsmount ky_services exastro	
7-23	HAソフトのサービスを開始する	●	●	systemctl enable corosync systemctl enable pacemaker systemctl start corosync systemctl start pacemaker	

構築手順

実行ユーザ／接続条件
rootユーザで実行すること rootユーザがsshログイン可能であること

No.	作業項目	サーバ		設定内容 （※master、slaveの両方が対象の場合は、並行して設定してください。）	備考
		master	slave		
7-24	クラスターのステータスを確認する	●	●	pcs status (表示例) Cluster name: aws-cluster Cluster Summary: ＊Stack: corosync ＊Current DC: <ita-ha-sv01のホスト名> (version 2.0.5-9.el8_4.1-ba59be7122) - partition with quorum ＊Last updated: Thu Jun 10 06:46:16 2021 ＊Last change: Wed Jun 9 18:39:29 2021 by root via cibadmin on <ita-ha-sv01のホスト名> ＊2 nodes configured ←左記の表示であること ＊34 resource instances configured Node List: ＊Online: [<ita-ha-sv01のホスト名> <ita-ha-sv02のホスト名>] ←master,slaveがOnlineであること Full List of Resources: ＊Resource Group: exastro: ＊eip (ocf::heartbeat:awseip): Started <ita-ha-sv01のホスト名> ←masterがStartedであること ＊httpd (systemd:httpd): Started <ita-ha-sv01のホスト名> ＊Resource Group: efsmount: ＊symphony (ocf::heartbeat:Filesystem): Started <ita-ha-sv01のホスト名> ＊conductor (ocf::heartbeat:Filesystem): Started <ita-ha-sv01のホスト名> ＊ansible_driver (ocf::heartbeat:Filesystem): Started <ita-ha-sv01のホスト名> ＊ita_sessions (ocf::heartbeat:Filesystem): Started <ita-ha-sv01のホスト名> ＊webconfs_sheets (ocf::heartbeat:Filesystem): Started <ita-ha-sv01のホスト名> ＊webconfs_users (ocf::heartbeat:Filesystem): Started <ita-ha-sv01のホスト名> ＊temp (ocf::heartbeat:Filesystem): Started <ita-ha-sv01のホスト名> ＊uploadfiles (ocf::heartbeat:Filesystem): Started <ita-ha-sv01のホスト名> ＊webroot_uploadfiles (ocf::heartbeat:Filesystem): Started <ita-ha-sv01のホスト名> ＊menus_sheets (ocf::heartbeat:Filesystem): Started <ita-ha-sv01のホスト名> ＊menus_users (ocf::heartbeat:Filesystem): Started <ita-ha-sv01のホスト名> ＊Resource Group: ky_services: ～～～～～～～～～～～～～～～～～～～～～省略～～～～～～～～～～～～～～～～～～～～～ Daemon Status: corosync: active/enabled pacemaker: active/enabled pcsd: active/enabled	赤字は、環境に合わせて読み替える
7-25	ITAの接続を確認する	●		以下のURLより、ログイン画面にアクセスする http(s)://(ElasticIP)	赤字は、環境に合わせて読み替える
7-26	MariaDBの削除を実施する	●	●	yum list installed | grep -i mariadb yum remove mariadb.x86_64 mariadb-backup.x86_64 mariadb-common.x86_64 mariadb-connector-c.x86_64 mariadb-connector-c-config.noarch mariadb-errmsg.x86_64 mariadb-gssapi-server.x86_64 mariadb-server.x86_64 mariadb-server-utils.x86_64	前項の手順について、ITAへ正常にアクセスが可能であることを、確認してから削除を行う。

作業終了