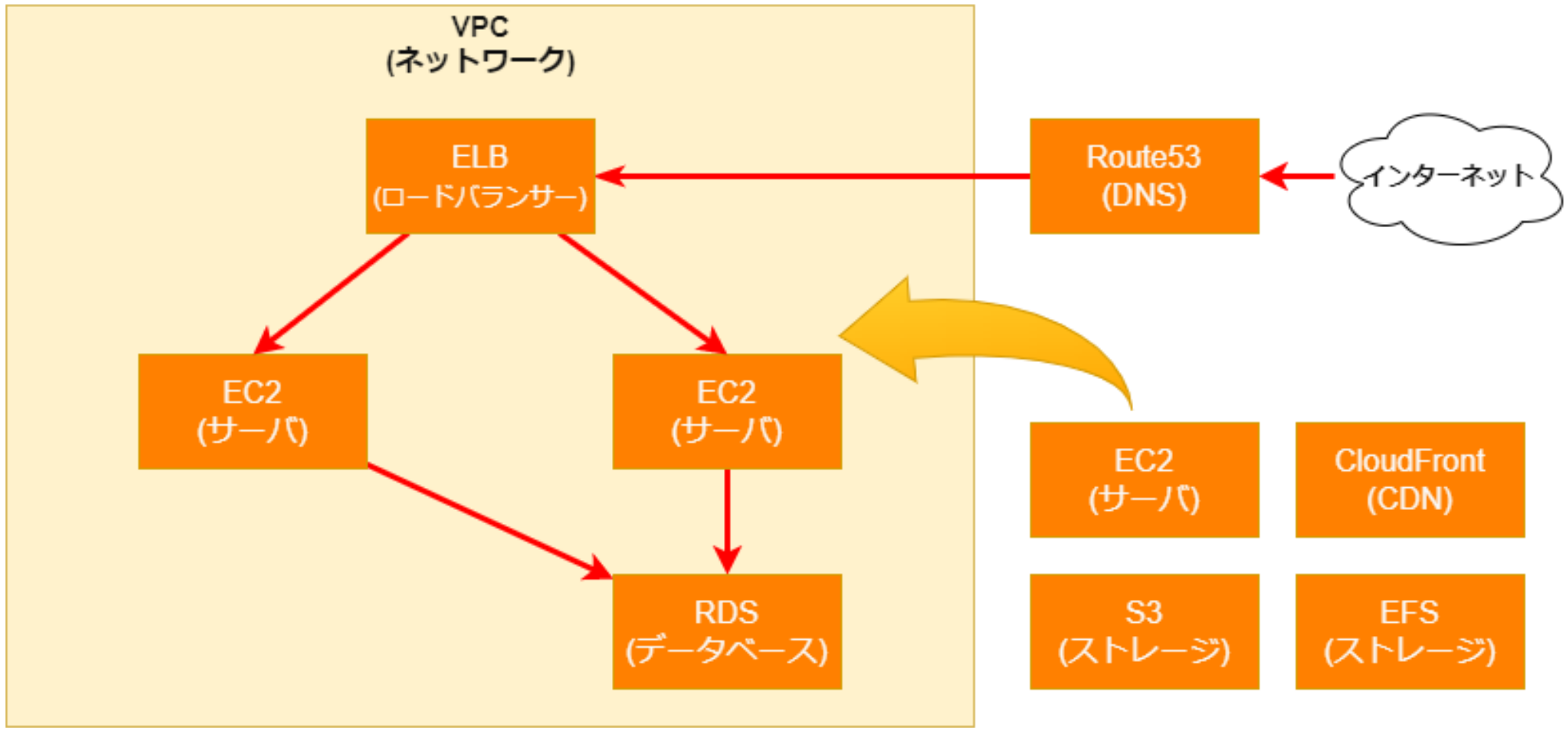
実用的なWordPressを構築してみよう

AWSはどんなことができるの？

サーバやデータベース、ストレージ等をブロックパーツのように
オンライン上で自分の好きなように構成を実現できる



本日のお題

実用的なWordPress環境を構築する

- ✓ 単一AZ障害への耐性があること
- ✓ 管理画面、テーマ有効化、画像添付が問題なく利用出来ること
- ✓ 負荷上昇に対して自動でスケールすること

構成案を考えよう

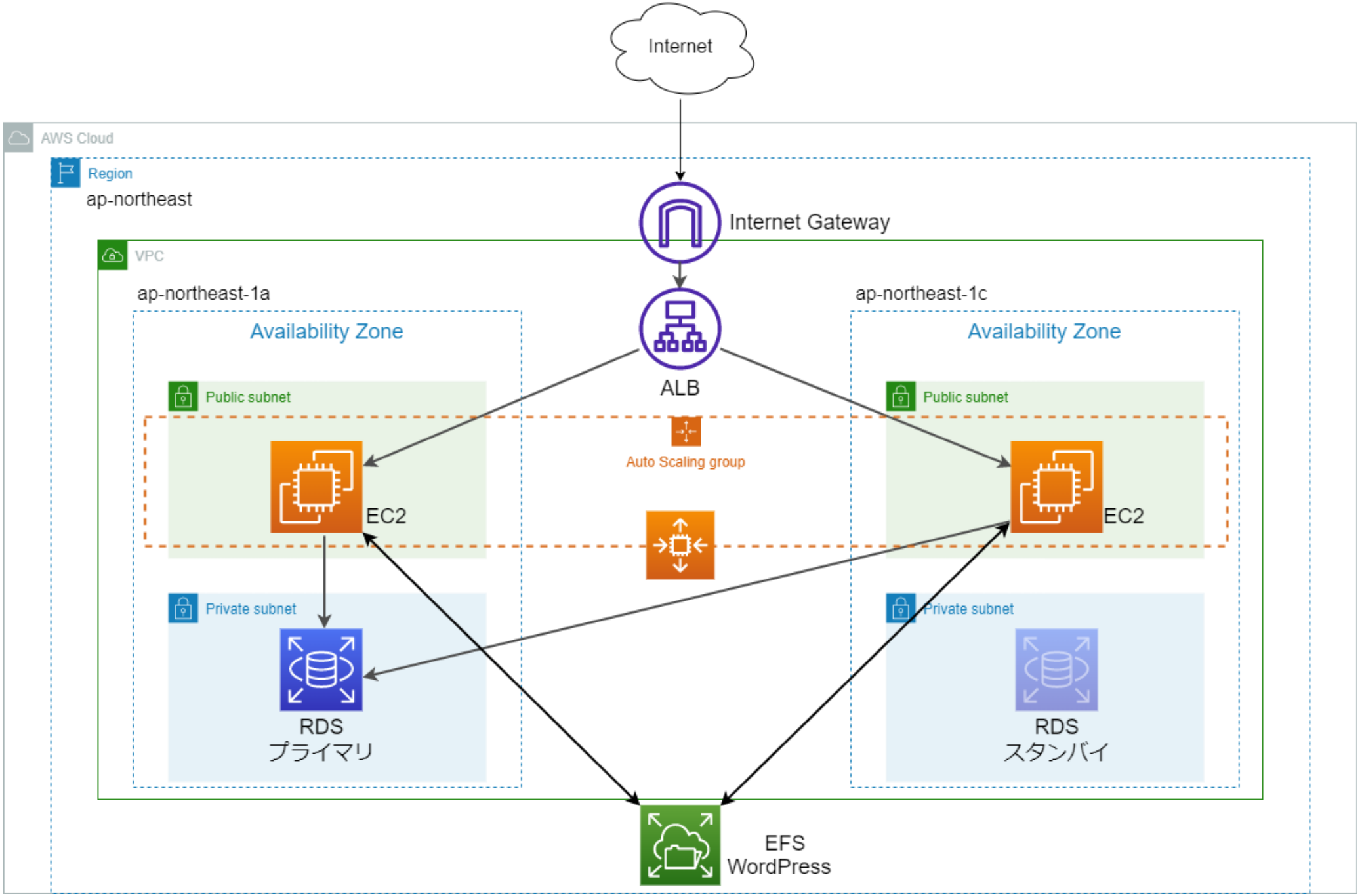
構成案

ALB + EC2 + AutoScaling + RDS + EFS

役割

- **ALB** : ロードバランサー。 WEBサーバの負荷を分散してくれる
- **EC2** : AWS上の仮想サーバ。今回はWEBサーバとして利用する
- **AutoScaling** : 負荷状況に応じて自動でリソースを増減する
- **RDS** : Relational Database Service マネージド型データーベース
- **EFS** : Elastic File System ファイル型ストレージ。データ共有用

構成図を考えよう


Internet
AWS Cloud
Region
ap-northeast
Internet Gateway
VPC
ap-northeast-1a
ap-northeast-1c
Availability Zone
Availability Zone
ALB
Public subnet
Public subnet
Auto Scaling group
EC2
EC2
Private subnet
Private subnet
RDS
プライマリ
RDS
スタンバイ
EFS
WordPress

簡単な用語説明

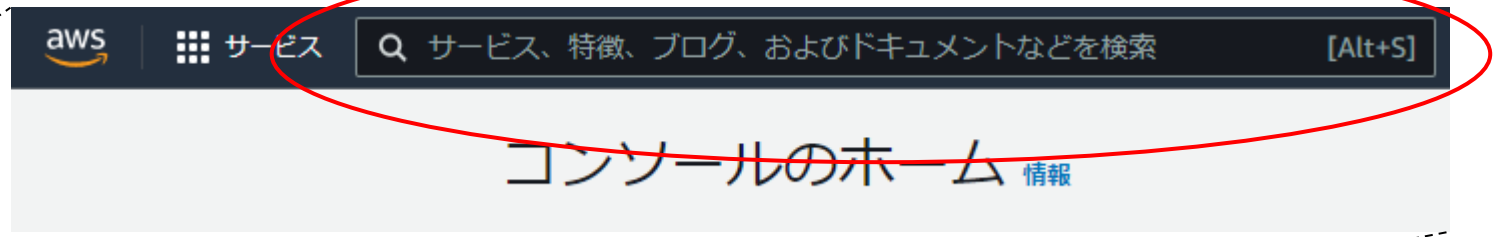
- リージョン
  - 直訳で地域や領域。AWSは世界各地からサービスを立てることができる
- AZ(Availability Zone)
  - リージョン内をさらに物理的に分割された領域
- VPC(Virtual Private Cloud)
  - プライベート範囲な枠組み。その中にサーバ等を立てる
- サブネット
  - リージョン＞AZ＞VPC＞サブネットとさらにIP範囲を細分化させ自由な構成を作れるようにする

STEP1 ～ 構築してみよう ～

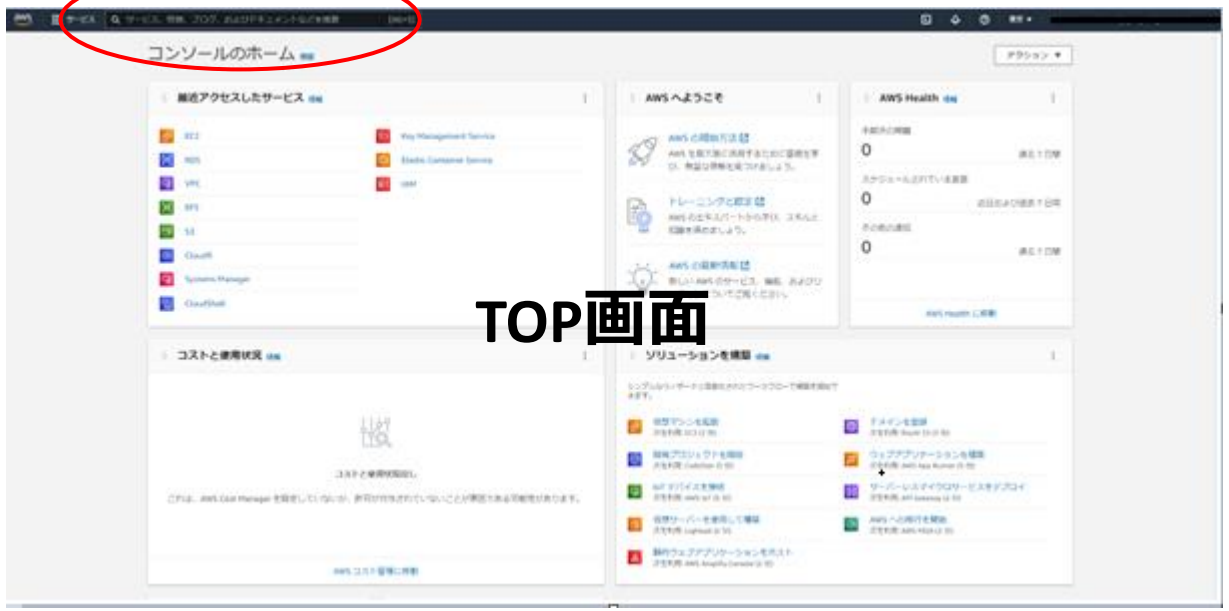
- AWSアカウントログイン
  - 画面右上のアカウント→設定 から言語設定ができます

準備OK ？

AWSで各サービスを使うための基本的な動作を抑えよう！



サービスを検索する



TOP画面

検索結果の候補がでます。
目的のものをクリック！

本講座で使う主な項目

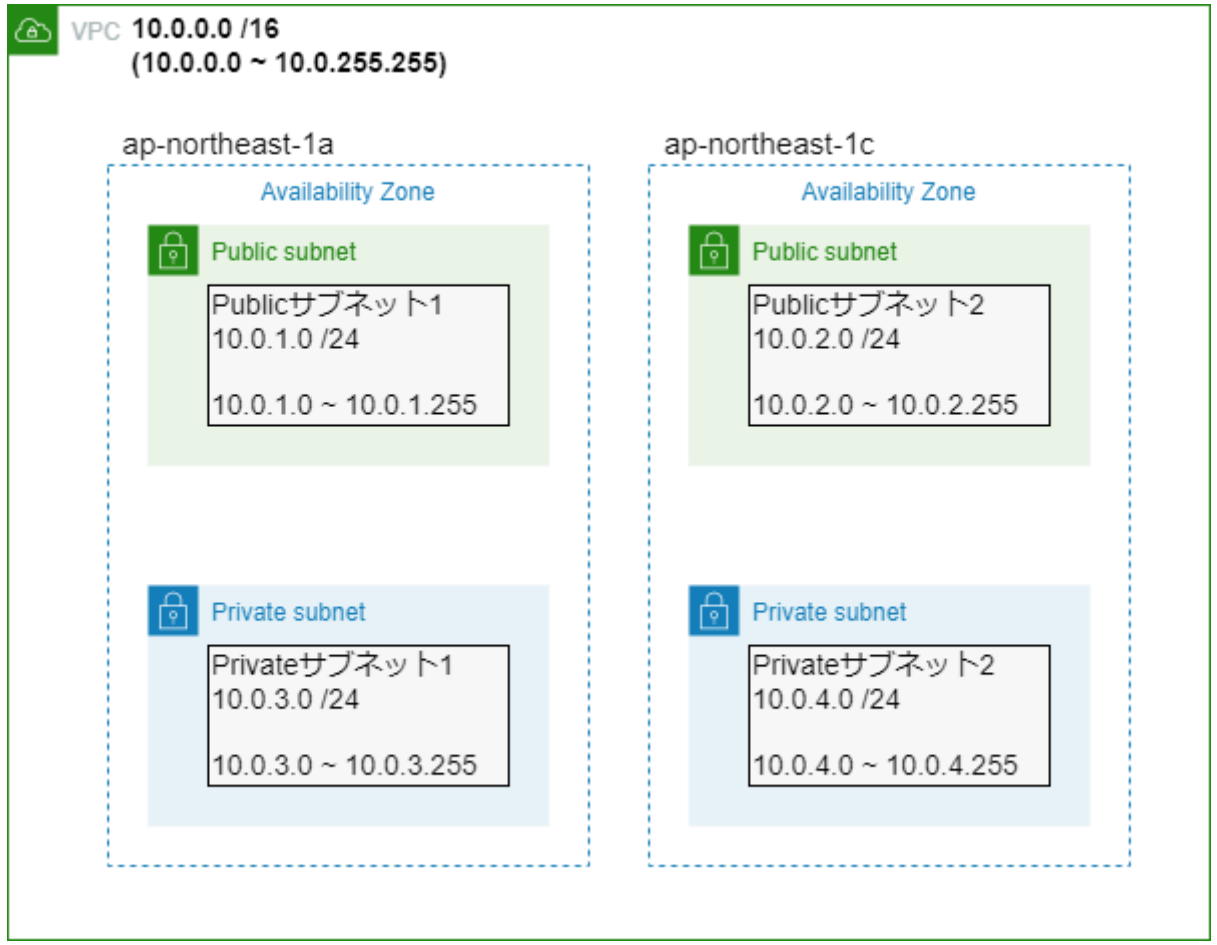
各サービスの
左メニューに項目があるヨ

- VPCサービス
  - VPC
  - サブネット
  - ルートテーブル
  - インターネットゲートウェイ
  - セキュリティグループ
- EFSサービス
  - ファイルシステム
- EC2サービス
  - インスタンス
  - AMI
  - セキュリティグループ
  - キーペア
  - ロードバランシング
    - ロードバランサー
    - ターゲットグループ
  - AutoScaling
    - 起動設定
    - AutoScalingグループ
- RDSサービス
  - データベース
  - サブネットグループ

サーバ(EC2)作成！ その前に・・

“サーバ設置場所” を先に作る！ リージョン、VPC、サブネットなど

サブネットの範囲とは



VPCという枠の範囲(IPアドレス)を決め
その中でさらにEC2等のサーバーを設置する
サブネットを決めていく。

サブネットの作成時点では
Public、Privateという分別はありません。
作成後、

Publicサブネットのルートテーブルを編集
↓
ルートテーブルに
インターネットゲートウェイ追加
↓
外部と通信できる**Publicサブネット**となる

STEP2 ～ ネットワークの範囲を作成 ～ 1/4

- **リージョンを決める**
  - 画面右上のアカウント横にある(初期はバージニア北部)中から選ぶ
  - 今回は**アジアパシフィック(東京)**を選んでください (**ap-northeast-1**)
- **VPCサービスへ移動**
  - 画面上部の検索バーから「VPC」と検索
  - VPC を作成 **押下**
  - VPCの設定
    - **VPCのみ** を選択
    - 名前タグ: 任意 (名前-vpcなど)
    - IPv4 CIDR: **10.0.0.0/16**
    - 他はデフォルトのままでOK
  - VPCを作成
  - 作成したVPC選択→アクション→「DNSホスト名編集」→ **有効化チェックして保存**
- **左メニュー: インターネットゲートウェイ →** インターネットゲートウェイの作成 **押下**
  - 名前: [Name]-igw → インターネットゲートウェイの作成 押下
  - アクション → VPCにアタッチ → VPC選択

STEP2 ～ ネットワークの範囲を作成 ～ 2/4

- セキュリティグループ → セキュリティグループを作成 押下
  - 表の上から順に作成してください
  - 表の項目以外はデフォルトでOK ※アウトバンドルールもそのまま！
  - VPC欄はVPC名を入力すると候補がでます

ルールを追加から
始めてください

名前	説明	VPC	インバウンドルール
[Name]-sg-alb-web	ALB用	作成したVPC	タイプ: HTTP (80番) , ソース: [カスタム] 0.0.0.0/0
[Name]-sg-efs	EFS用	作成したVPC	タイプ: NFS (2049番) , ソース: [カスタム] [Name]-sg-ec2 ※ EC2用のセキュリティグループを選択
[Name]-sg-ec2	EC2用	作成したVPC	タイプ: HTTP (80番) , ソース: [カスタム] 0.0.0.0/0 タイプ: SSH (22番) , ソース: [マイIP] 表示されたIP
[Name]-sg-db	RDS用	作成したVPC	タイプ: MYSQL/Aurora (3306番) , ソース: [カスタム] [Name]-sg-ec2 ※ EC2用のセキュリティグループを選択

STEP2 ～ ネットワークの範囲を作成 ～ 3/4

- サブネット → サブネットを作成 押下
  - VPCID: 作成したVPCを選択
  - 下記の表を基に作成
    - 1つできたら「新しいサブネット追加」で**まとめて4つ作成**

サブネット名	アベイラビリティゾーン	IPv4 CIDR ブロック
[Name]-**public-sub**-1a	ap-northeast-1a	10.0.1.0/24
[Name]-**public-sub**-1c	ap-northeast-1c	10.0.2.0/24
[Name]-**private-sub**-1a	ap-northeast-1a	10.0.3.0/24
[Name]-**private-sub**-1c	ap-northeast-1c	10.0.4.0/24

Point: サブネット名は**public** or **private**と**AZ**が判別できるようにするとよい

STEP2 ～ ネットワークの範囲を作成 ～ 4/4

- **Publicサブネットの自動割り当てを有効にする**
  - [Name]-**public-**sub-1a と [Name]- **public-**sub-1c の2つが対象
  - 作成したPublicサブネットを選択 → 右上のアクション → サブネット設定を編集
  - パブリックIPv4アドレスの自動割り当てを有効にチェックして保存

STEP3 ～ ルートテーブルの設定 ～ 1/4

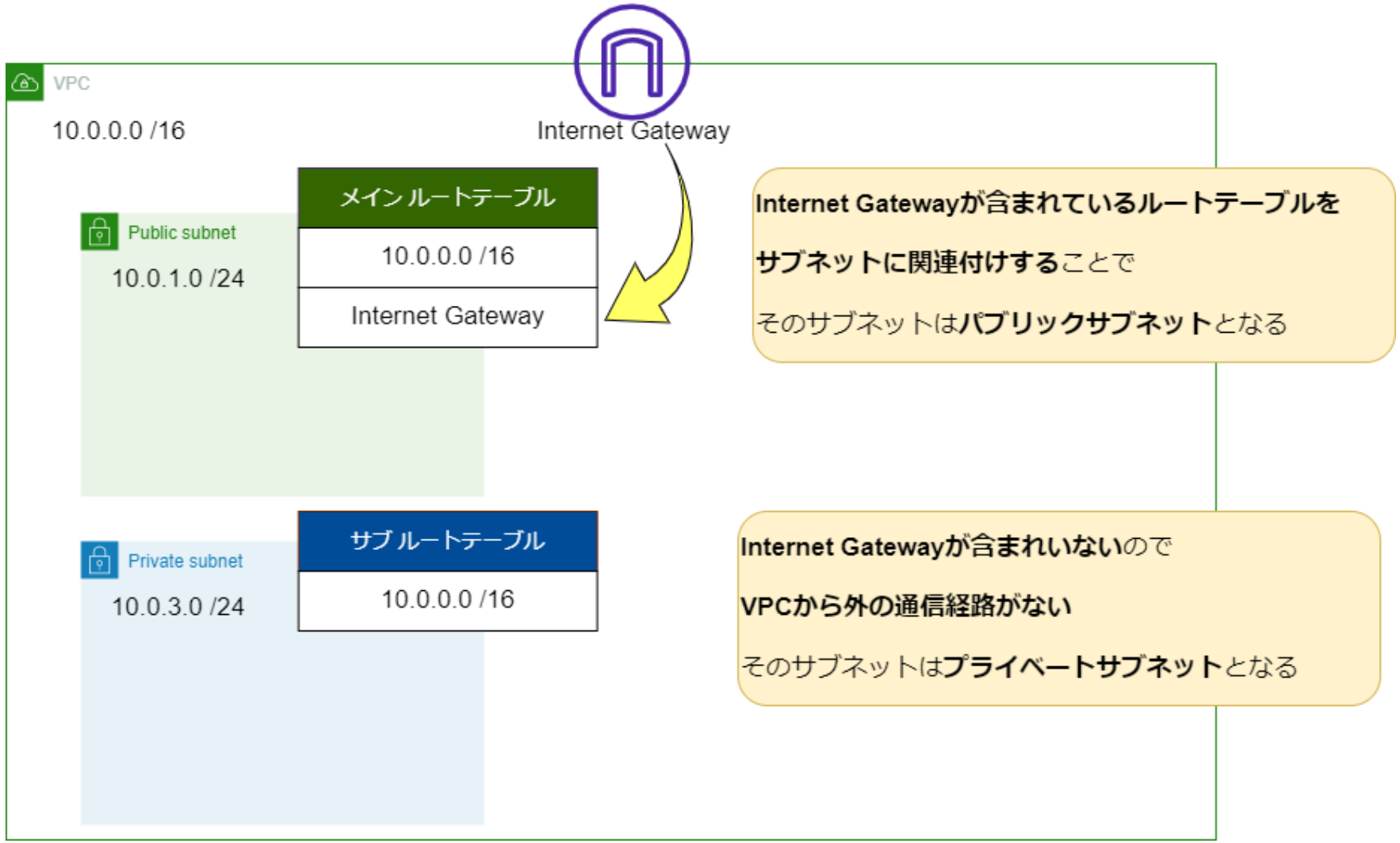
STEP2にて**public**と**private**サブネット2つずつ作りましたが
この時点ではどちらも外部疎通できない不完全な状態です。
Nameだけpublicと宣言しているだけ。

STEP3でやること…

- Internet Gatewayが含まれているルートテーブル＋サブネット
  → 外部と通信可能な**publicサブネットを作る**

- Internet Gatewayが含まれていないルートテーブル＋サブネット
  → 外部とは直接通信できない**privateサブネットを作る**

STEP3 ～ ルートテーブルの設定 ～ 2/4



STEP3 ～ ルートテーブルの設定 ～ 3/4

- **サブネットへ移動する**
  - 作成したパブリックサブネット→ 詳細タブ
  - ルートテーブルのリンクをクリック

- **ルートタブからルートを編集**
  - ルートを追加→ ターゲット→ インターネットゲートウェイ選択
    - 全部で2つ
      1. 送信先: 10.0.0.0/16 ターゲット: local (デフォルトであるはず)
      2. 送信先: 0.0.0.0/0
         ターゲット: インターネットゲートウェイ→自分が作成したものを選択
    - 保存
- **サブネットの関連付けタブ選択**
  - **明示的な**サブネットの関連付けを編集
    - **public**サブネット2つ選択して保存
- **パブリック用のルートテーブル完成**

STEP3 ～ ルートテーブルの設定 ～ 4/4

- **プライベート用のサブネットを作る**
  - ルートテーブル → ルートテーブルを作成 押下
    - 名前: [Name]-private-rt
    - VPC: 作成したVPCを選択
- **サブネットの関連付けタブ選択**
    - **明示的な**サブネットの関連付けを編集
      - **private**サブネット2つ選択
- **プライベート用ルートテーブル完成**

ひと息いれましょう！

EC2などを設置する為の環境構築はこれで完了です。
次からサーバなどリソースを作成・構築していきます！

※ 次ステップから料金発生が伴います。
　時間が怪しい場合は日を改めて続きからでも大丈夫です。

STEP4 ～ EFSを作成しよう ～ 1/2

- 画面左上からEFSを検索して移動
- ファイルシステムの作成 押下
  - カスタマイズ を選択
  - 名前: [Name]-efs
  - 他はデフォルトのまま次へ
  - 作成したVPCを選択する
  - サブネットID: **Publicサブネットをそれぞれ選択**
  - セキュリティグループ: デフォルトを消して**EFS用**を選択
  - 次へ
  - 以降はデフォルトのまま最後まで進んで作成する

STEP4 ～ EFSを作成しよう ～ 2/2

- 作成したEFSを選択
  - 「アタッチ」から[EFS マウントヘルパーの使用]のコマンドを控える

控えたら閉じてSTEP4完了

STEP5 ～ EC2 を作成しよう ～ 1/2

- **左上からEC2と検索して移動**
- インスタンスを起動 ▼ **押下 (候補が出たら「インスタンスを起動」選択)**
  - 名前とタグ: 任意
  - AMI: クイックスタート → Amazon Linux (Amazon Linux2)

  - インスタンスタイプ: **t2micro**
  - 新しいキーペアを作成
    - 任意の名前をつけローカルPC内に秘密鍵を保存 ※保存場所忘れないように
    - キーペア名以外はデフォルトのままでOK

STEP5 ～ EC2 を作成しよう ～ 2/2

- ネットワーク設定
  - ネットワーク設定の右にある「**編集**」を押下
  - VPCは作成したものを選択
  - サブネット: [Name]-public-sub-1a
- パブリックIPの自動割り当て: **有効化**
- ファイアウォール(セキュリティグループ)
  - 「既存のセキュリティグループを選択する」を選択
  - EC2用に作成したものを選択

- 他はデフォルトのままでOK
- インスタンスを起動
- インスタンス一覧から「実行中」が確認できればOK

STEP6 ～ RDSを作成しよう ～ 1/3

- **画面左上からRDSを検索して移動**
- **サブネットグループ →** DB サブネットグループを作成 **押下**
  - 名前: 任意 , 説明: 任意(名前と同じで良い)
  - 作成したVPCを選択
  - アベイラビリティゾーン: ap-northeast-**1a**と**1c** 2つ選択
  - サブネット: **Private**サブネット2つ選択(10.0.3.0/24 , 10.0.4.0/24)
  - 作成

STEP6 ～ RDSを作成しよう ～ 2/3

- **データベース → データベースの作成 押下**
  - データベース作成方法
    - 標準作成を選択
  - エンジンのオプション
    - MySQLを選択 (バージョン MySQL 8.0.27)
  - テンプレート
    - 開発/テスト (✕本番稼働用は高い！)
  - 可用性と耐久性
    - **マルチ AZ DBクラスター**
  - 設定
    - DB インスタンス識別子: 任意 (DBインスタンス名です)
    - マスターユーザー名
      - パスワード: 自動生成でも任意でもOK ※後で使います

STEP6 ～ RDSを作成しよう ～ 3/3

- インスタンスの設定
  - バースト可能クラス: **db.t4g.micro**
- ストレージタイプ
    - **汎用SSD(gp2)** / 20GiB (✕プロビジョンドIOPS選ぶと すごく高い！！！)
- 接続
  - 作成したVPC、サブネットグループを選択
  - セキュリティグループ: RDS用を選択 (デフォルトは削除)
- 他はデフォルトのままでOK
- データベースの作成 押下

※作成完了まで１５分程度かかります

※ 注意パスワード自動生成選んだ方

デーベースの作成押下後、まっさきに画面右上の
「認証情報の詳細を表示」 必ず確認＆控えてください！！
画面が遷移すると消えてPWが見れなくなります！！

STEP6 ～ RDSを作成しよう ～ 3/3

- インスタンスの設定
  - バースト可能クラス: **db.t4g.micro** or **db.t3.micro**
- ストレージタイプ
  - 作成したVPC、サブネットグループを選択
- 接続
  - 作成したVPC、サブネットグループを選択
  - 作成したデーターベース用のセキュリティグループを選択
- 他はデフォルトのままでOK
- データベースの作成 押下

※作成完了まで１５分程度かかります

作成完了をまたずに次のステップに進んでしまいましょう！

※ 注意パスワード自動生成選んだ方

デーベースの作成押下後、画面右上の
「認証情報の詳細を表示」 必ず確認＆控えてください！！
他のページに移ったりすると消えてしまいます！！

STEP7 ～ EC2へログインしてみよう～ 1/2

- Public IPアドレスからSSHログイン
  - ユーザ: ec2-user
  - 認証方法: 公開鍵
    - EC2を作成した時に保存した秘密鍵を選択

EC2サービス移動して
作成したEC2選択 → 詳細タブ
にPublic IPが記載されています。

- 好きなTerminalでOK
  - MacPCのターミナルから直接SSHログインする場合
    - デフォルトだと鍵の権限を絞らないとログインできない場合があります
    - $ chmod 600 鍵ファイル
    - $ ssh -i 鍵ファイル ec2-user@IPアドレス

STEP7 ～ EC2へログインしてみよう ～ 2/2

- SSHターミナル使わない場合の対応
  - IAM へ移動
    - ロール作成
      - ユースケース: EC2
      - [AmazonSSMManagedInstanceCore]でフィルタしたものをチェック
      - ロール名: 任意
  - EC2へ移動　IAMから移動した際はリージョン要確認！
    - 作成したインスタンスを選択 → アクション → セキュリティ → IAMロール変更
    - 作成したIAMロールを選択して保存
- 接続する
  - インスタンス選択して[接続] → セッションマネージャー → 接続

STEP8 ～ 構築してみよう ～ 1/3

▼ rootへ昇格

```
# sudo su -
```

テキストエディタ等に
コピペしてから使用してください。

コピーミスに気を付けてください。

▼ 必要なミドルウェアをインストール

```
# amazon-linux-extras install epel -y
# yum -y install https://rpms.remirepo.net/enterprise/remi-release-7.rpm
# yum -y install httpd mysql ncurses-compat-libs
# yum -y install --enablerepo=remi-php74 --disablerepo=amzn2-core php php-mysql
```

▼ その他 (後ほど使います)

```
# curl -L -O https://rpmfind.net/linux/dag/redhat/el7/en/x86_64/dag/RPMS/stress-1.0.2-1.el7.rf.x86_64.rpm
# rpm -ivh stress-1.0.2-1.el7.rf.x86_64.rpm
# rpm -qa | egrep "httpd|mysql|ncurses|php|stress"
```

▼ WordPress ダウンロードと設定

```
# curl -L -O https://ja.wordpress.org/latest-ja.zip
# unzip latest-ja.zip
# mv -iv wordpress /var/www/html/
# chown -R apache:apache /var/www/html/wordpress
```

STEP8 ～ 構築してみよう ～ 2/3

▼ httpd.conf修正

cp -ipv /etc/httpd/conf/httpd.conf /etc/httpd/conf/.httpd.conf.org

sed -i '/^DocumentRoot/s/DocumentRoot "¥/var¥/www¥/html"/DocumentRoot "¥/var¥/www¥/html¥/wordpress"/' /etc/httpd/conf/httpd.conf

sed -i '/^<Directory/s/Directory "¥/var¥/www¥/html"/Directory "¥/var¥/www¥/html¥/wordpress"/' /etc/httpd/conf/httpd.conf

▼ 差分を確認

diff -U0 /etc/httpd/conf/.httpd.conf.org /etc/httpd/conf/httpd.conf

```
[root@ip-10-0-1-163 ~]# diff -U0 /etc/httpd/conf/.httpd.conf.org /etc/httpd/conf/httpd.conf
--- /etc/httpd/conf/.httpd.conf.org     2021-12-30 21:39:01.000000000 +0000
+++ /etc/httpd/conf/httpd.conf  2022-04-26 12:20:46.162761124 +0000
@@ -119 +119 @@
-DocumentRoot "/var/www/html"
+DocumentRoot "/var/www/html/wordpress"
@@ -131 +131 @@
-<Directory "/var/www/html">
+<Directory "/var/www/html/wordpress">
[root@ip-10-0-1-163 ~]#
```

STEP8 ～ 構築してみよう ～ 3/3

<STEP6のRDSが作成完了していることを確認してから実施してください>

▼ RDSへ接続してwordpress用のデータベース、ユーザ等を作成

mysql -h RDSのエンドポイント –uユーザ名 –p

※ RDSエンドポイントの記載場所

→ 作成したRDSをクリック → 接続とセキュリティタブ → エンドポイント があります

赤字の部分を変えてね

> create database wp;

> create user 'wpuser'@'10.0.%' identified by 'パスワード';

> grant all on wp.* to 'wpuser'@'10.0.%';

> exit

▼ Apache起動

systemctl enable httpd

systemctl start httpd

データベース名: wp
ユーザ名: wpuser
パスワード: 任意

WordPress初期設定で使います！

STEP9 ～ WordPress管理画面の設定 ～

- ブラウザのアドレスバーへEC2のPublicIPを貼り付けて検索
- WordPress 初期設定
  - STEP8 で設定したデータベース情報をいれます
    - データベース名: wp
    - ユーザ名: wpuser
    - パスワード: 設定したもの
    - データベースホスト名: RDSのエンドポイント
    - テーブル接頭辞: そのまま
  - サイトのタイトル: 任意
  - ユーザ名: 任意
  - パスワード: 任意
  - メールアドレス: ご自分の会社アドレスなど
  - 検索に表示しないのチェックをいれておく(念のため程度で入れてもよい)

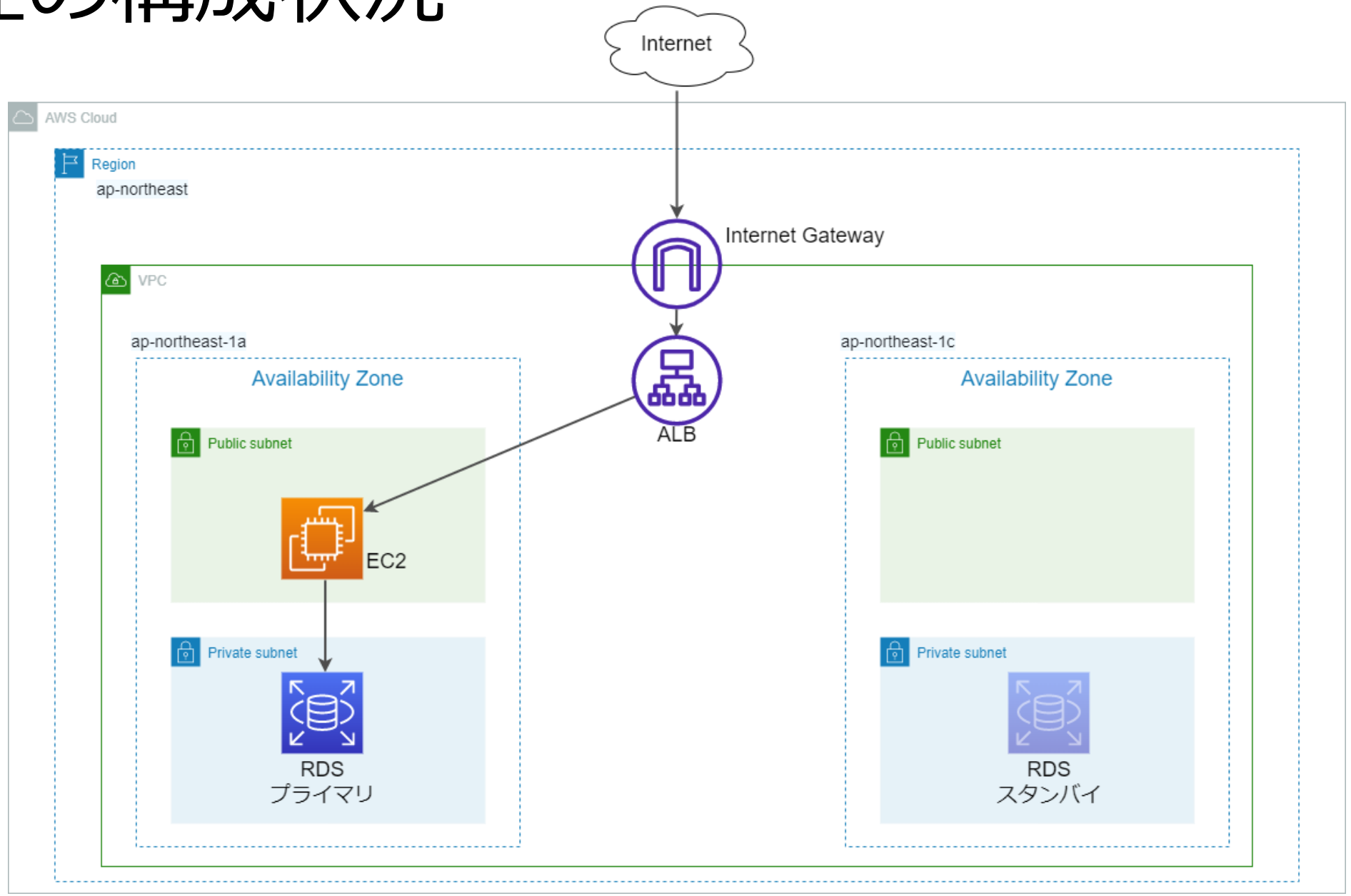
STEP10 ～ WordPressを見てみよう ～

- 好きなテーマをインストールしてみよう
  - 外観 → テーマ → 新規追加 → 好きなの選んで有効化
- 画像をアップロードしてみよう
  - メディア → ライブラリ → 新規追加 → ローカルPCからドラッグ＆ドロップ

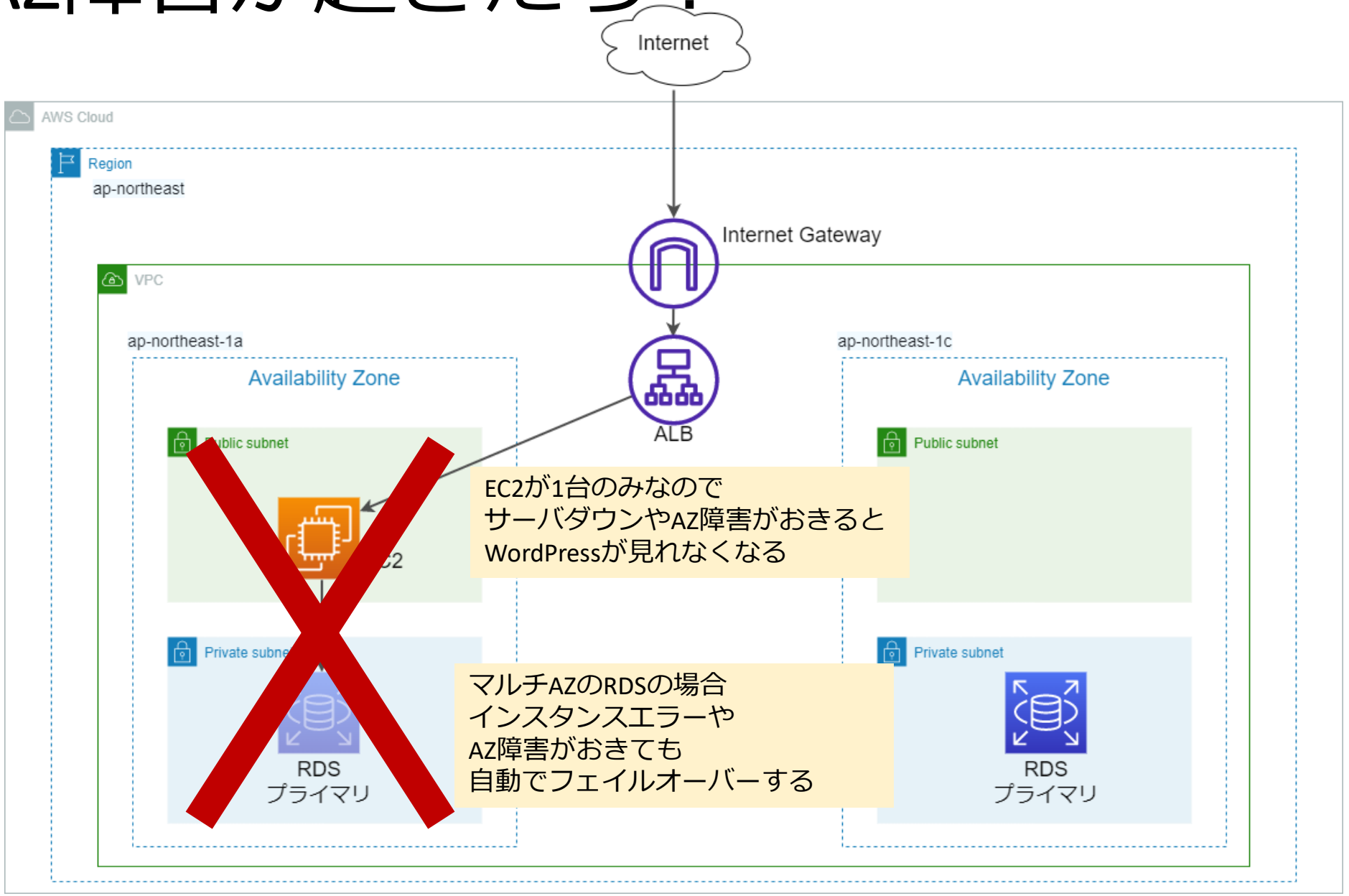
- 投稿を更新する
  - 投稿 → Hello world！ (お好きにどうぞ！)
- 投降したページを確認してみる
  - アドレスバーにEC2のPublic IPを入力し検索してみる (新しいタブなどがよいです)
  - ページが見れていればOK！

簡易ですがネット世界に放たれました！
投稿する内容やアップする画像は
見られても恥ずかしくないものにしておきましょう

現在の構成状況


Internet
AWS Cloud
Region
ap-northeast
Internet Gateway
VPC
ap-northeast-1a
Availability Zone
ALB
ap-northeast-1c
Availability Zone
Public subnet
EC2
Public subnet
Private subnet
RDS
プライマリ
Private subnet
RDS
スタンバイ

もしAZ障害が起きたら？



もしAZ障害が起きたら？

冗長構成を組んでみよう

STEP11 ～ EFSをマウントしよう ～

▼インストール

yum -y install amazon-efs-utils

▼ マウント前

df -h

▼ マウントする STEP4 で控えたコマンドを実行 ※若干修正あり

mkdir /efs

mount -t efs -o tls ファイルシステムID:/ マウント先ディレクトリ ※ マウント先 /efs

例 赤文字が修正するところ
mount -t efs -o tls fs-0efb6ea64e1830a45:/ /efs

▼マウント後

df -h

▼ サーバ起動時に自動でマウントする

cp -ipv /etc/fstab /etc/.fstab.org

sed -i '2a ファイルシステムID:/ /efs efs nofail,_netdev 0 0' /etc/fstab

▼差分の確認

diff -U0 /etc/.fstab.org /etc/fstab

```
[root@ip-10-0-1-163 ~]# diff -U0 /etc/.fstab.org /etc/fstab
--- /etc/.fstab.org     2022-04-26 12:50:02.974602805 +0000
+++ /etc/fstab  2022-04-26 12:50:11.646621815 +0000
@@ -2,0 +3 @@
+fs-0e611ae93f0409245:/ /efs efs nofail,_netdev 0 0
[root@ip-10-0-1-163 ~]#
```

STEP12 ～ EFSへコンテンツを移す ～ 1/2

▼ EFSへコンテンツ移動

mkdir /efs/html

mv -i /var/www/html/wordpress /efs/html

※ コマンド完了するまで少し時間かかります

chown -R apache:apache /efs/html/wordpress

▼ httpd.conf修正

cp -ipv /etc/httpd/conf/httpd.conf /etc/httpd/conf/.httpd.conf.`date +%Y%m%d`

sed -i '/^DocumentRoot/s/DocumentRoot "¥/var¥/www¥/html¥/wordpress"/DocumentRoot "¥/efs¥/html¥/wordpress"/' /etc/httpd/conf/httpd.conf

sed -i '/^<Directory/s/Directory "¥/var¥/www¥/html¥/wordpress"/Directory "¥/efs¥/html¥/wordpress"/' /etc/httpd/conf/httpd.conf

▼ 差分を確認

diff -U0 /etc/httpd/conf/.httpd.conf.`date +%Y%m%d` /etc/httpd/conf/httpd.conf

```
@@ -119 +119 @@
-DocumentRoot "/var/www/html/wordpress"
+DocumentRoot "/efs/html/wordpress"
@@ -131 +131 @@
-<Directory "/var/www/html/wordpress">
+<Directory "/efs/html/wordpress">
```

STEP12 ～ EFSへコンテンツを移す ～ 2/2

▼ 起動順を変える

```
# cp -ipv /usr/lib/systemd/system/httpd.service
/usr/lib/systemd/system/.httpd.service.`date +%Y%m%d`
# sed -i '/^After/s/$/ efs.mount/' /usr/lib/systemd/system/httpd.service
```

▼差分確認

```
# diff -U0 /usr/lib/systemd/system/.httpd.service.`date +%Y%m%d`
/usr/lib/systemd/system/httpd.service
```

差分の出力結果 「-」変更前 、 「+」変更後であり、変更後の最後に【efs.mount】があるか確認！ コピーペースト気を付けてね

```
-After=network.target remote-fs.target nss-lookup.target httpd-init.service
+After=network.target remote-fs.target nss-lookup.target httpd-init.service efs.mount
```

▼ 反映

```
# systemctl daemon-reload
# systemctl restart httpd
```

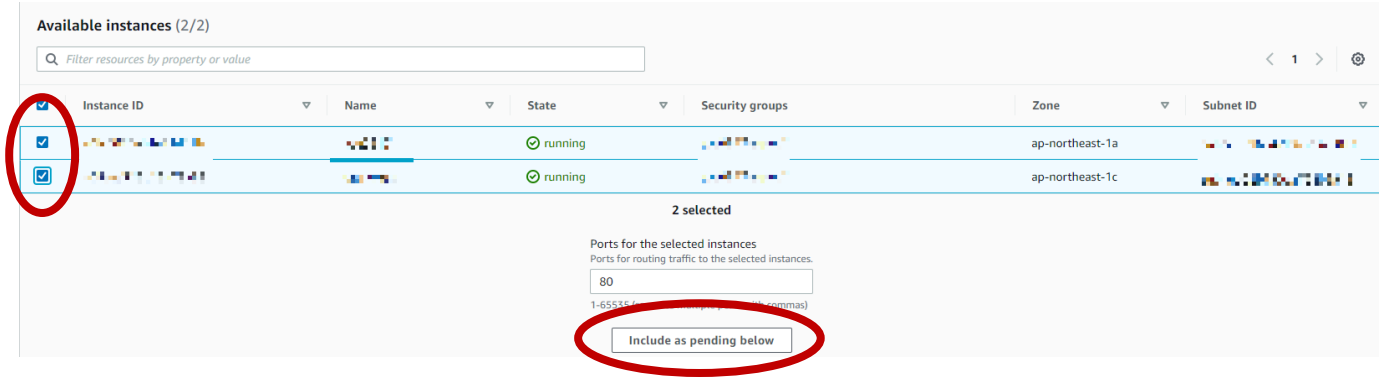
STEP13 ～ AMIを作成する ～

AMI(Amazon Machine Image)とは・・・
OS情報などを含めたEC2を作成するテンプレートみたいなもの
STEP12までの作業した情報を基に**AMI**を作成することで
簡単に同じ構成のEC2を複製することができる。

- EC2サービスへ移動
- AMI作成
  - EC2を選択 → アクション → イメージとテンプレート
    - イメージを作成
- AMI からインスタンスを起動 押下
  - 名前とタグ: 任意
  - VPC、キーペア、セキュリティグループは最初のEC2と同じ
  - サブネット: **[Name]-public-sub-1c** ※ STEP5のEC2作成で選ばなかった方

STEP14 ～ 負荷分散しよう ～ 1/3

- ターゲットグループ作成
  - ターゲットグループ → **Create target group** 押下
    - Target group name: 任意
    - 作成したVPC選択 (他項目は変更不要でNext)
    - EC2 2台選択して **Include as pending below**



    - Review targets に追加されたことを確認
    - Create target group 押下

STEP14 ～ 負荷分散しよう ～ 2/3

- ロードバランサー → ロードバランサーの作成 押下
  - Application Load Balancer → Create
  - 名前: 任意
  - 作成したVPC , AZ＋**Public**サブネットを選択
    ※ デフォルトで選択されているサブネットがprivateの場合があるので注意！
  - セキュリティグループ: ALB用を選択 (デフォルトは消す)
  - Listeners and routing : **Default action**から選択 ※候補は1つしかないはず！
  - 他はデフォルトのままでOK で Create load balancer 押下

- ALB経由で接続できるようにする
  - ターゲットグループ → 作成したターゲットグループを選択
    - AttributesタブからEdit → Stickinessをチェック(Load balancer generated cookie)
    - 変更を保存

STEP14 ～ 負荷分散しよう ～ 3/3

- EC2ログイン(どれでも)
- # mysql -h RDSのエンドポイント –uユーザ名 -p

  > use wp;

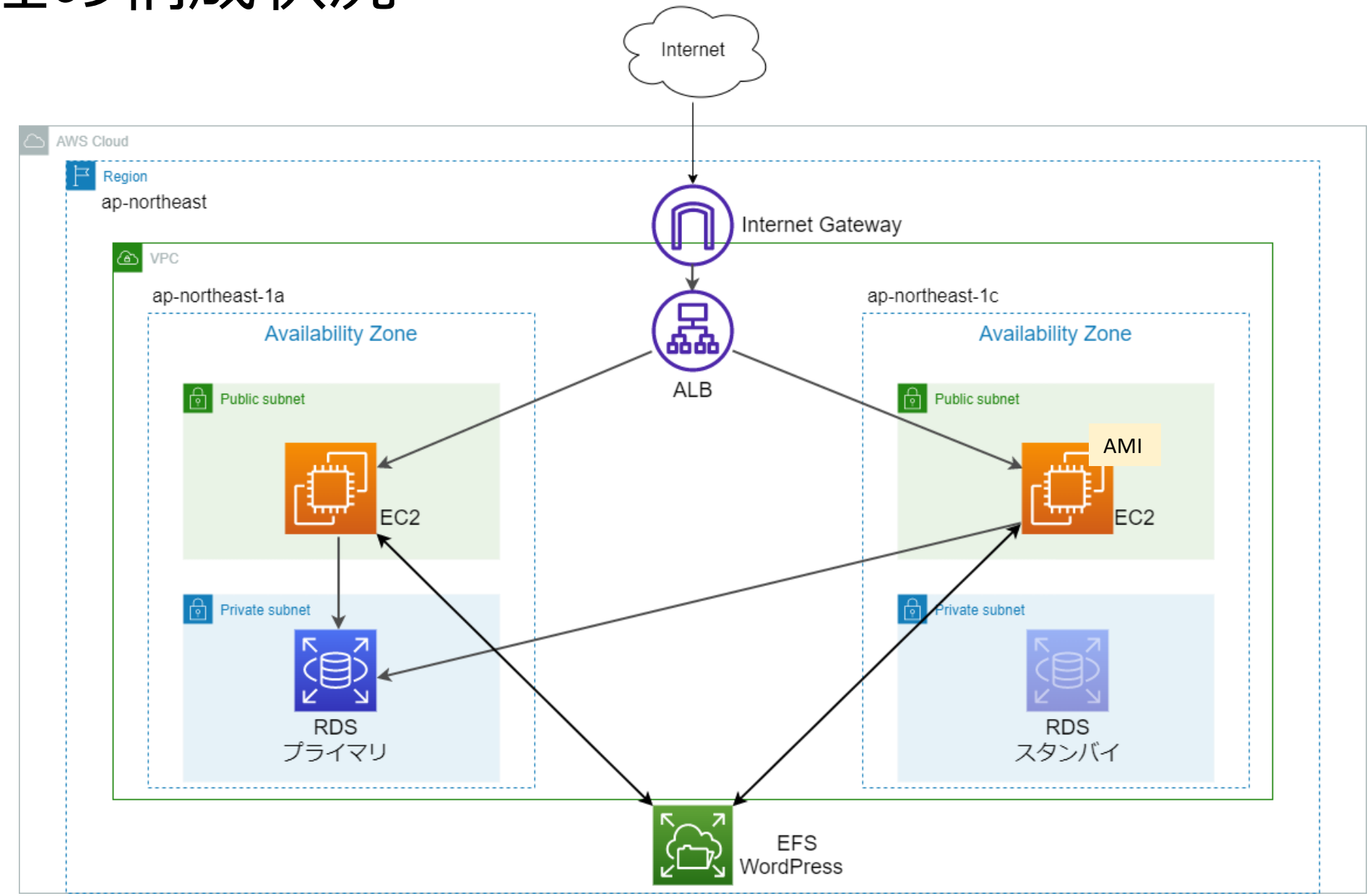
  > SELECT * FROM wp_options WHERE option_name IN ('home','siteurl');

  > UPDATE wp_options SET option_value = 'http://ALBのDNS名' where option_name IN ('home','siteurl');

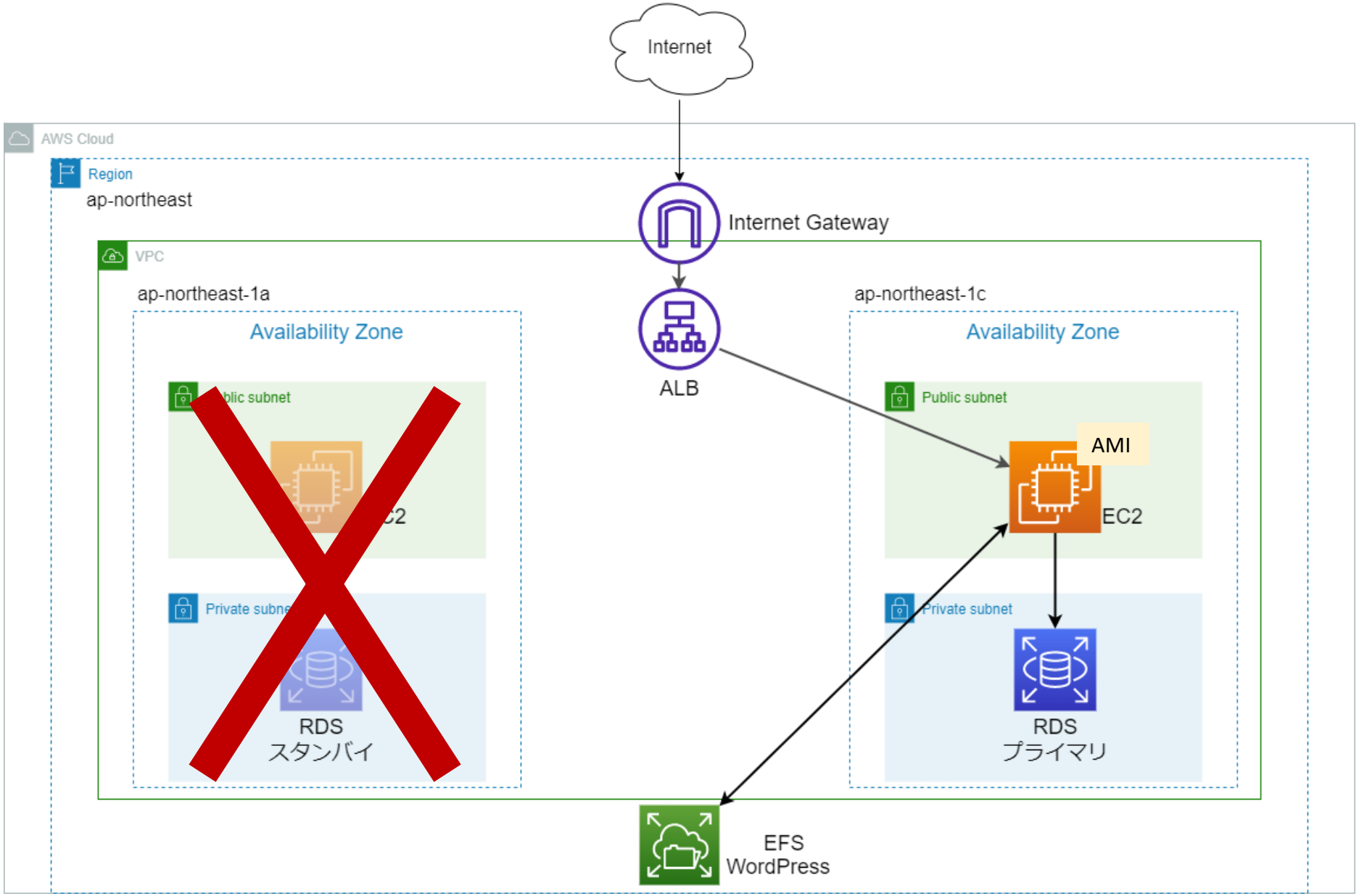
  > SELECT * FROM wp_options WHERE option_name IN ('home','siteurl');

  > exit
- 作成したALBから**DNS名**をコピー
- ブラウザアドレスバーへALBのDNS名を貼り付けて検索
  - ロードバランサー → 作成したALB選択 → 説明タブ にあります
  - WordPress画面が見れるか確認してみましょう
- 確認終わったら**EC2は2台とも停止**しておきましょう
  - 最後の後片付けで停止から**終了**します

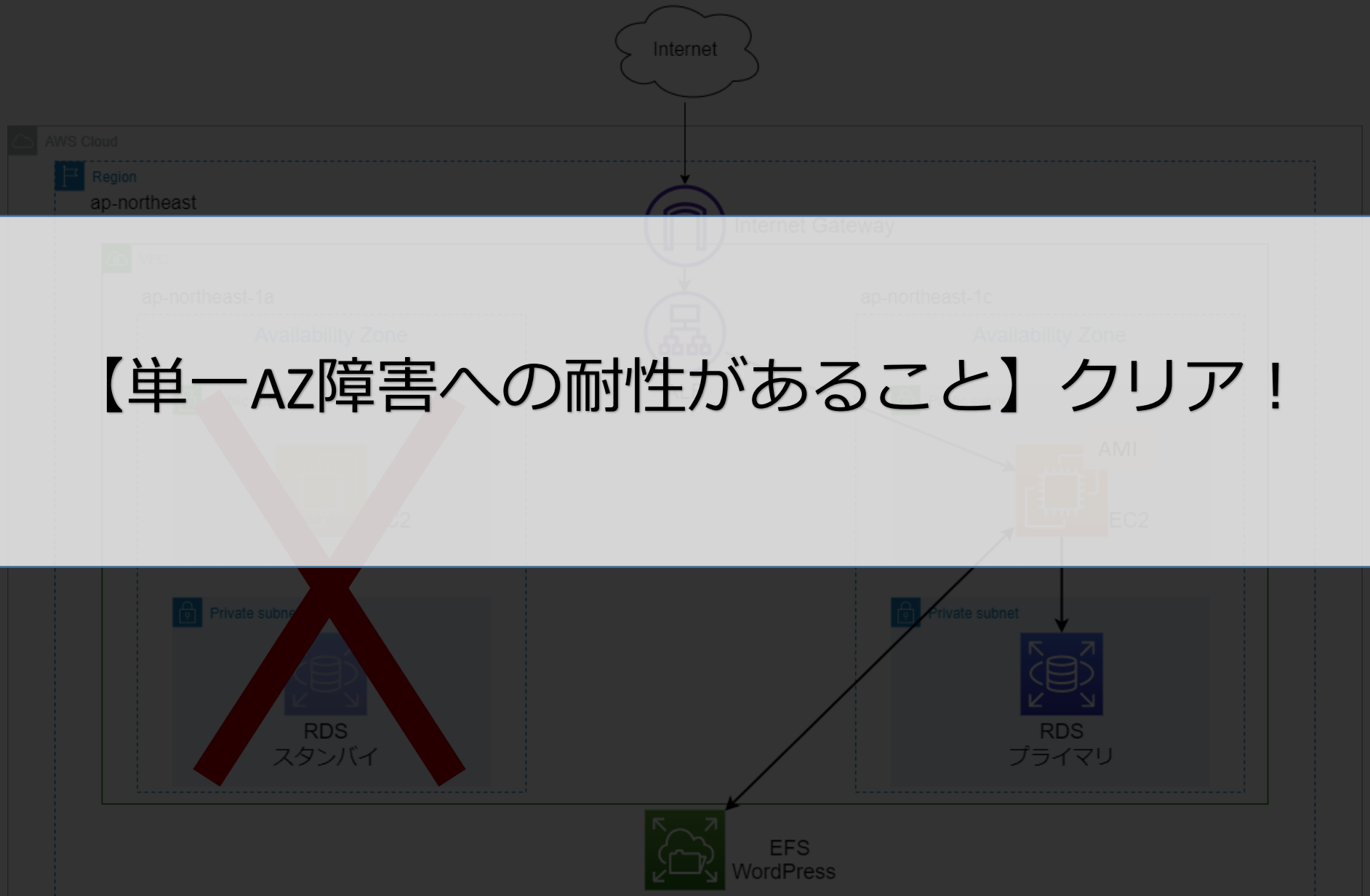
現在の構成状況


Internet
AWS Cloud
Region
ap-northeast
Internet Gateway
VPC
ap-northeast-1a
Availability Zone
ALB
ap-northeast-1c
Availability Zone
Public subnet
EC2
Public subnet
AMI
EC2
Private subnet
RDS
プライマリ
Private subnet
RDS
スタンバイ
EFS
WordPress

AZ単位や単一障害が起きても・・・



AZ単位や単一障害が起きても・・・



【単一AZ障害への耐性があること】クリア！

STEP15 ～ 自動でスケーリングする ～ 1/3

- **AutoScaling 起動設定 →** 起動設定の作成 **押下**
  - 作成したAMIを選択
  - インスタンスタイプ: t2.micro
  - WEB用のセキュリティグループ
  - 作成したキーペア
- **AutoScalingグループ →** Auto Scaling グループを作成する **押下**
  - 名前: 任意
  - 起動テンプレート→ **起動設定に切り替える**クリック → 作成した起動設定を選択
  - 作成したVPC、Publicサブネット2つ選択
  - 既存のロードバランサーにアタッチを選択
  - 既存のロードバランサーターゲットグループを選択
  - ヘルスチェック: ELBにチェック追加
  - その他の設定: モニタリングのCloudWatchチェック追加
  - 希望する容量: 2
  - 最小キャパシティ: 2
  - 最大キャパシティ: 4
  - ターゲット追跡スケーリングポリシー
  - 平均CPU使用率: 30 (デフォルトのままでもok)
  - タグ
    - キー: Name , 値: 任意

他はデフォルトのままで最後まで進めて作成を完了させてください。

STEP15 ～ 自動でスケーリングする ～ 2/3

ここまで問題なく設定できれば

STEP14最後でEC2は2台とも停止中なので稼働中は0台

- 希望する容量: 2
- 最小キャパシティ: 2

先ほど上記のように設定したので
AMIを基に自動で2台作成されるはずです！
作成が無事終わったら次の手順へ

STEP15 ～ 自動でスケーリングする ～ 3/3

- **詳細タブ → 高度な設定 → 編集**
    - 終了ポリシー: NewestInstance を選択
    - クールダウン: 60 (スケーリングによる増減が発生したら60秒間はそのまま)
    - 更新
- **ブラウザからWordPress確認**
    - ALBのDNS名でブラウザ検索してみましょう
    - テーマや画像、投降したものが反映されているか確認
    - 追加してもヨシ

【管理画面、テーマ有効化、画像添付が問題なく利用出来ること】

クリア！

STEP16 ～ スケーリングを確認しよう ～ 1/2

- **インスタンス1台を終了させてみる**
  - 任意のインスタンス選択 → インスタンスの状態 → インスタンス終了
  - 希望と最小が2台の設定で1台終了させると…？
- **サーバ負荷に対しサーバが増加するか確認**
  - サーバへログイン(どれでもよい)
    - stressによるCPU負荷をかけるコマンド

```
# stress -c 1 -q &
```

CPU使用率の平均値が閾値を上回れば3台にスケールアウトする

STEP16 ～ スケーリングを確認しよう ～ 2/2

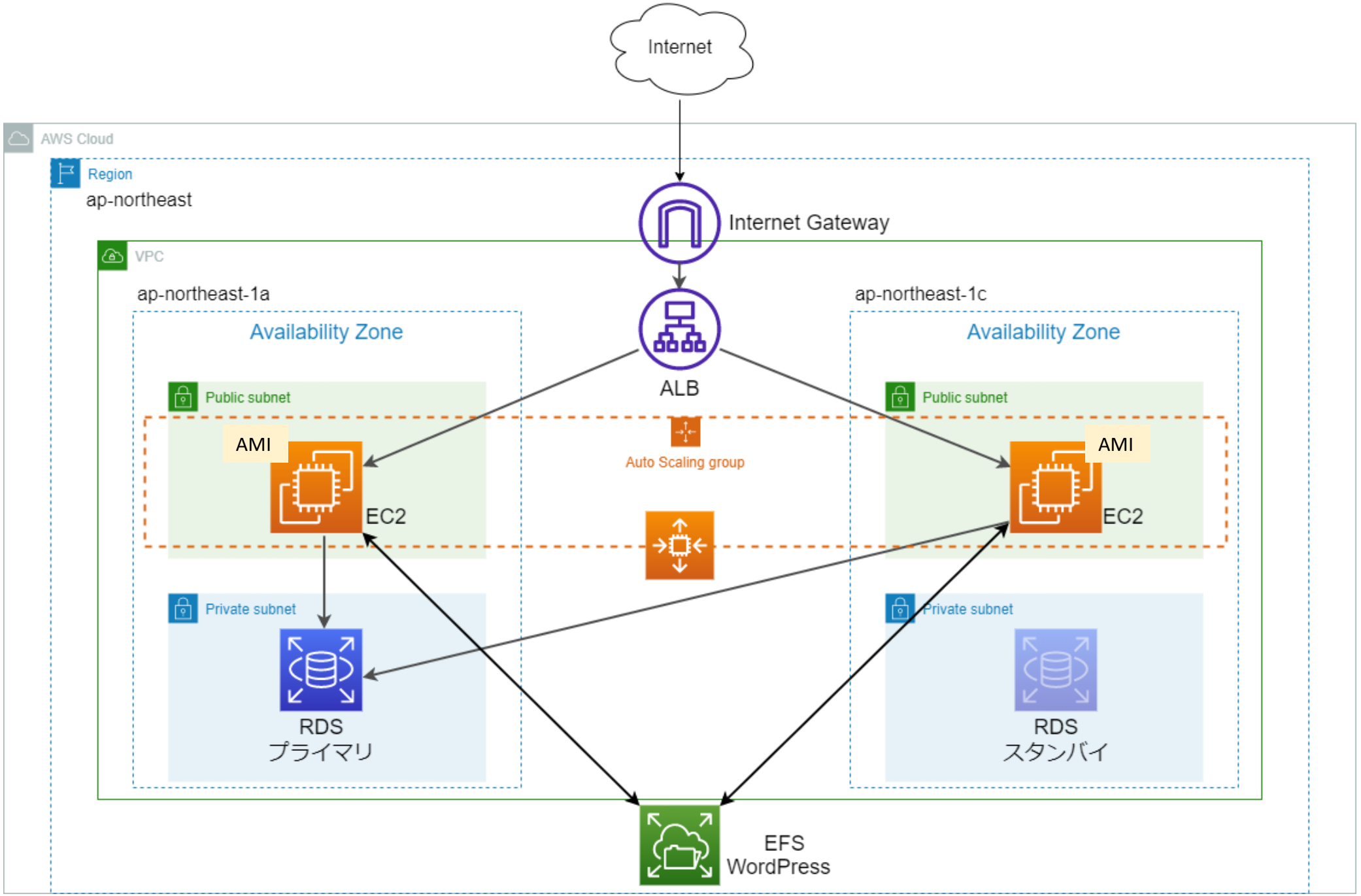
- **負荷収束後の挙動を見てみる**
  - stressコマンドをkillする = 負荷が下がる

  # pkill stress
  - プロセスが消えたことを確認

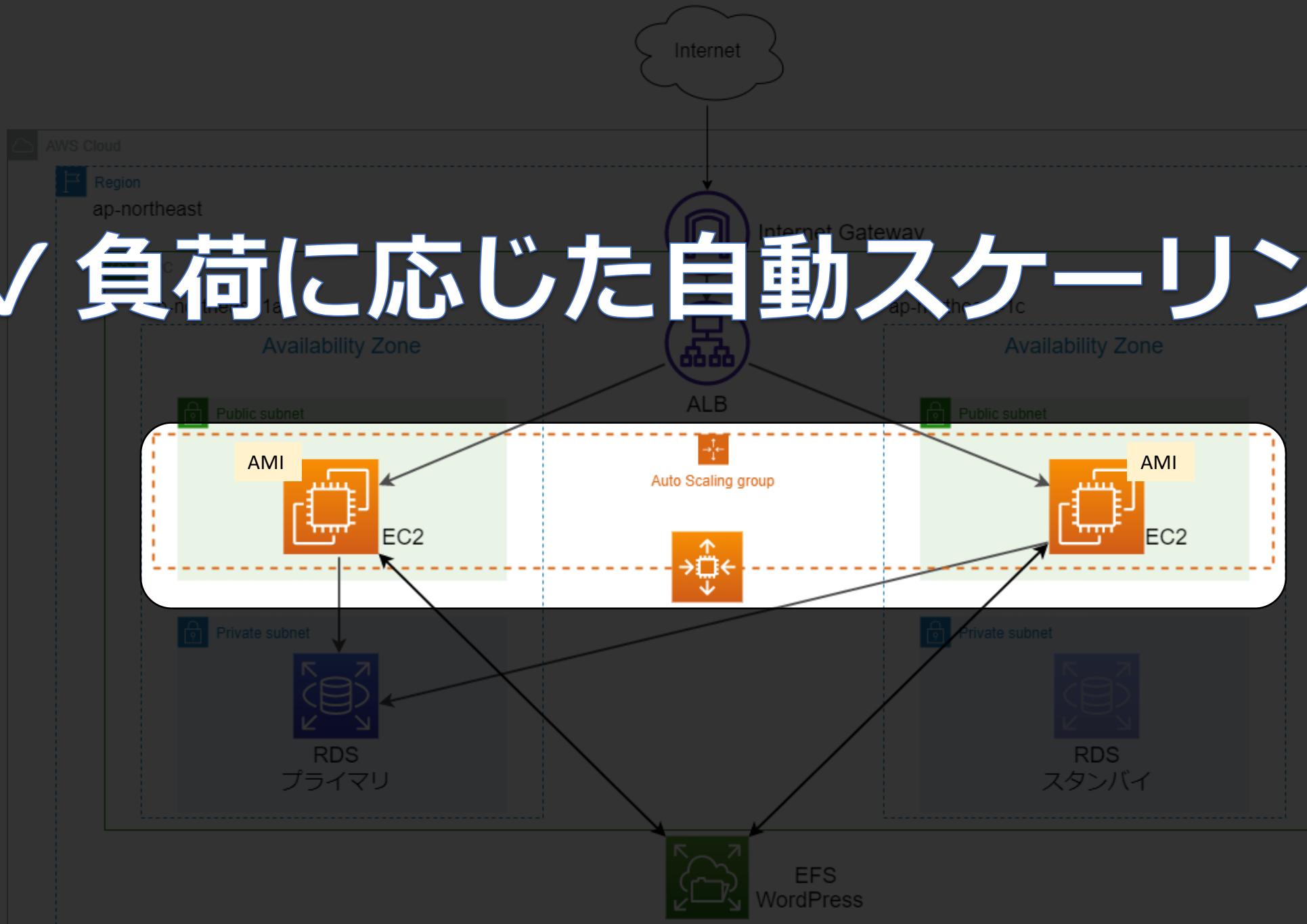
  # ps auxww |grep stress

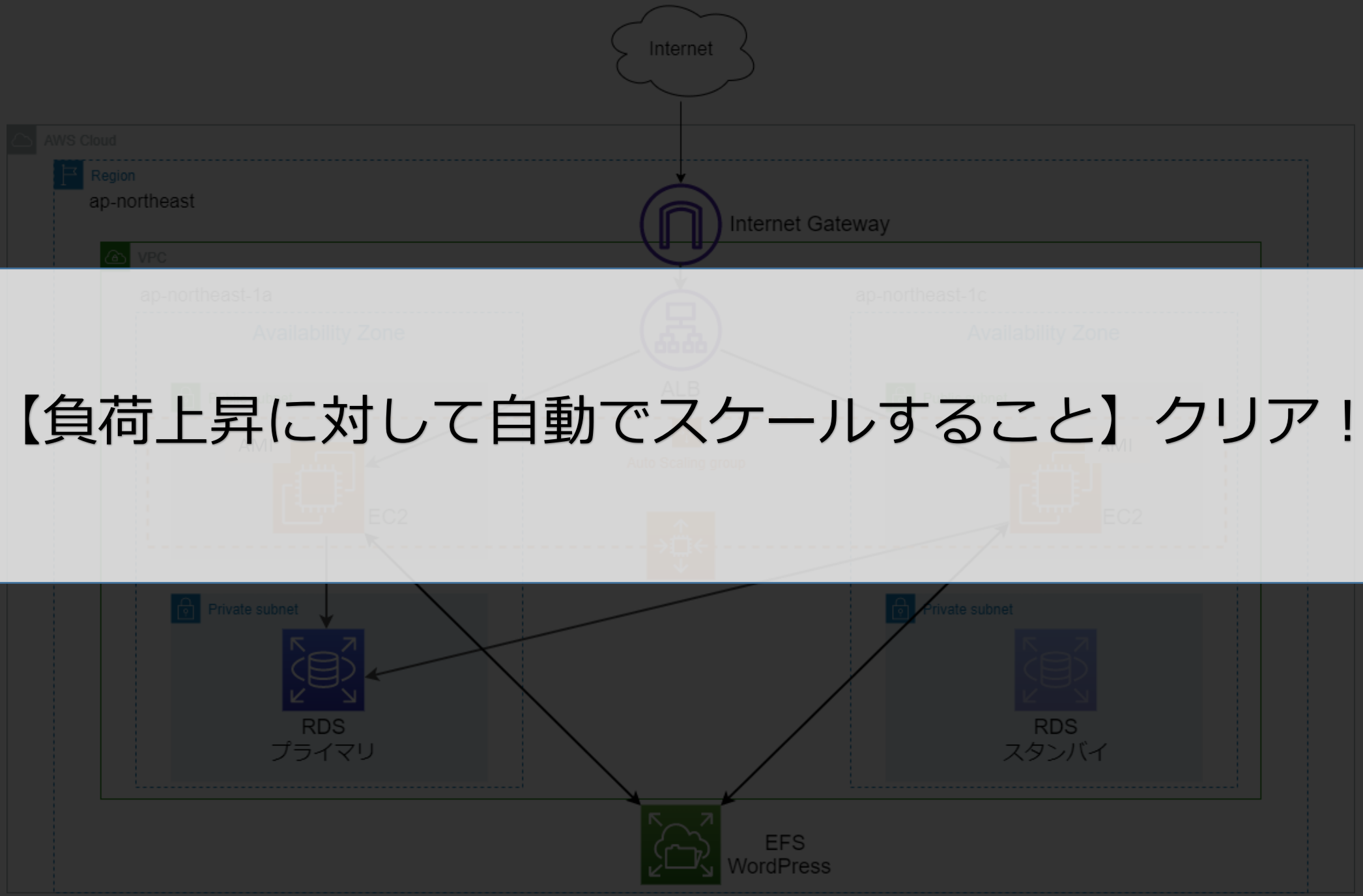
クールダウンタイムを過ぎ、CPU使用率の平均が閾値を下回れば
2台にスケールインします

※ 反映まで結構時間を要します
　希望する容量が初期2から3へ上がっていますが
　負荷収束からしばらくすると2に戻ります。


Internet
AWS Cloud
Region
ap-northeast
Internet Gateway
VPC
ap-northeast-1a
Availability Zone
ALB
ap-northeast-1c
Availability Zone
Public subnet
AMI
EC2
Auto Scaling group
Public subnet
AMI
EC2
Private subnet
RDS
プライマリ
Private subnet
RDS
スタンバイ
EFS
WordPress

✓負荷に応じた自動スケーリング





【負荷上昇に対して自動でスケールすること】クリア！

～ リソースの後片付け ～ 1/2

- AutoScalingグループを削除
  - 結構時間がかかる
  - 終わったらEC2が削除された事を確認する
- 起動設定の削除
- STEP14の最後に停止したEC2 2台とも**終了**させる
- RDSの削除
  - 作成したRDSを選択 → アクション → 削除
  - **最終スナップショットの✓を外す**
  - **Retain automated backupsの✓を外す**
    - 「インスタンスの削除後、システム～～～」の部分にチェックを入れて削除する
      - 削除完了に時間がかかるので次に進んでOK

AutoScaling消す前に
EC2終了させてもまた作成されちゃうよ

RDSは停止だけだと
7日後に自動で起動されますのでご注意

～ リソースの後片付け ～ 2/2

- EFSの削除
  - 作成したEFSを選択して画面右上の削除 押下、ファイルシステムIDを入れて削除
- ALBの削除
  - 作成したALBを選択し削除
- TargetGroupの削除
  - Actions → Deleteで消せる
- AMIの削除
  - アクション → AMIを登録解除 を選択し削除
- VPCの削除
  - ルートテーブルとセキュリティグループ,IGW,Subnetもまとめて削除できる
  - ※RDSとEC2が消えてないと削除できない
- キーペアの削除
  - 自分の鍵にチェックを入れて削除できる
  - 自分のPCに保存した秘密鍵も削除

セクションはすべて終了！
お疲れ様でした！！