



AWSハンズオン

神戸大学大学院システム情報学研究科

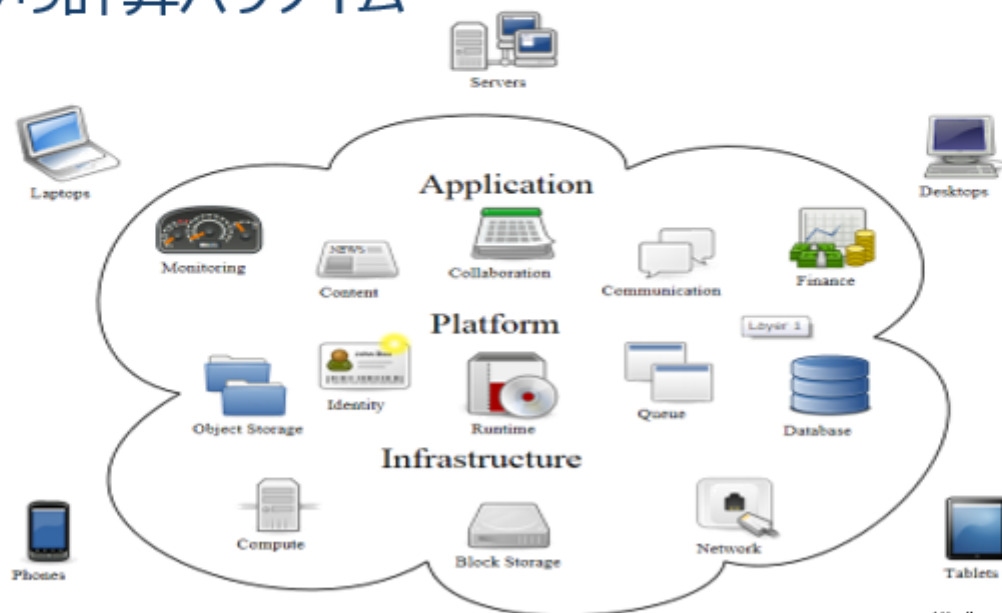
中村 匡秀



復習: クラウドコンピューティングとは

1.2 クラウドコンピューティング

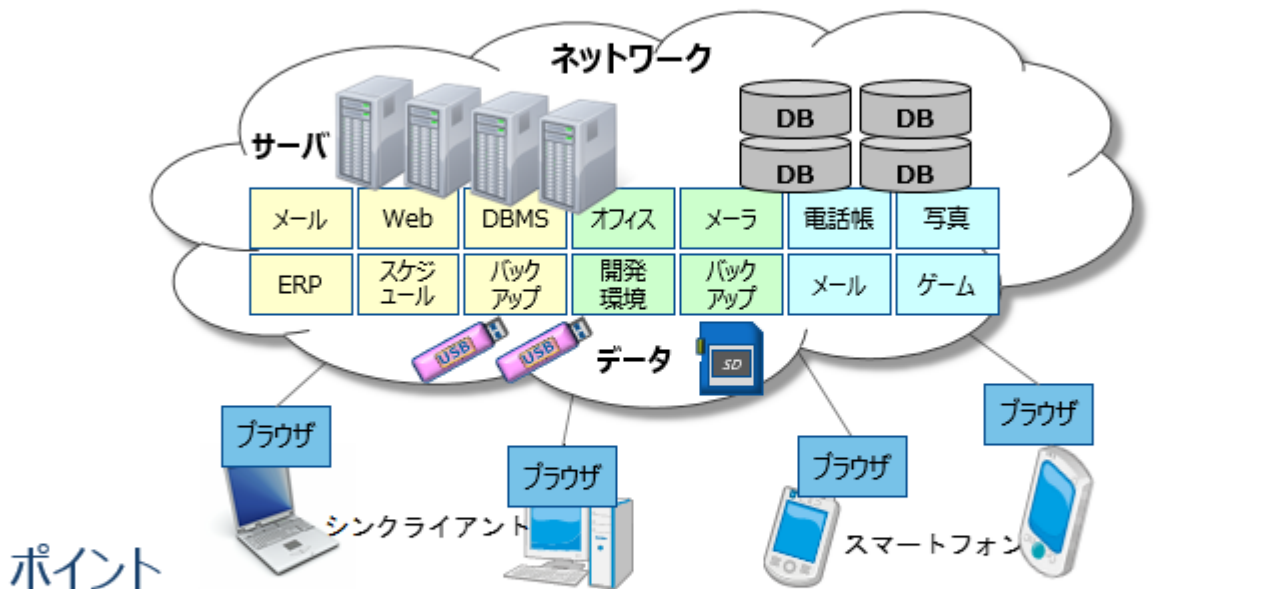
- ネットワークのどこかで管理され、必要に応じて利用する
- 様々な**計算資源**がネットワーク（のどこか）で管理され、ユーザは必要に応じてそれらを「**サービス**」として利用するという計算パラダイム



復習: クラウドコンピューティング環境

1.6 クラウドコンピューティング環境

■ ネットワークは多様な計算資源を収容する仮想的な巨大コンピュータ



ポイント

- ◆ ネットワークは多様な計算資源を収容する仮想的な巨大コンピュータ
- ◆ ハードウェア, ソフトウェア, データはクラウド内のサービスとして存在する
- ◆ 業務や目的に応じて, サービスを選択・利用. 不要になればやめる
- ◆ 手元のコンピュータは巨大コンピュータにアクセスするための軽量な窓口



復習：クラウドの定義

2.3 クラウドを性質づける3つの側面

■ 5つの本質的性質, 3つのサービスモデル, 4つの配備モデル

■ 5つの本質的性質 (Essential Characteristics)

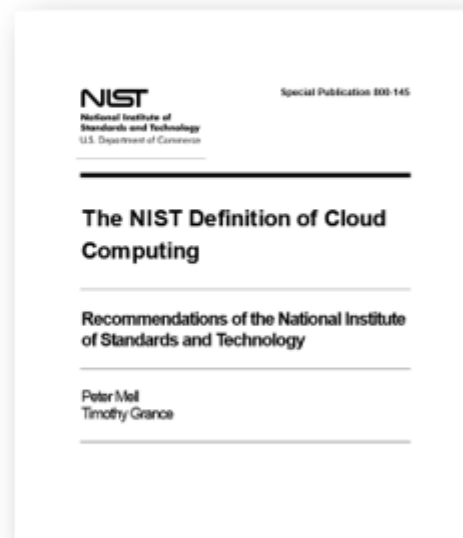
1. On-demand self-service
2. Broad network access
3. Resource pooling
4. Rapid elasticity
5. Measured service

■ 3つのサービスモデル (Service Models)

1. Software as a Service (SaaS)
2. Platform as a Service (PaaS)
3. Infrastructure as a Service (IaaS)

■ 4つの配備モデル (Deployment Models)

1. Public cloud
2. Private cloud
3. Community cloud
4. Hybrid cloud





Amazon Web Services (AWS)

■ 米Amazon社が提供するクラウドサービスプラットフォーム

- ◆ 基盤から応用まで多彩な資源をサービスとして提供
- ◆ Amazon社の収益の柱 (利益率25%, EC企業は4.5%)

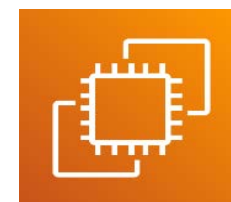


■ 複数のサービスを組み合わせてクラウドシステムを実装

- ◆ 2020年現在176個のサービスが存在
- ◆ 利用者は目的・用途に応じて、自分でサービスを取捨選択
- ◆ サービス実行は全てAPI経由で行われる

■ AWSで最も基本的で重要なサービスは**EC2**

- ◆ Amazon Elastic Computing Cloud
- ◆ 汎用仮想サーバを時間貸しするIaaS
 - 必要な時に必要な数だけサーバを調達
 - 様々なタイプのインスタンスをAMIから生成
 - 従量課金 (t2.microの場合, 1時間0.0116USD = 1.19円)



Amazon EC2

AWS誕生秘話*



Dan Rose
@DanRose999



■ Dan Rose氏 (@DanRose999) Twitter談 2021-01-09

- ◆ 2000年インターネットバブル崩壊. ネット企業への投資が途絶える
- ◆ AmazonのEC事業もピンチに. 信頼性を保ちつつコスト削減が必須
- ◆ 高性能サーバ (Sun)のライセンス料が高すぎる!
- ◆ SunのサーバをすべてLinuxに代えよう! (インフラコスト80%減)
- ◆ 移行作業は1年以上. 新規開発はストップ. あと数か月で倒産の危機
- ◆ ラッキーにも移行作業は時間通り成功. 無限のインフラを入手した!
- ◆ Jeff (ベゾス)はこう考えた

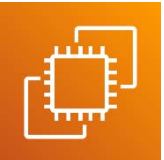
- 小売業として我々は毎年11月と12月だけ, トラフィックと売りが跳ね上がる. 裏を返せば, 年間46週分の**余ったサーバ能力**がある. これを**他社に貸し出したら**どうだろう?
- 開発スピードを加速するため, **他のチームからの許可なく, 別のチームがビルド**できるようにしよう
- ビジネスを始めるのになぜ自社で発電所(データセンター)を持たないといけないのだろう? 我々は**自分の生活をするのに自分で発電する**かい?


* <https://www.publickey1.jp/blog/21/awsamazonsunhplinux.html>





代表的なAWSのサービス・その1

計算装置


★  Amazon EC2


 Amazon Lightsail

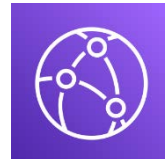
 AWS Lambda


 AWS Elastic Beanstalk

ネットワーク


★  Amazon VPC


 Amazon Route 53

 Amazon CloudFront

★  Elastic Load Balancing

記憶装置

★  Amazon Elastic Block Store (EBS)

 Amazon Simple Storage Service (S3)


 Amazon Aurora


データベース


 Amazon RDS

 Amazon DynamoDB

管理・統制

 Amazon CloudWatch

★  AWS Mgmt Console

 AWS Auto Scaling


サービス連携・統合


 Amazon Simple Notification Service

 Amazon Simple Queue Service

 AWS Step Functions

移行・マイグレーション

 AWS Server Migration Service

 AWS Database Migration Service



代表的なAWSのサービス・その2

認証・セキュリティ

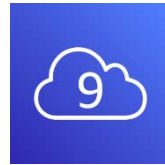


AWS Identity & Access Management



AWS Shield

開発ツール

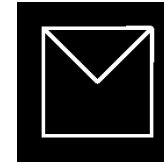


AWS Cloud9



AWS Command Line Interface

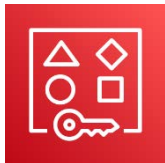
ビジネスツール



Amazon WorkMail



Alexa for Business



AWS Resource Access Manager

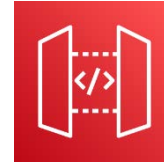


AWS Firewall Manager

モバイル



AWS Amplify



Amazon API Gateway

コンテナ



Amazon Elastic Container Registry



Amazon Elastic Container Kubernetes

モノのインターネット (IoT)



AWS IoT Core



AWS IoT Analytics



AWS IoT Device Management

ビッグデータ



Amazon EMR



AWS Lake Formation

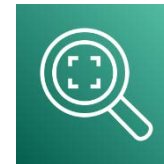
AI・機械学習



Amazon SageMaker



AWS Deep Learning



Amazon Rekognition



ハンズオン1: EC2でWEBサイトを構築する



第1話:「ポータルサイト」を構築せよ

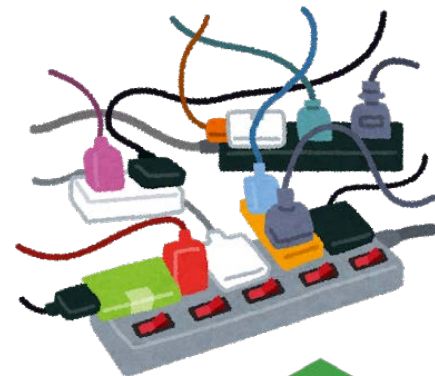
- 上司: 今度, 社内に「CPS研究センター」という新しい組織を立ち上げる話があるんだ. 社長の肝いりでね.
- 私: 初めて聞きました. 私の周りは, 誰も知らないですよ.
- 上司: そうなんだよ. それで, 早急に広報のためのポータルWebサイトが欲しいんだ. とりあえずコンテンツは外注したので, Webサーバの調達とセットアップ, コンテンツの設置と公開をお願いしていいかな?
- 私: 期限はいつまでですか? あと予算はいくらぐらいでしょう?
- 上司: ひと月も待てない. 2週間ぐらいでやってくれ. 予算はなるべく安く. 頼んだよ.





さあ大変だ！

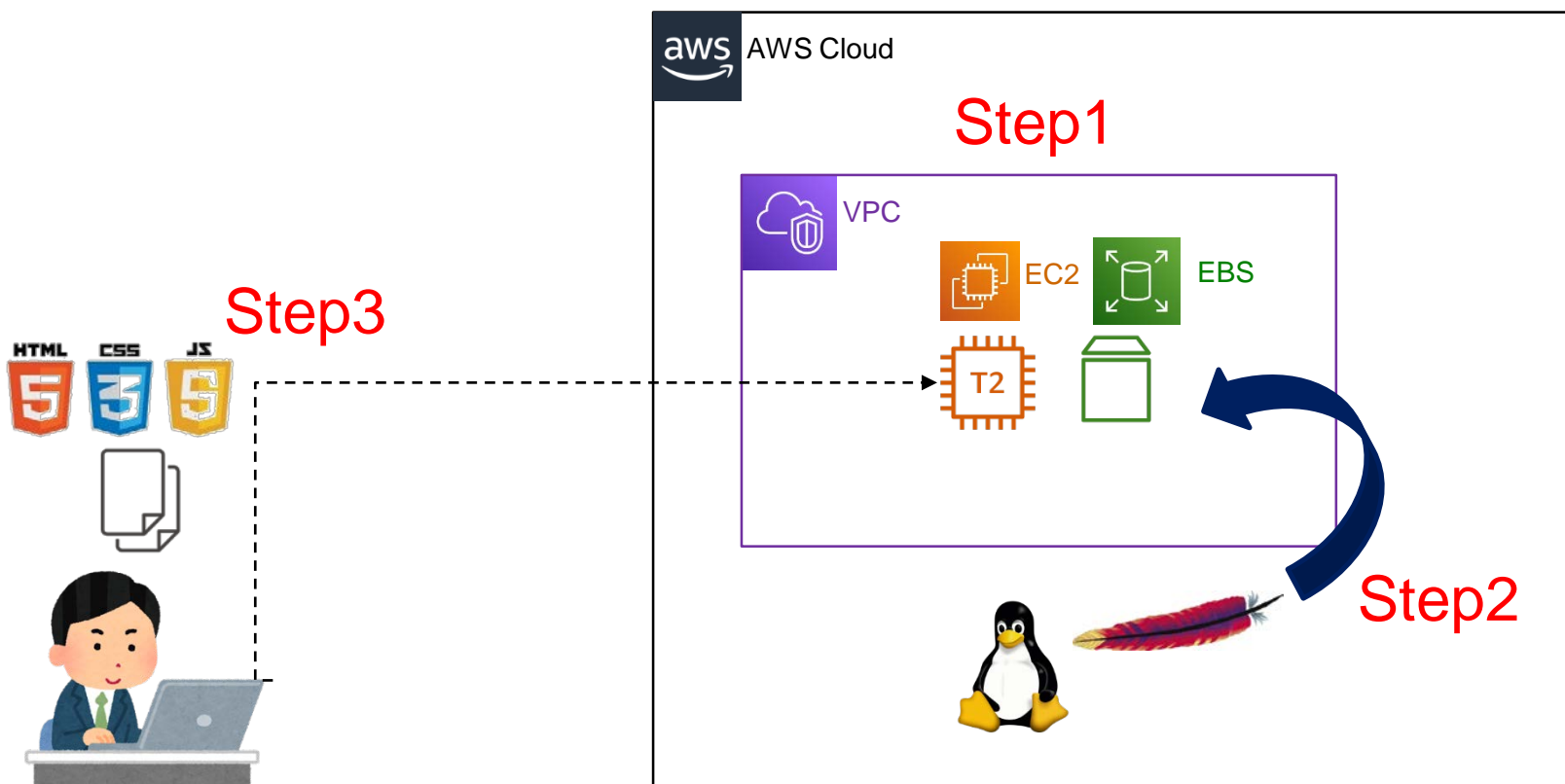
- どんなサーバを買う？
- どこに置く？
- どうやってつなぐ？
- OSは？サーバソフトは？
- 誰が管理する？





ハンズオン1: EC2でWebサイトを構築する

- Step1: AWSにEC2インスタンスを作成
- Step2: サーバに必要なソフトウェアをインストール
- Step3: HTMLコンテンツをデプロイ(配備)





Step1: EC2インスタンスの作成(1 / 10)

■ Webブラウザを開きAWS管理コンソールにアクセス

- ◆ AWS Educateを開いて, Email, パスワードでサインイン
- ◆ Class RoomのVocareumの画面からAWS Console
- ◆ 右上「バージニア北部」になっていることを確認



AWS Management Console

Workbench

labs.vocareum.com/main/main.php?m=editor&nav=1&asid=417729&stepid=417730

vocareum

Welcome to your AWS Educate Account

Your AWS Account Status

Active
full access (masa-n@cs.kobe-u.ac.jp)

\$49.91
remaining credits (estimated)

2:34
session time

Account Details

AWS Console

Please use AWS Educate Account responsibly. Remember to shut down your instances when not in use to make the best use of your credits. And, don't forget to logout once you are done with your work!

ALERT:
Due to recent changes within Amazon RDS, you won't be able to create a database by using the default options. You will need to go to Additional configuration and uncheck the Enable Enhanced monitoring selection under the Monitoring section.



AWS マネジメントコンソール

us-east-2.console.aws.amazon.com/console/home?region=us-east-2#

aws

サービス

サービス、機能、マーケットプレースの製品、 [Alt+S]

muretti0 14

オハイオ

サポート

AWS マネジメントコンソール **バージニア北部**

AWS のサービス

最近アクセスしたサービス

Step Functions Billing VPC

すべてのサービス

ソリューションの構築

シンプルなウィザードと自動化されたワークフローで作業を開始しましょう。

仮想マシンを起動する

EC2 を使用 2~3分

ウェブアプリを構築する

Elastic Beanstalk を使用 6分

仮想サーバーを使用する

Lightsail を使用 1~2分

外出先でも AWS サービスに常時接続

AWS コンソールモバイルアプリを iOS または Android モバイルデバイスにダウンロードします。 詳細はこちらから

AWS を試す

Amazon Redshift

データレイクにクエリを拡張できる、高速かつシンプルで、費用対効果の高いデータウェアハウス。 詳細はこちらから

AWS Fargate を使ってサーバーレスコンテナを実行

AWS Fargate が実行され、サーバーやクラスターを管理することなく、コンテナをスケーリングします。 詳細はこちらから

フィードバック 日本語

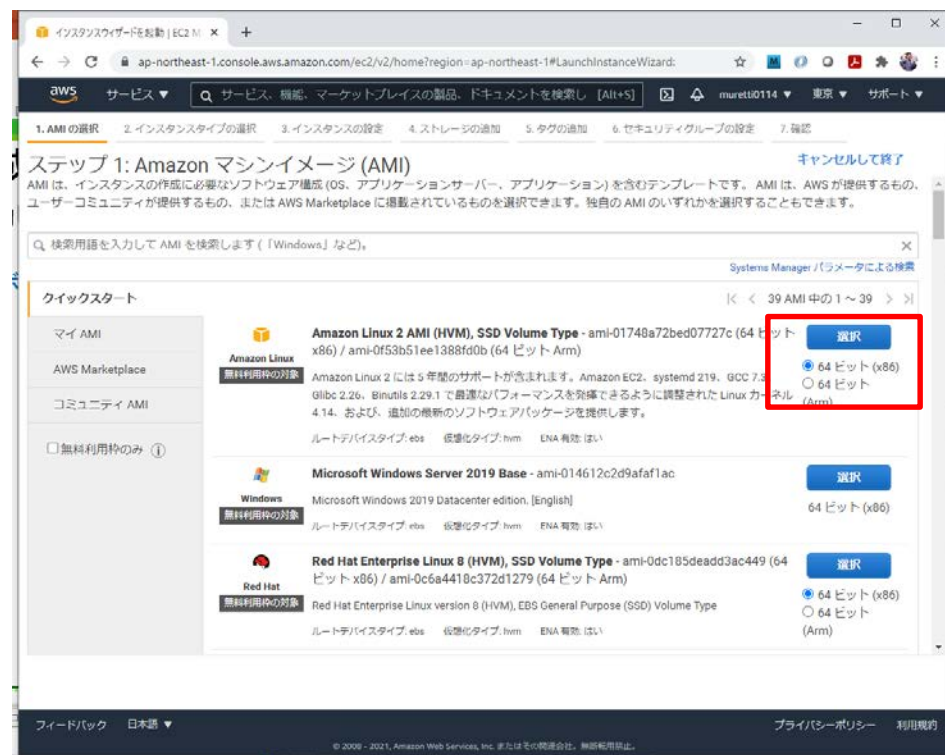
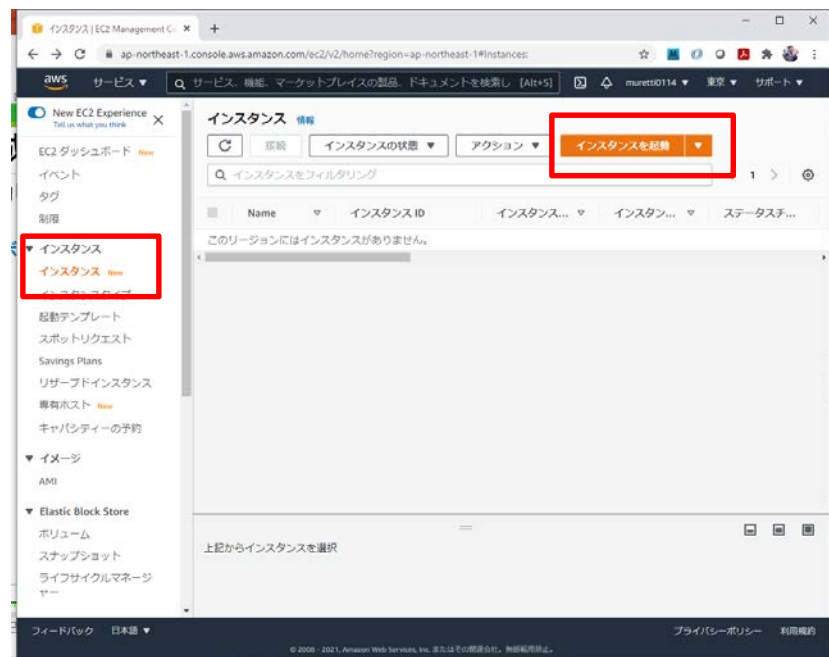
プライバシーポリシー 利用規約

© 2008 - 2021, Amazon Web Services, Inc. またはその関連会社。無断転載禁止。

Step1: EC2インスタンスの作成 (2 / 10)

■ EC2コンソールからインスタンスを起動

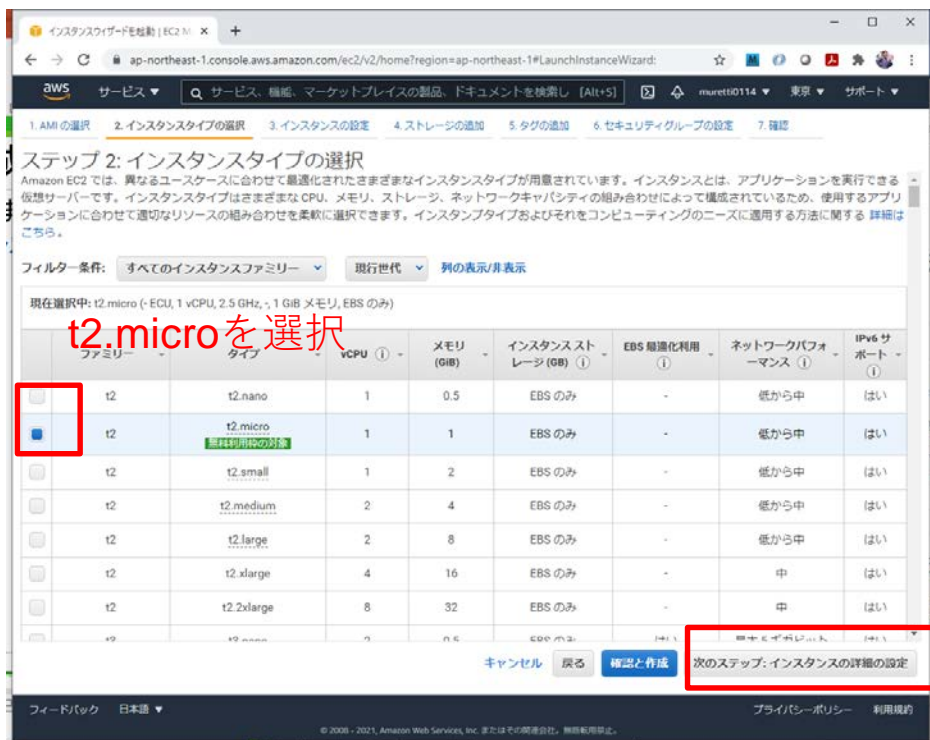
- ◆ 「すべてのサービス」→「EC2」をクリック
- ◆ メニューの「インスタンス」→右側オレンジボタン「インスタンスを起動」
- ◆ Amazon Linux 2 AMI (HVM), SSD Volume Type を選択



Step1: EC2インスタンスの作成 (3 / 10)

■ 作成するEC2のインスタンス型を決定

- ◆ t2.microを選択し「次のステップ: インスタンス詳細の設定」をクリック
- ◆ 次の画面では設定を触らないが, 「ネットワーク」の名前をメモしておく
- ◆ 「次のステップ: ストレージの追加」をクリック



ステップ 2: インスタンスタイプの選択

Amazon EC2 では、異なるユースケースに合わせて最適化されたさまざまなインスタンスタイプが用意されています。インスタンスとは、アプリケーションを実行できる仮想サーバーです。インスタンスタイプはさまざまな CPU、メモリ、ストレージ、ネットワークキャパシティの組み合わせによって構成されているため、使用するアプリケーションに合わせて適切なリソースの組み合わせを柔軟に選択できます。インスタンスタイプおよびそれをコンピューティングのコースに適用する方法に関する詳細はこちら。

フィルター条件: すべてのインスタンスファミリー | 現行世代 | 列の表示/非表示

現在選択中: t2.micro (ECU, 1 vCPU, 2.5 GHz, 1 GiB メモリ, EBS のみ)

ファミリー	タイプ	vCPU	メモリ (GiB)	インスタンスストレージ (GB)	EBS 最適化利用	ネットワークパフォーマンス	IPv6 サポート
<input type="checkbox"/>	t2.nano	1	0.5	EBS のみ	-	低から中	はい
<input checked="" type="checkbox"/>	t2.micro	1	1	EBS のみ	-	低から中	はい
<input type="checkbox"/>	t2.small	1	2	EBS のみ	-	低から中	はい
<input type="checkbox"/>	t2.medium	2	4	EBS のみ	-	低から中	はい
<input type="checkbox"/>	t2.large	2	8	EBS のみ	-	低から中	はい
<input type="checkbox"/>	t2.xlarge	4	16	EBS のみ	-	中	はい
<input type="checkbox"/>	t2.2xlarge	8	32	EBS のみ	-	中	はい

キャンセル 戻る 確認と作成 次ステップ: インスタンスの詳細の設定



ステップ 3: インスタンスの詳細の設定

実行に備えて 1 インスタンスを設定します。同じ AMI から複数の 1 インスタンスを作成し、より詳細な表現 9 のにのりメット インスタンスのソリューション、インスタンスへのアクセス管理ロール割り当てなどを行うことができます。

インスタンス数 ① 1 Auto-Scaling グループに作成する ①

購入のオプション ① スポットインスタンスの購入

ネットワーク ① vpc-5045 (デフォルト) 新しい VPC の作成

サブネット ① 既定のサブネット (サブネットの作成)

自動割り当てパブリック IP ① サブネット設定を使用 (有効) 新しいサブネットの作成

配置グループ ① インスタンスをプレシメントグループに追加します。

キャパシティの予約 ① 開く

ドメイン結合ディレクトリ ① ディレクトリなし 新しいディレクトリの作成

IAM ロール ① なし 新しい IAM ロールの作成

CPU オプション ① CPU オプションを指定

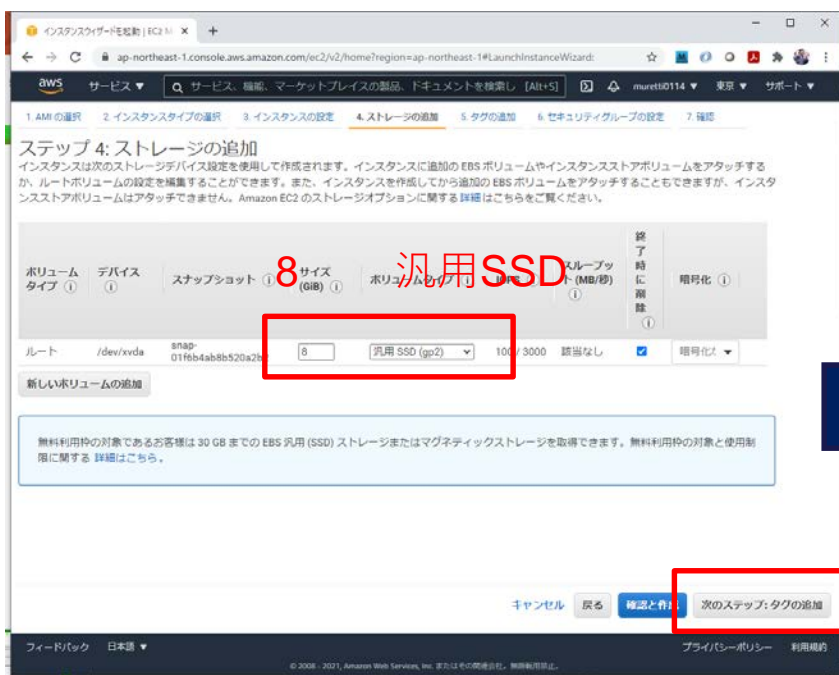
シャットダウン動作 ① 停止

停止 - 休止動作 ① 停止動作に休止動作を追加する

キャンセル 戻る 確認と作成 次ステップ: ストレージの追加

Step1: EC2インスタンスの作成(4 / 10)

- サーバにストレージ(SSD)を追加し名前を付けます
 - ◆ サイズを確認(8GB, 汎用SSD)して, 「次のステップ: タグの追加」
 - ◆ キーに「Name」, 値に「WebServer」と入力
 - ◆ 「次のステップ: セキュリティグループの設定」をクリック



ステップ 4: ストレージの追加

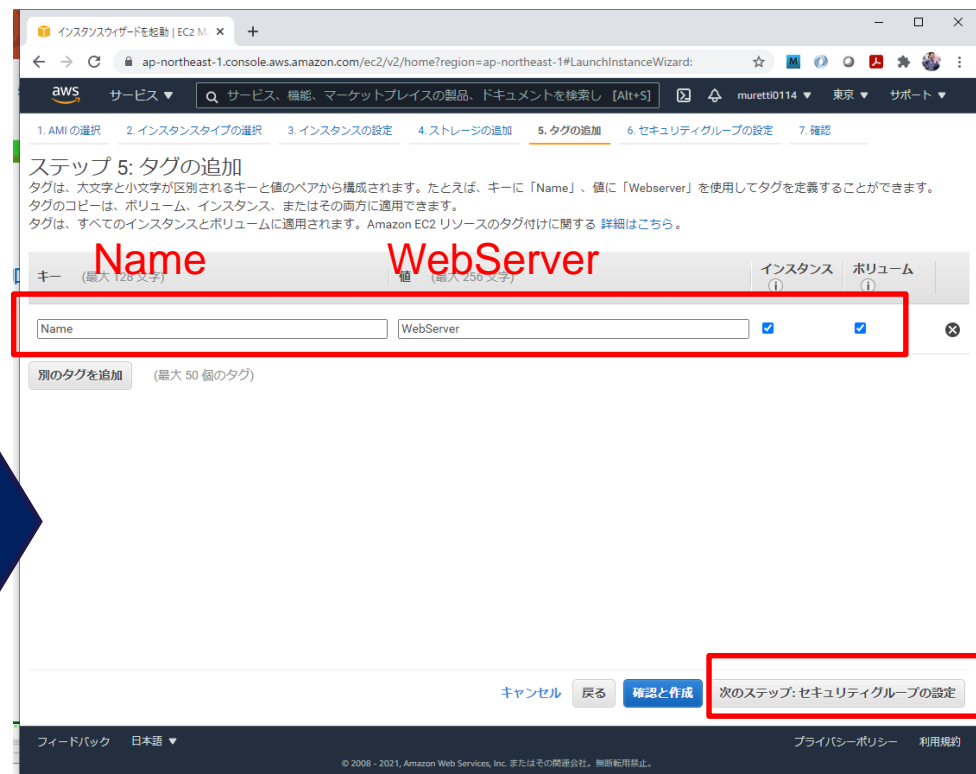
インスタンスは次のストレージデバイス設定を使用して作成されます。インスタンスに追加の EBS ボリュームやインスタンスストアボリュームをアタッチするか、ルートボリュームの設定を編集することができます。また、インスタンスを作成してから追加の EBS ボリュームをアタッチすることもできますが、インスタンスストアボリュームはアタッチできません。Amazon EC2 のストレージオプションに関する詳細はこちらをご覧ください。

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	スループット (MB/秒)	終了時に削除	暗号化
	/dev/xvda	snap-01f6b4ab8b520a2...	8	汎用 SSD (gp2)	100	3000	該当なし

新しいボリュームの追加

無料利用枠の対象であるお客様は 30 GB までの EBS 汎用 (SSD) ストレージまたはマグネティックストレージを取得できます。無料利用枠の対象と使用制限に関する詳細はこちら。

キャンセル 戻る **確認と作成** 次のステップ: タグの追加



ステップ 5: タグの追加

タグは、大文字と小文字が区別されるキーと値のペアから構成されます。たとえば、キーに「Name」、値に「Webserver」を使用してタグを定義することができます。タグのコピーは、ボリューム、インスタンス、またはその両方に適用できます。タグは、すべてのインスタンスとボリュームに適用されます。Amazon EC2 リソースのタグ付けに関する詳細はこちら。

キー (最大 128 文字)	値 (最大 256 文字)	インスタンス	ボリューム
Name	WebServer	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

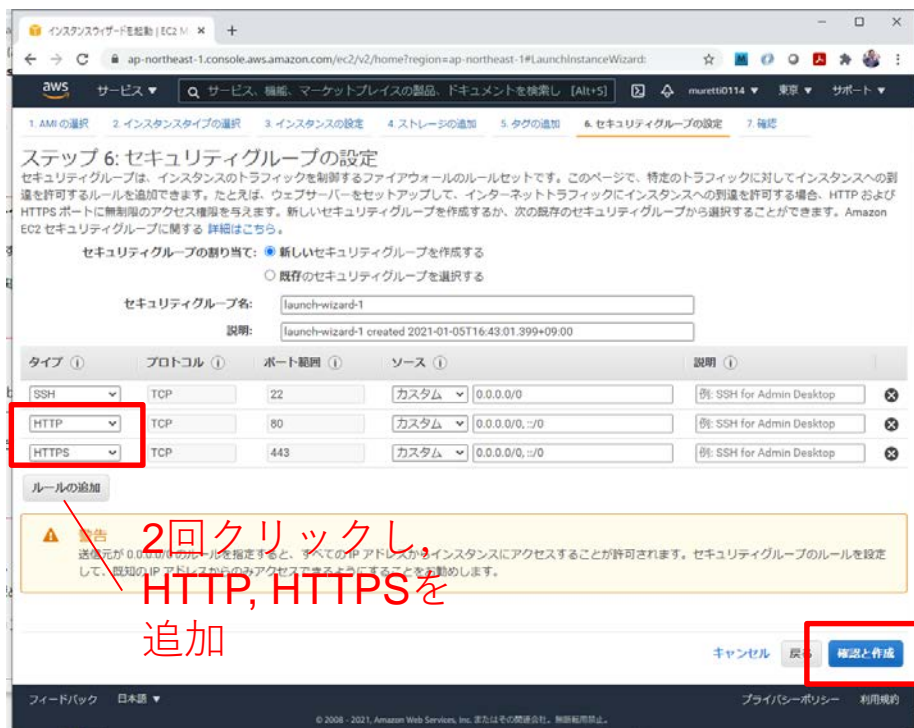
別のタグを追加 (最大 50 個のタグ)

キャンセル 戻る **確認と作成** 次のステップ: セキュリティグループの設定

Step1: EC2インスタンスの作成 (5 / 10)

■ サーバのセキュリティ(開放するポート)を設定する

- ◆ ルールの追加を2回押すと, SSHの下に2行新しい行が出現する
- ◆ 2行目にHTTP, 3行目にHTTPSを選択し, 「確認と作成」をクリック
 - WebサーバにするためHTTP(80),HTTPS(443)のポートを許可する
 - ソース(許可するアクセス元)は特に触らない
- ◆ 次の画面で諸元を確認. セキュリティ警告は無視して「起動」



ステップ 6: セキュリティグループの設定

セキュリティグループの割り当て: 新しいセキュリティグループを作成する

セキュリティグループ名:

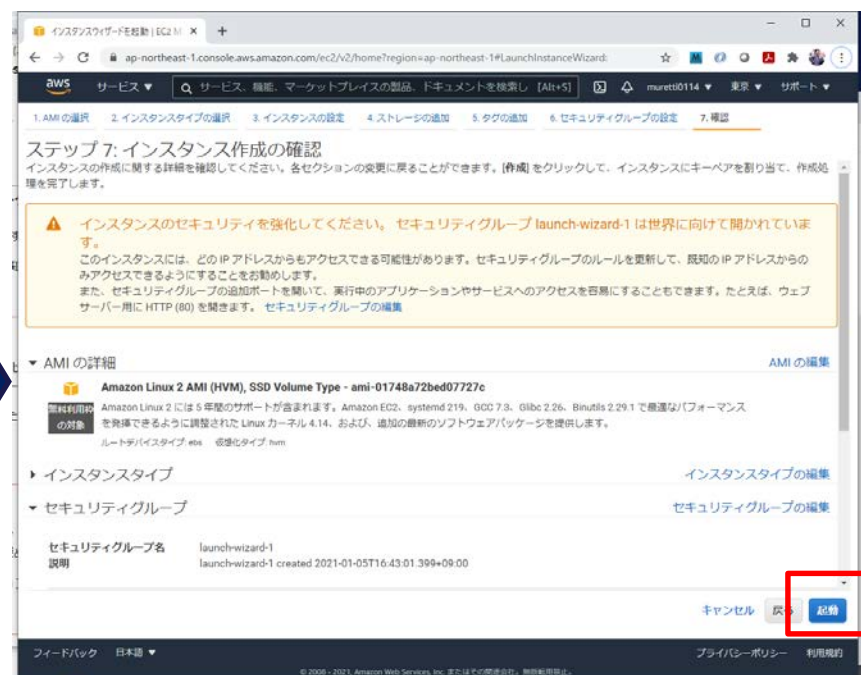
説明: launch-wizard-1 created 2021-01-05T16:43:01.399+09:00

タイプ	プロトコル	ポート範囲	ソース	説明
SSH	TCP	22	カスタム 0.0.0.0/0	例: SSH for Admin Desktop
HTTP	TCP	80	カスタム 0.0.0.0/0, ::0	例: SSH for Admin Desktop
HTTPS	TCP	443	カスタム 0.0.0.0/0, ::0	例: SSH for Admin Desktop

ルールの追加

警告: 送信元が 0.0.0.0/0 のルールを指定すると、すべての IP アドレスからインスタンスにアクセスすることが許可されます。セキュリティグループのルールを設定して、既知の IP アドレスからのみアクセスできるようにすることをお勧めします。

キャンセル 戻る **確認と作成**



ステップ 7: インスタンス作成の確認

インスタンスの作成に関する詳細を確認してください。各セクションの変更に戻ることができます。(作成) をクリックして、インスタンスにキーペアを割り当て、作成処理を完了します。

警告: インスタンスのセキュリティを強化してください。セキュリティグループ launch-wizard-1 は世界に向けて開かれています。このインスタンスには、どの IP アドレスからもアクセスできる可能性があります。セキュリティグループのルールを更新して、既知の IP アドレスからのみアクセスできるようにすることを勧めます。また、セキュリティグループの追加ポートを開いて、実行中のアプリケーションやサービスへのアクセスを容易にすることもできます。たとえば、ウェブサーバー用に HTTP (80) を開きます。セキュリティグループの編集

AMIの詳細 [AMIの編集](#)

Amazon Linux 2 AMI (HVM, SSD Volume Type - ami-01748a72bed07727c)

Amazon Linux 2 には 5 年間のサポートが含まれます。Amazon EC2、systemd 219、GCC 7.3、Glibc 2.26、Binutils 2.29.1 で最適なパフォーマンスを発揮できるように調整された Linux カーネル 4.14、および、追加の最新のソフトウェアパッケージを提供します。

ルートデバイスタイプ: ebs 仮想化タイプ: hvm

インスタンスタイプ [インスタンスタイプの編集](#)

セキュリティグループ [セキュリティグループの編集](#)

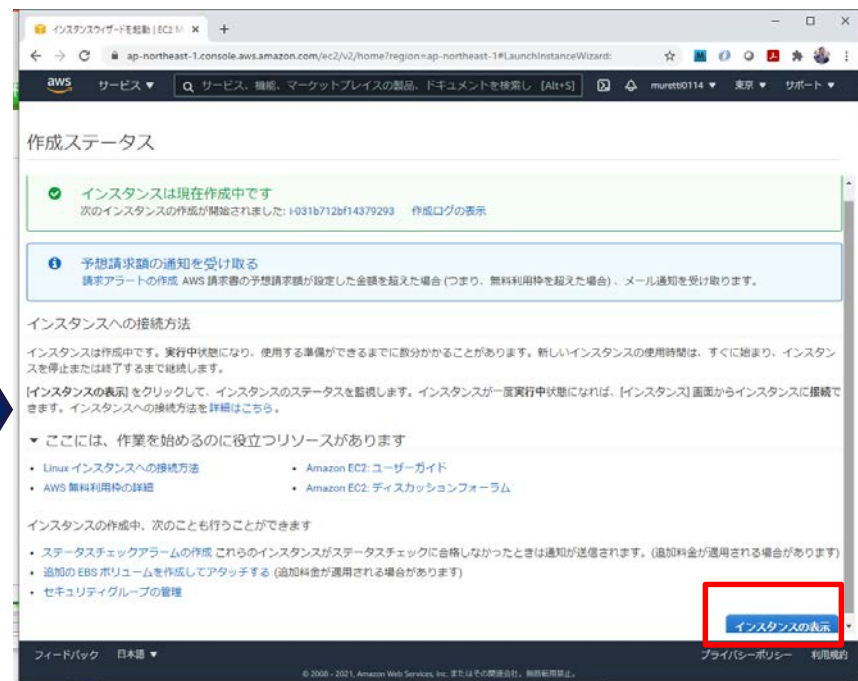
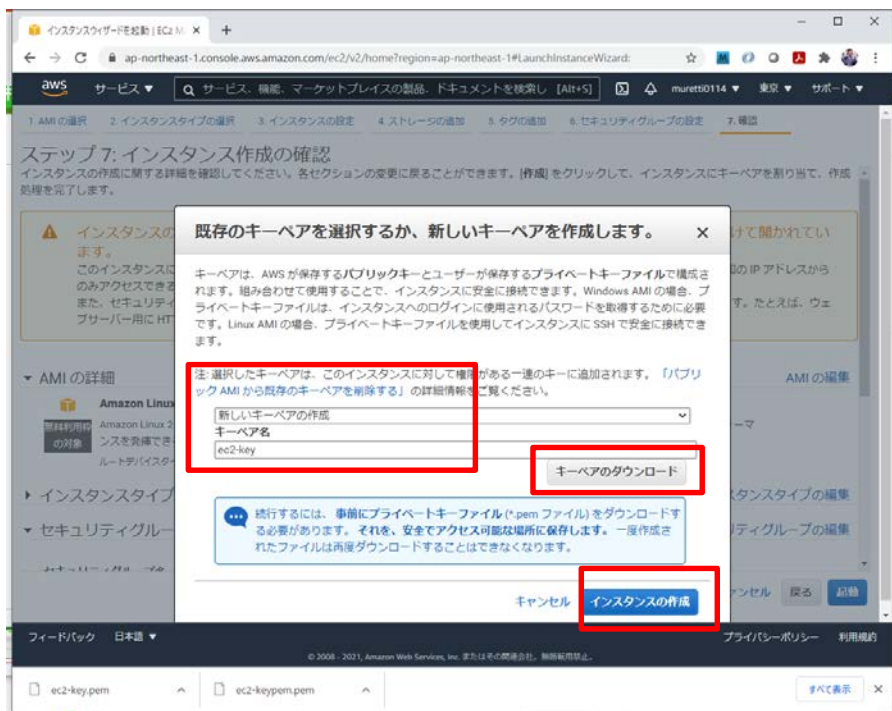
セキュリティグループ名: launch-wizard-1
説明: launch-wizard-1 created 2021-01-05T16:43:01.399+09:00

キャンセル 戻る **起動**

Step1: EC2インスタンスの作成(6 / 10)

■ サーバにログインするための暗号鍵を作成します

- ◆ 「新しいキーペアの作成」を選択し、キーペア名に任意の名前（例：ec2-key）を入力して、「キーペアのダウンロード」をクリック
- ◆ ec2-key.pemファイルがダウンロードされる
 - 二度と生成できないので絶対に無くさないこと、他人に公開しないこと
- ◆ 「インスタンスの作成」→作成ステータス→「インスタンスの表示」



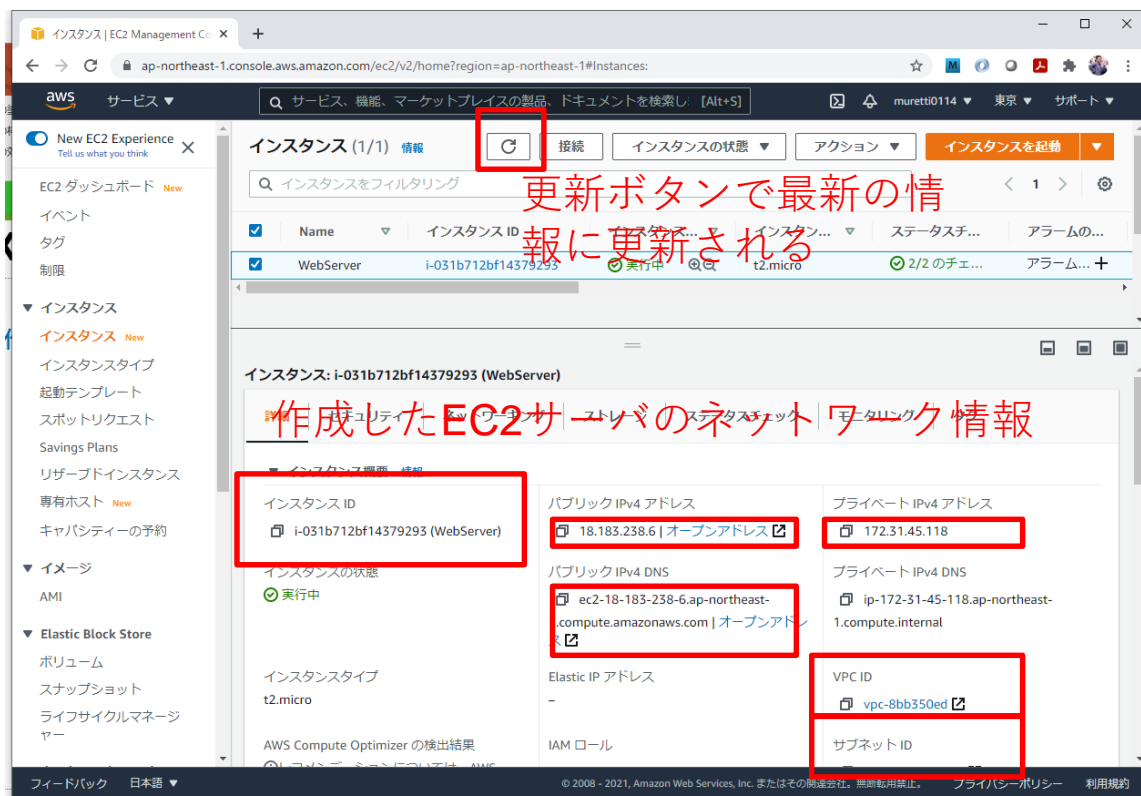
Step1: EC2インスタンスの作成(7/10)

■ 作成したインスタンスのネットワーク情報を確認する

◆ 更新ボタンを何回か押し、ステータスにチェックマークが作成完了

◆ リストからWebServerを選び「詳細」から情報を確認する

- インスタンスID, パブリックIPv4アドレス, プライベートIPv4アドレス, パブリックIPv4DNS, VPC ID, サブネットID



The screenshot shows the AWS Management Console interface for an EC2 instance named 'WebServer'. The instance ID is 'i-031b712bf14379293'. The status is '実行中' (Running). The network configuration is displayed in a table below the instance details.

Property	Value
インスタンス ID	i-031b712bf14379293 (WebServer)
パブリック IPv4 アドレス	18.183.238.6 オープンアドレス
プライベート IPv4 アドレス	172.31.45.118
パブリック IPv4 DNS	ec2-18-183-238-6.ap-northeast-1.compute.amazonaws.com オープンアドレス
プライベート IPv4 DNS	ip-172-31-45-118.ap-northeast-1.compute.internal
Elastic IP アドレス	-
VPC ID	vpc-8bb350ed
サブネット ID	-

Red boxes in the image highlight the '更新' (Refresh) button, the instance ID, the public IPv4 address, the public IPv4 DNS, the VPC ID, and the subnet ID.

更新ボタンで最新の情報に更新される

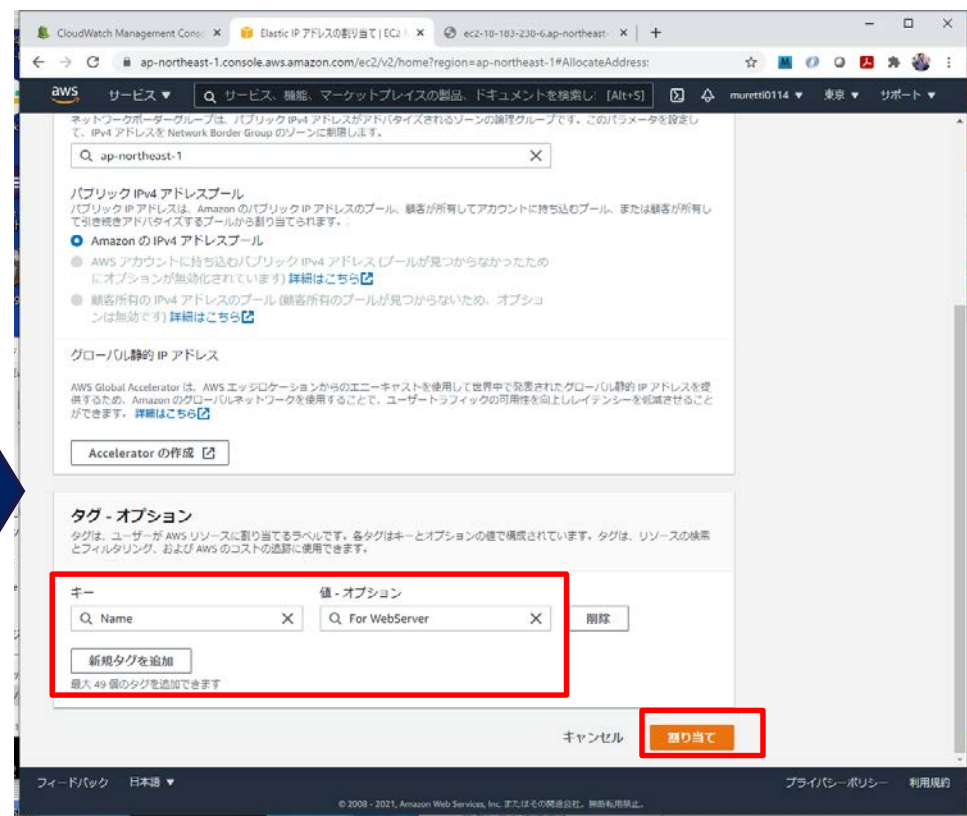
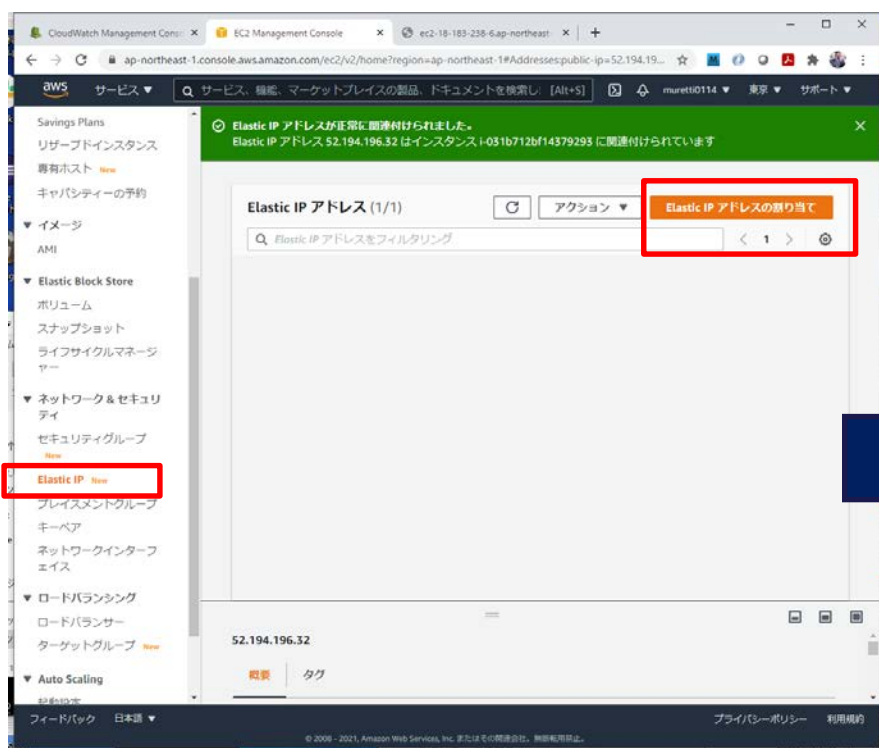
作成したEC2サーバのネットワーク情報

※ IPv4アドレスはインスタンスを再起動する度に変わってしまう

→ Webサーバのアドレスが固定的でなくなるので困る！！

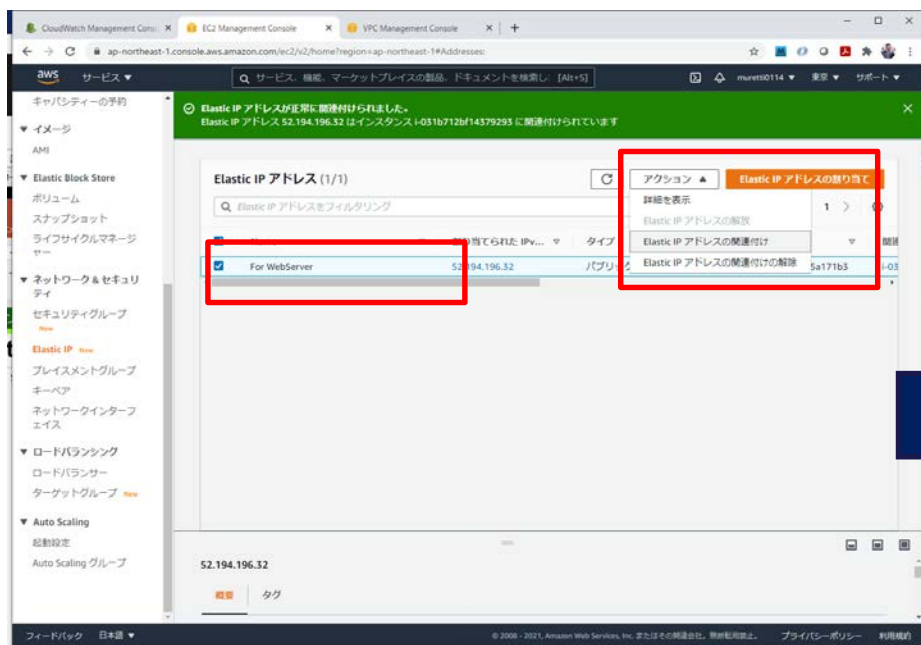
Step1: EC2インスタンスの作成(8 / 10)

- Elastic IPで固定的なパブリックIPv4アドレスを取得する
 - ◆ 左側のメニュー「ネットワーク & セキュリティ」→ Elastic IP
 - ◆ 橙のボタン「Elastic IPの割り当て」をクリック
 - ◆ 最下部: キー「Name」値「For WebServer」→ 「新規タグを追加」→ 「割り当て」



Step1: EC2インスタンスの作成(9/10)

- 割り当てたElastic IPを作成したEC2インスタンスに関連付ける
 - ◆ Elastic IPアドレスのリストから「For WebServer」を選択
 - ◆ 「アクション」→「Elastic IPアドレスの関連付け」
 - ◆ インスタンスからEC2のインスタンスを選択し「関連付ける」

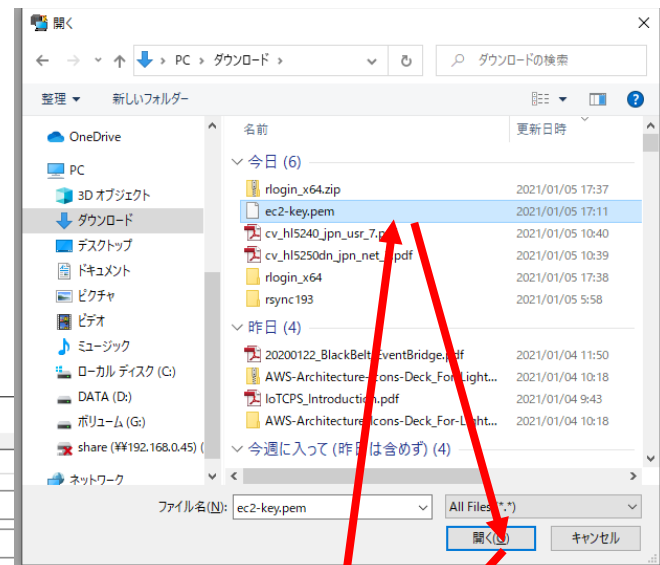
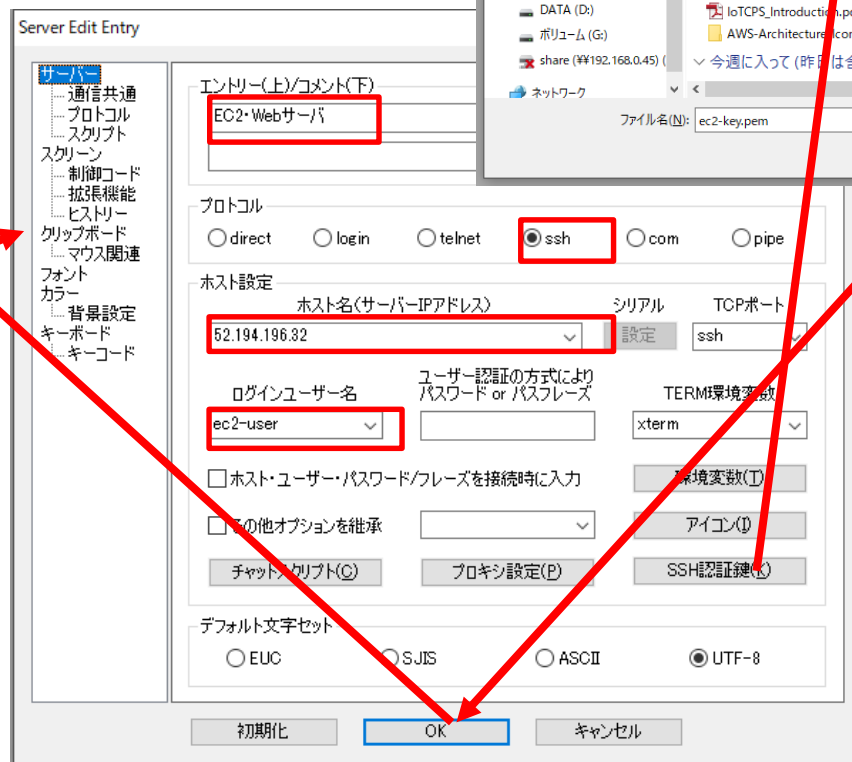
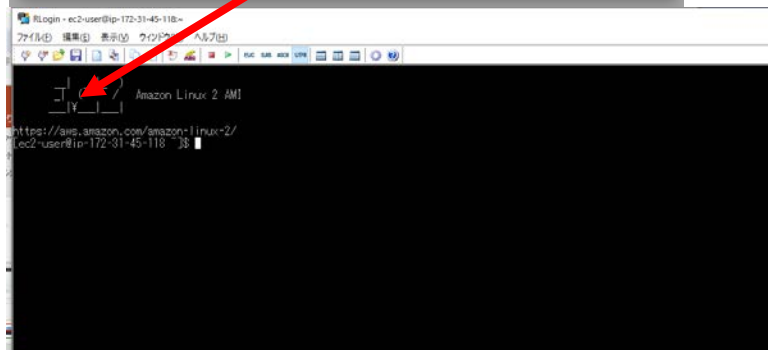
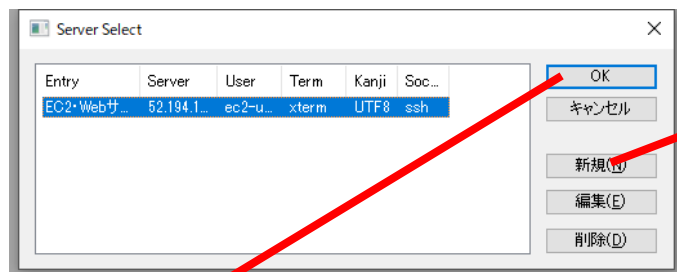


- ◆ 再びインスタンスの詳細画面(7/10参照)に戻ってネットワーク情報をチェック
 - 新たに割り当てられた**パブリックIPv4アドレス**, **パブリックIPv4DNS**をメモしておく

Step1: EC2インスタンスの作成(10 / 10)

■ RLoginで作成したEC2インスタンスにログインする

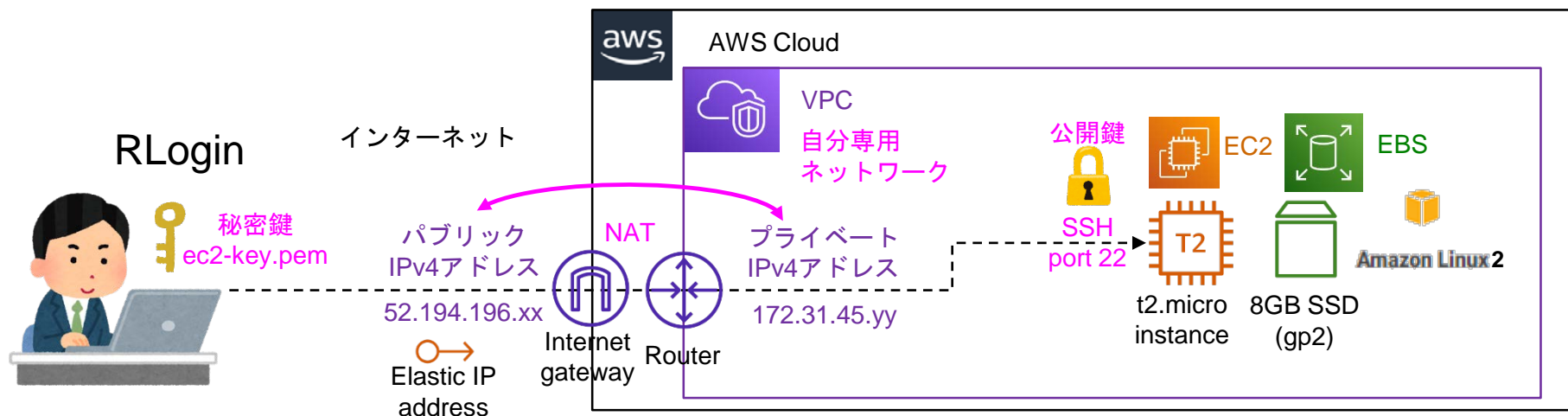
- ◆ 手元のWindowsマシンからRLoginを起動します
- ◆ 「新規」をクリックし下記入力する
 - エントリー(上) : **EC2・Webサーバ**, プロトコル: **SSH**
 - ホスト名: (**作成したEC2のパブリックIPv4アドレス, または, DNS**)
 - ログインユーザ名: **ec2-user** パスワード: (**なし**)
 - SSH認証鍵: 作成した**ec2-key.pem**ファイルを指定
- ◆ EC2・Webサーバを選択し「OK」メッセージが出るが続行
- ◆ プロンプトが出たら成功





Step1でやったことの解説

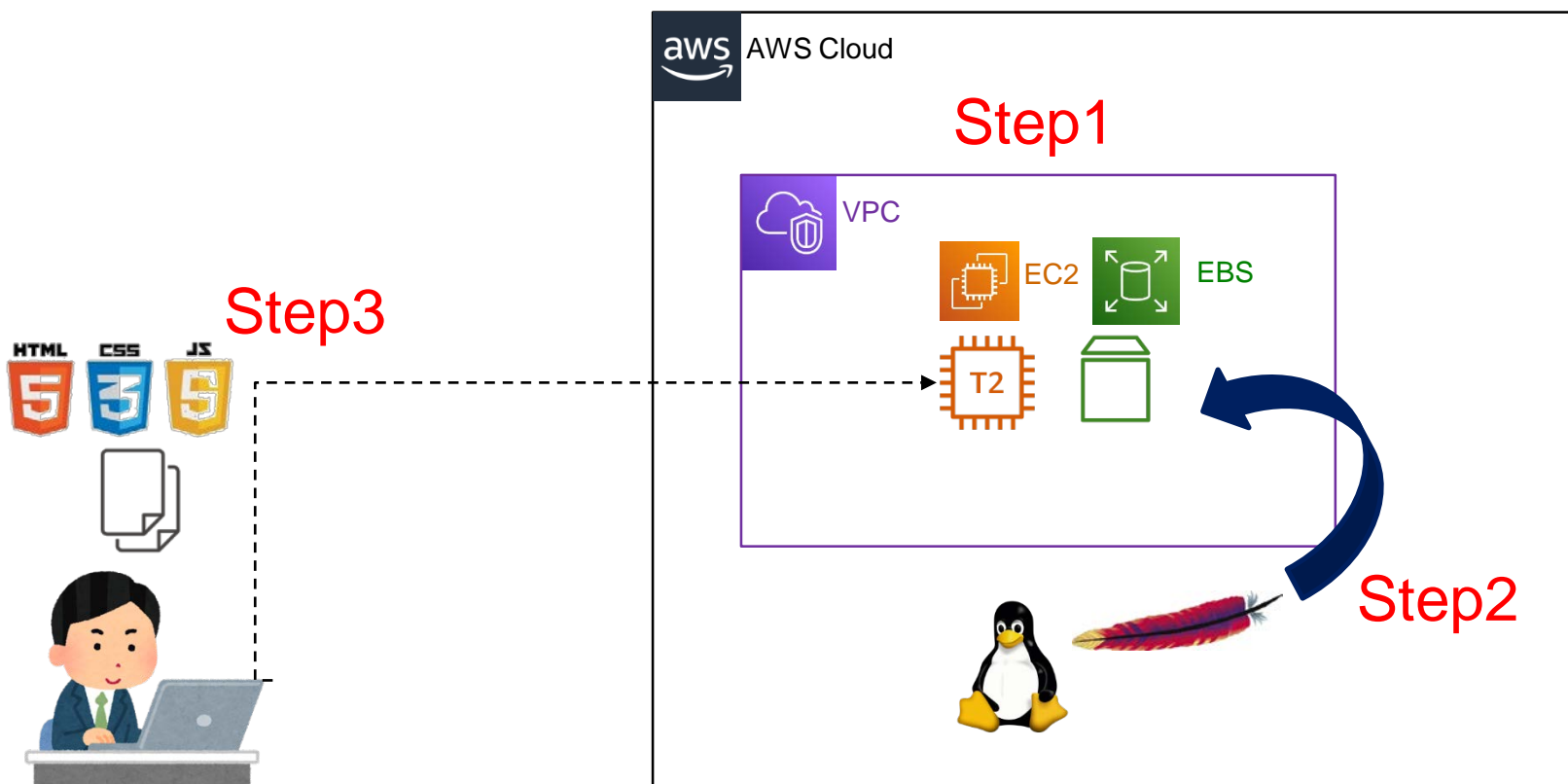
- EC2で仮想サーバ(Linux 2, SSD 8GB)を1台作成し, RLoginでログインした
 - ◆ EC2では様々なOSの仮想サーバを作成可能. **AMI**と呼ばれるテンプレートで定義される
 - AMI: Amazon Machine Image. 今回は, Amazon Linux2 を選択した
 - ◆ サーバの性能はインスタンスタイプで選択
 - t2.microは小規模ながら標準的なアプリケーション要件を満たすコストパフォーマンス型サーバ
 - ◆ 作成した仮想サーバの実体は**インスタンス**と呼ばれ, サーバを終了すると消えてなくなる
 - 同じAMI (型, テンプレ) から複数のインスタンスを作成できる. またインスタンスからカスタムAMIも作成可.
 - ◆ 最初のEC2インスタンスを作成する際, **自分専用のVPC**が**自動で作られ**その中に**配備される**
 - Virtual Private Cloud: 利用者個人のローカルネットワークを構築し, セキュリティを担保
 - ◆ サーバのIPアドレスを固定するため, **Elastic IP**を割り当てて**インスタンスに関連付け**
 - インターネットとVPCの通信は, ゲートウェイのNAT機能によってアドレス変換される
 - ◆ サーバの操作は**RLogin**の**SSH**プロトコルでログインして行う. 認証は作成したキーペアを使う
 - デフォルトのユーザ名はec2-user, パスワードはなし.





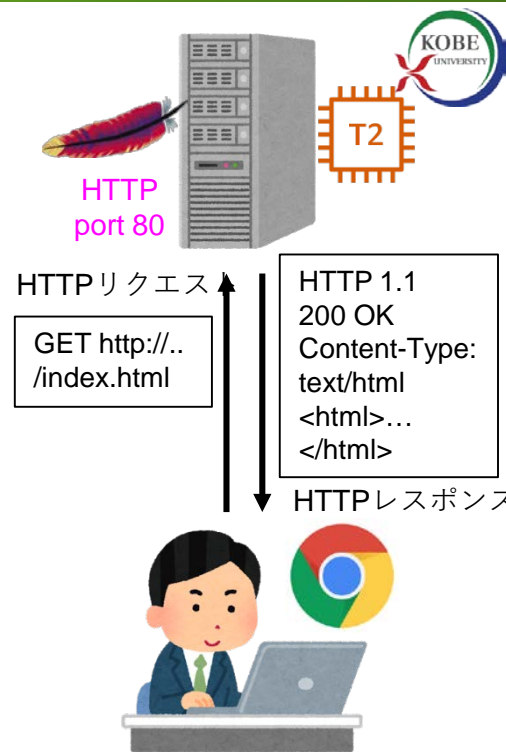
ハンズオン1: EC2でWebサイトを構築する

- Step1: AWSの東京リージョンにEC2インスタンスを作成
- Step2: サーバに必要なソフトウェアをインストール
- Step3: HTMLコンテンツをデプロイ(配備)



Webサーバ(ソフトウェア)の仕組み

- クライアント(Webブラウザ等)からリクエストがあると、要求されたリソース(コンテンツ)をデータストリームにして返す
 - ◆ HTTPリクエスト: クライアントからサーバへのデータの送信要求
 - 要求メソッド(GET, POST, PUT, DELETE等)と要求リソースのアドレス(URL)等を指定
 - ◆ HTTPレスポンス: Webサーバからクライアントへのデータ送信
 - ステータスコード, コンテンツタイプ, データ本体等を返す
- Webコンテンツ: HTML, CSS, JS形式で書かれたテキストファイル, および, その他のメディアファイル(書類, 画像, 動画, 音声など)で構成される
 - ◆ HTML (Hyper Text Markup Language)
 - Webページの文書構造と内容を定義する
 - ◆ CSS (Cascade Style Sheet)
 - Webページのレイアウトやスタイルを定義する
 - ◆ JS (JavaScript)
 - Webページの動的な振る舞いを定義する
 - ◆ Webサーバの指定されたディレクトリに**配備(デプロイ)**する
 - Apache HTTPDの場合, /var/www/html 配下にコピーする



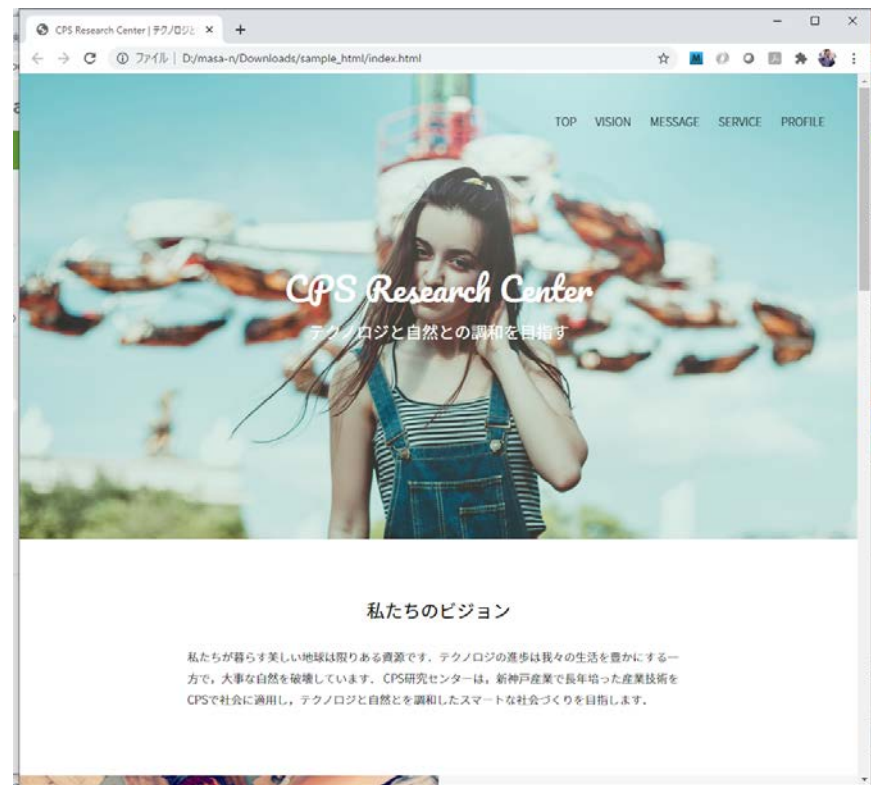
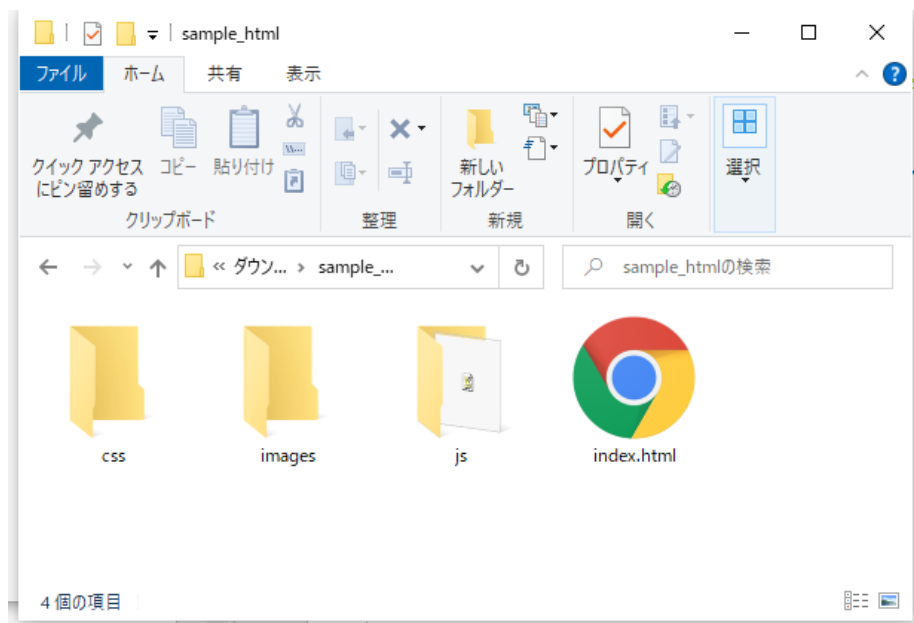


Webサイトのコンテンツを見てみよう

■ 「CPS研究センター」のWebコンテンツ（中村作のサンプル）

◆ Wikiからsample_html.zipをダウンロードし, zipを解凍して中身を見てみよう

- ブラウザでindex.htmlを開く
- 各ファイルをテキストエディタで開く





Step2: サーバにソフトをインストール (1/2)

■ Apache HTTPDサーバをインストールする

- ◆ 作成したEC2インスタンスの上に、サーバソフトウェア Apache HTTPDをインストールする
- ◆ RLoginでサーバにログインした後、下記のコマンドを1つずつ実行する
 - 先頭の\$や#は入力なくてよい(プロンプト \$:ユーザ, #:管理者)

- | | |
|-----------------|---------------------------------------|
| 1. 管理者権限を取得 | <code>\$ sudo su</code> |
| 2. 更新リストを最新に | <code># yum update -y</code> |
| 3. HTTPDをインストール | <code># yum install -y httpd</code> |
| 4. HTTPDを起動 | <code># systemctl start httpd</code> |
| 5. 再起動時にも有効化 | <code># systemctl enable httpd</code> |
| 6. 状態を確認 | <code># systemctl status httpd</code> |

■ Linux コマンド解説 (RHEL7/CentOS7系)

- ・ **sudo**: 管理者(root) 権限でコマンドを実行する
- ・ **su**: ユーザの切り替え、引数を省略するとrootになる
- ・ **yum**: ソフトウェアのインストールやアップデートを行う
- ・ **systemctl**: システムサービス (常駐プログラム) を管理する

```

RLogin - root@ip-172-31-45-118/home/ec2-user
ファイル(F) 編集(E) 表示(V) ウィンドウ(W) ヘルプ(H)
EUC SAS ADOJ UFRS
Installed:
  httpd.x86_64 0:2.4.46-1.amzn2
Dependency Installed:
  apr.x86_64 0:1.6.3-5.amzn2.0.2 apr-util.x86_64 0:1.6.1-5.amzn2.0.2 apr-util-bdb.x86_64 0:1.6.1-5.amzn2.0.2
  httpd-tools.x86_64 0:2.4.46-1.amzn2 mailcap.noarch 0:2.1.41-2.amzn2
Complete!
[root@ip-172-31-45-118 ec2-user]# systemctl start httpd
[root@ip-172-31-45-118 ec2-user]# systemctl enable httpd
Created symlink from /etc/systemd/system/multi-user.target.wants/httpd.service to /usr/lib/systemd/system/httpd.service
[root@ip-172-31-45-118 ec2-user]# ^C
[root@ip-172-31-45-118 ec2-user]# systemctl status httpd
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; vendor preset: disabled)
   Active: active (running) since Tue 2021-01-05 15:26:15 UTC; 1min 1s ago
     Docs: man:httpd.service(8)
  Main PID: 2463 (httpd)
   Status: "Total requests: 0; Idle/Busy workers 100/0; Requests/sec: 0; Bytes served/sec: 0 B/sec"
    CGroup: /system.slice/httpd.service
            └─2463 /usr/sbin/httpd -DFOREGROUND
              └─2464 /usr/sbin/httpd -DFOREGROUND
                └─2465 /usr/sbin/httpd -DFOREGROUND
                  └─2466 /usr/sbin/httpd -DFOREGROUND
                    └─2467 /usr/sbin/httpd -DFOREGROUND
                      └─2468 /usr/sbin/httpd -DFOREGROUND

Jan 05 15:26:15 ip-172-31-45-118.ap-northeast-1.compute.internal systemd[1]: Starting The Apache HTTP Server...
Jan 05 15:26:15 ip-172-31-45-118.ap-northeast-1.compute.internal systemd[1]: Started The Apache HTTP Server.
[root@ip-172-31-45-118 ec2-user]#
  
```

Step2: サーバにソフトをインストール (2/2)

■ Webサーバの起動を確認する

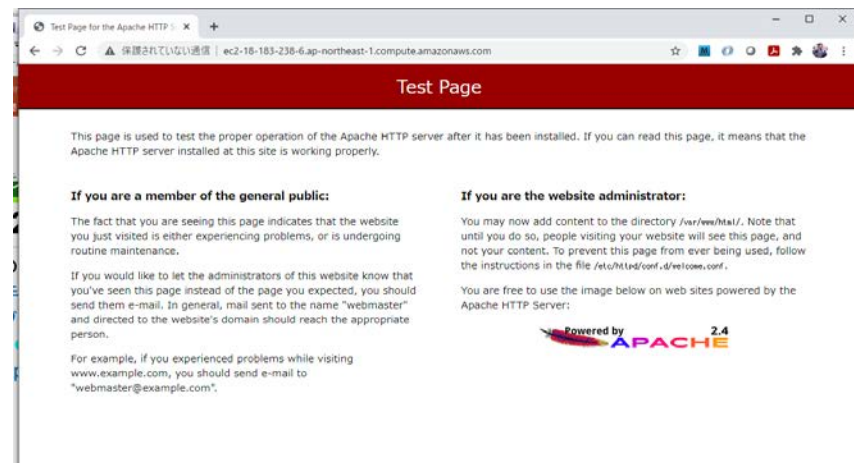
◆ 手元のPCでWebブラウザを開き, URLにパブリックIPv4DNS を入力

- `http://ec2-xx-yy-zz-x.ap-northeast-1.compute.amazonaws.com`
- パブリックIPv4アドレスでもよい

◆ Test Pageが表示されれば成功

◆ 表示されない場合

- `systemctl` でステータスを確認, または,
- EC2のセキュリティグループでHTTP (Port 80)が開いているかチェック



Webブラウザ

`http://ec2-xx-yy-zz-x.ap-northeast-1.compute.amazonaws.com`



パブリック
IPv4アドレス

52.194.196.xx

Elastic IP
address



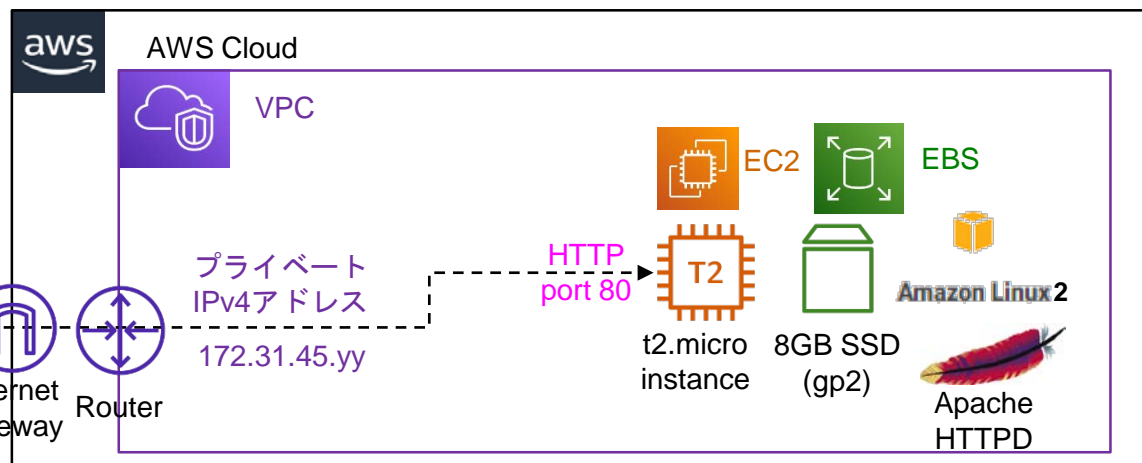
Internet
gateway



Router

プライベート
IPv4アドレス

172.31.45.yy

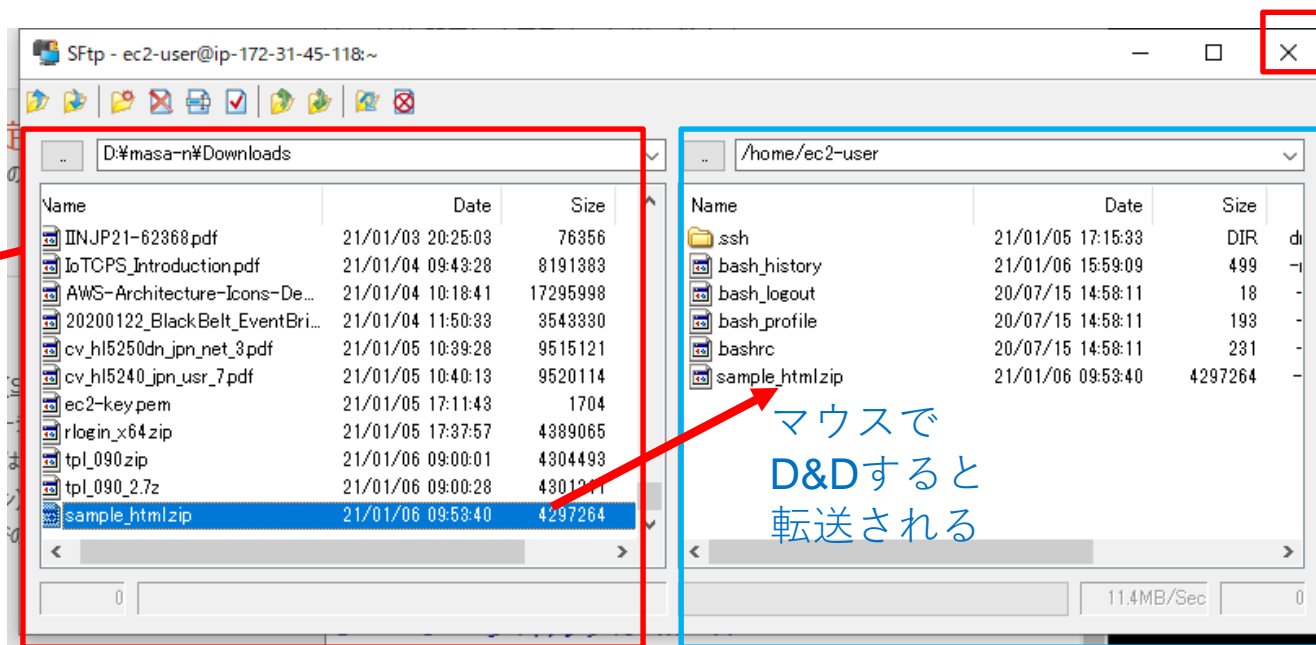


Step3: Webコンテンツのデプロイ (1/3)

■ 手元のWebコンテンツをサーバへ転送する

- ◆ RLoginのファイル→SFTPファイルの転送
- ◆ 左側のウィンドウでコンテンツをダウンロードしたフォルダを開く
 - 通常は, C:¥Users¥(ユーザ名)¥Downloads など
- ◆ sample_html.zip を右側のウィンドウへマウスでドラッグ & ドロップ
- ◆ ファイルがアップロードされたことを確認して閉じる

ローカルPC
(Windows)
のファイル



リモート
サーバ
(EC2) の
ファイル

マウスで
D&Dすると
転送される



Step3: Webコンテンツのデプロイ (2/3)

■ Webコンテンツの配備先の所有者と権限を変更する

◆ RLoginでサーバにログインした後, 下記のコマンドを1つずつ実行する

- コンテンツ配備先のディレクトリ (/var/www) がrootでしか書き込みできないので, ec2-userおよびApache HTTPDが書き込み出来るように変更する

- | | |
|---------------------------|--|
| 1. 管理者権限を取得 | <code>\$ sudo su</code> |
| 2. ec2-userをapacheグループに追加 | <code># usermod -a -G apache ec2-user</code> |
| 3. RLoginをいったん切断し再接続 | RLogin : ファイル→接続を閉じる
RLogin : ファイル→サーバに接続する |
| 4. 再び管理者権限を取得 | <code>\$ sudo su</code> |
| 5. /var/wwwに移動 | <code># cd /var/www</code> |
| 6. 所有者をapacheに変更 | <code># chown apache:apache *</code> |
| 7. グループ書込を許可 | <code># chmod 775 *</code> |
| 8. 通常ユーザに戻る | <code># exit</code> |

■ Linux コマンド解説 (RHEL7/CentOS7系)

- `usermod`: ユーザの属性を変更する
- `cd`: ディレクトリを移動する
- `chown`: ファイルやディレクトリの所有者を変更する
- `chmod`: ファイルやディレクトリの読み書き権限を変更する
- `exit`: シェルのセッションを終了する

```

RLogin - ec2-user@ip-172-31-45-118:~
ファイル(F) 編集(E) 表示(V) ウィンドウ(W) ヘルプ(H)
EUC SJS ASCII UTF8
Last login: Wed Jan  6 07:05:15 2021 from 133.30.159.204
┌───┴───┐
┌───┴───┐ Amazon Linux 2 AMI
└───┬───┘
└───┬───┘
https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-172-31-45-118 ~]$ sudo su
[root@ip-172-31-45-118 ec2-user]# cd /var/www
[root@ip-172-31-45-118 www]# chown apache:apache *
[root@ip-172-31-45-118 www]# chmod 775 *
[root@ip-172-31-45-118 www]# exit
exit
[ec2-user@ip-172-31-45-118 ~]$
  
```

Step3: Webコンテンツのデプロイ (3/3)

- 転送したコンテンツを解凍し、指定ディレクトリへコピーする
 - ◆ 下記のコマンドを1つずつ実行する

1. ホームディレクトリに戻る

```
$ cd
```
2. コンテンツがある確認

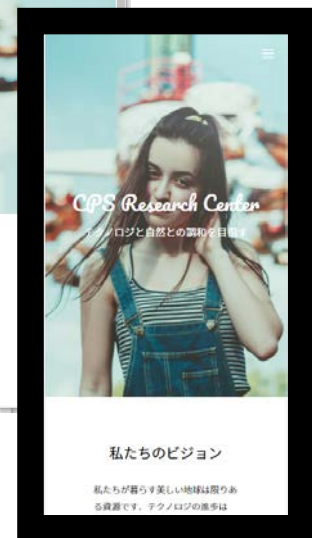
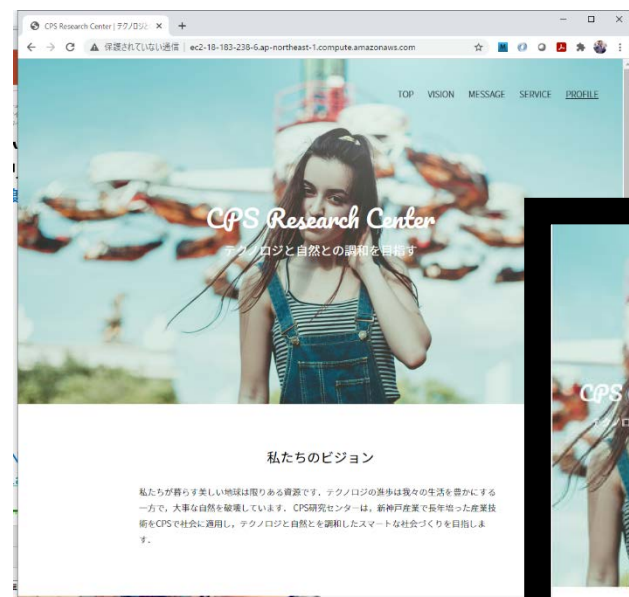
```
$ ls -l
```
3. zipを解凍する

```
$ unzip sample_html.zip
```
4. 回答された全ファイルを /var/www にコピー

```
$ cp -r sample_html/* /var/www/html/
```
5. ローカルPCでブラウザを開き、パブリックIPv4 DNSにアクセス

■ Linux コマンド解説 (RHEL7/CentOS7系)

- ・ ls: 現在のディレクトリの中身をリストする
- ・ unzip: zipファイルを解凍する
- ・ cp: ファイルをコピーする



- Webブラウザで確認する

- ◆ ローカルPCでブラウザを開き、パブリックIPv4 DNSにアクセス

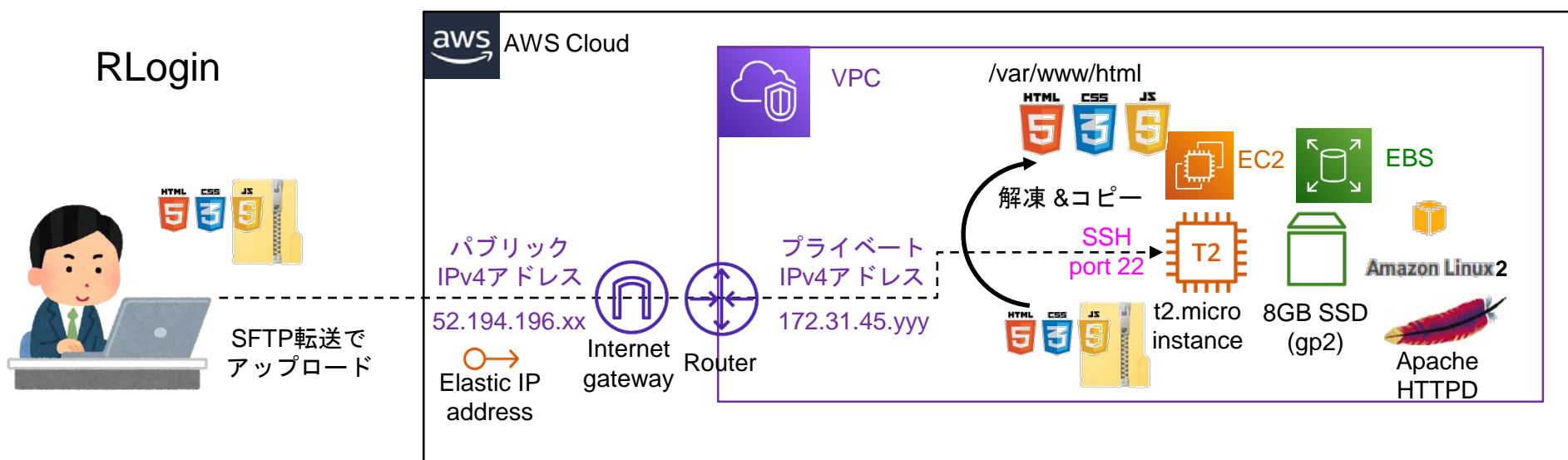
- <http://ec2-xx-yy-zz-ww.ap-northeast-1.compute.amazonaws.com/>

- ◆ スマートフォンでも確認してみよう



おめでとうございます！

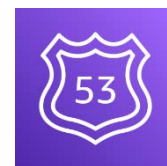
- 「CPS研究センター」のポータルサイトが立ち上がった
- Step3でやったことを下の図で確認しよう



- 商用リリースに向けてやること

- ◆ ドメイン名取得とDNSへの登録
- ◆ サーバ証明書の取得とSSL化

→ 無料枠では厳しいのでやりません



Amazon Route 53



AWS
Certificate
Manager



ハンズオン2: WEBアプリをデプロイしよう

Webアプリケーション, Webサービス

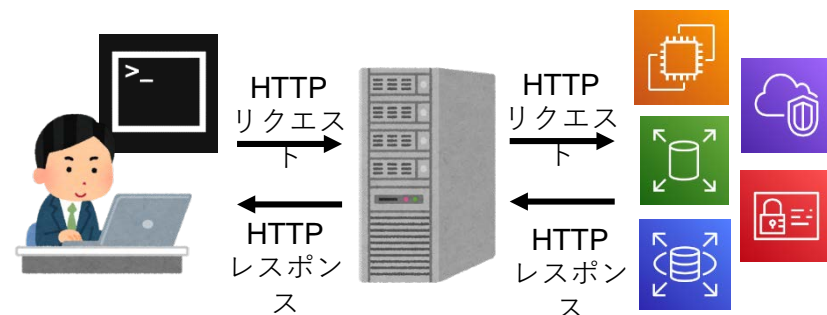
■ Webアプリケーション (動的Webサイト)

- ◆ Webの仕組みを利用して, ブラウザ上で動作するアプリケーション
- ◆ サーバ側のプログラムやDBと連動して, Webコンテンツを動的に生成する
- ◆ 情報提示だけでなく, 認証・認可, データ登録, 商取引等の情報処理が行われる



■ Webサービス (REST API)

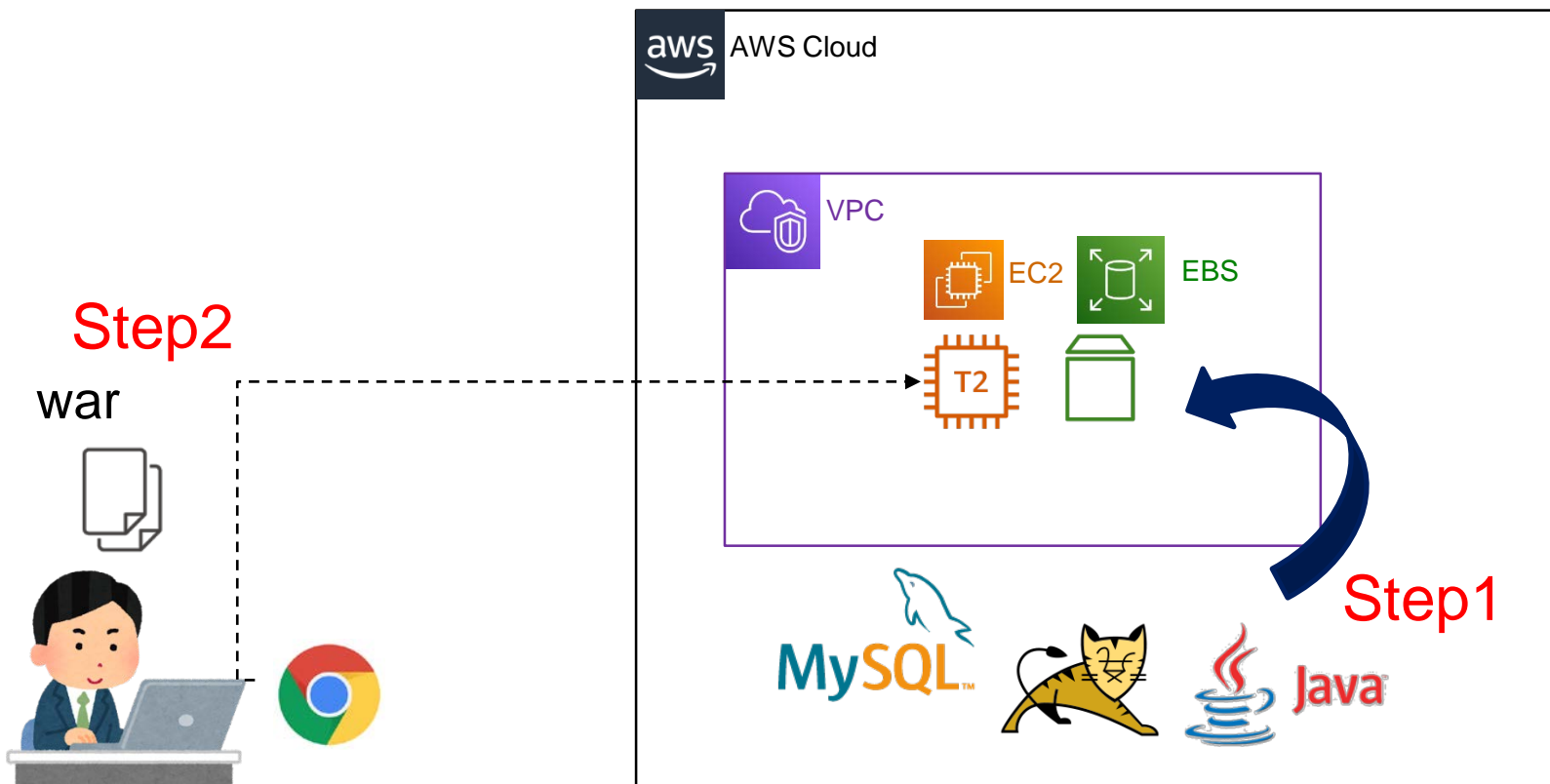
- ◆ Webの仕組みを利用して, プログラム同士がインターネットをまたいで, 処理を呼び出したりデータをやり取りする仕組み
- ◆ 要求メソッド(GET, POST, PUT, DELETE)でCRUD処理を指示. JSONやXMLでデータをやり取り
- ◆ AWSを含めほぼ全てのクラウドサービスは, Webサービスで制御されている





ハンズオン2: EC2にWebアプリを配備する

- Step1: Webアプリ実行環境をインストール
- Step2: アプリケーションアーカイブをデプロイ



Step1: Webアプリ実行環境をインストール (1/2)



■ wikiを参考にして, EC2に下記をインストールする

◆ Java-11 : Amazon Linux用のJava

- ※ yumで簡単に入ります

◆ Tomcat-9: Javaで動作するWebアプリケーションサーバ

- ※ yumで入らないので, 公式からダウンロードして入れます

◆ MySQL-8: 有名な関係データベース

- ※少し特殊な方法でyumで入れます



Webアプリ環境構築

- インストールはEC2上でroot権限が必要です

```
$ sudo su  
#
```

Java11のインストール

- yum であっさりとする

```
# yum install -y java-11
```

Tomcat9のインストール

- 作業用に/tmpに移動

```
# cd /tmp
```

- FTPサイトからtar ballをダウンロード

```
# wget https://downloads.apache.org/tomcat/tomcat-9/v9.0.46/bin/apache-tomcat-9.0.46.tar.gz
```



Step2: アプリケーションアーカイブをデプロイ

- wiki から demo.war をダウンロードする
- ハンズオン1の時と同様の手順で, EC2の/tmpに転送する
- 下記のコマンドで, Tomcatのwebappsにwarをコピー

1. 管理者権限を取得

```
$ sudo su
```

2. Tomcatのwebappsに移動

```
# cd /usr/tomcat/webapps
```

3. warをコピーする

```
# cp /tmp/demo.war .
```

4. ログを確認する

```
# tail /usr/tomcat/logs/  
catalina.out
```

5. ↓のようなログがあればOK

```
10-Jun-2021 11:37:03.258 INFO [main]  
org.apache.catalina.startup.HostConfig.deployWAR  
Deployment of web application archive  
[/usr/tomcat/webapps/demo.war] has finished in [8,200] ms
```

- ブラウザで以下のURLを開いてみる

- ◆ <http://EC2ホスト名:8080/demo/自分の名前/hello>
- ◆ 右のような猫の画像が出たらOK.
- ◆ あいさつを追加してみよう

Hello Spring Boot!

masa-nさん、Spring Bootへようこそじゃ！



あいさつを追加



Step2: アプリケーションアーカイブをデプロイ

■ 同様にcs27ch.war もデプロイ. 今回は事前にDBを作成

1. MySQLにログイン

```
# mysql -u root -p
```

2. データベース cs27chを作成

```
mysql > create database cs27ch;
```

3. ユーザspiralを作成

```
mysql > create user spiral identified by 'spiral';
```

4. cs27chの権限をspiralに委譲

```
mysql > grant all on cs27ch.* to spiral;
```

5. MySQLを抜ける

```
mysql > exit;
```

■ 前頁と同じ手順で, cs27ch.war を EC2に転送し, webappsにデプロイ

```
# cp /tmp/ch27ch.war /usr/tomcat/webapps/
```

■ ブラウザで以下のURLを開いてみる

- ◆ <http://EC2ホスト名:8080/cs27ch/>
- ◆ どこかでみたような...





サーバ費用はおいくら？

■ 今回のハンズオンでは無料枠を利用しているため無料

◆ 講義終了後の止め忘れに注意！超過分はカードに請求

- **停止**: サーバをシャットダウン. 課金がストップする. 再起動可能
- **終了**: サーバを消去. 課金がストップする. 再起動不可能



■ AWS Price Calculator から月額料金を見積もりできる

◆ 1か月連続稼働で12.06ドル (1,240円)

- 1日当たり40円. 1時間当たり1.8円. 場所代, 管理費, 光熱費込

◆ 内訳

- EC2 t2.microオンデマンドインスタンス
1 インスタンス x 0.0152 USD x 730 時間(1か月) = 11.10 USD
- EBS 汎用SSD (gp2) 8GB
8 GB x 0.12 USD x 1 インスタンス = 0.96 USD



AWS Cost Explorer





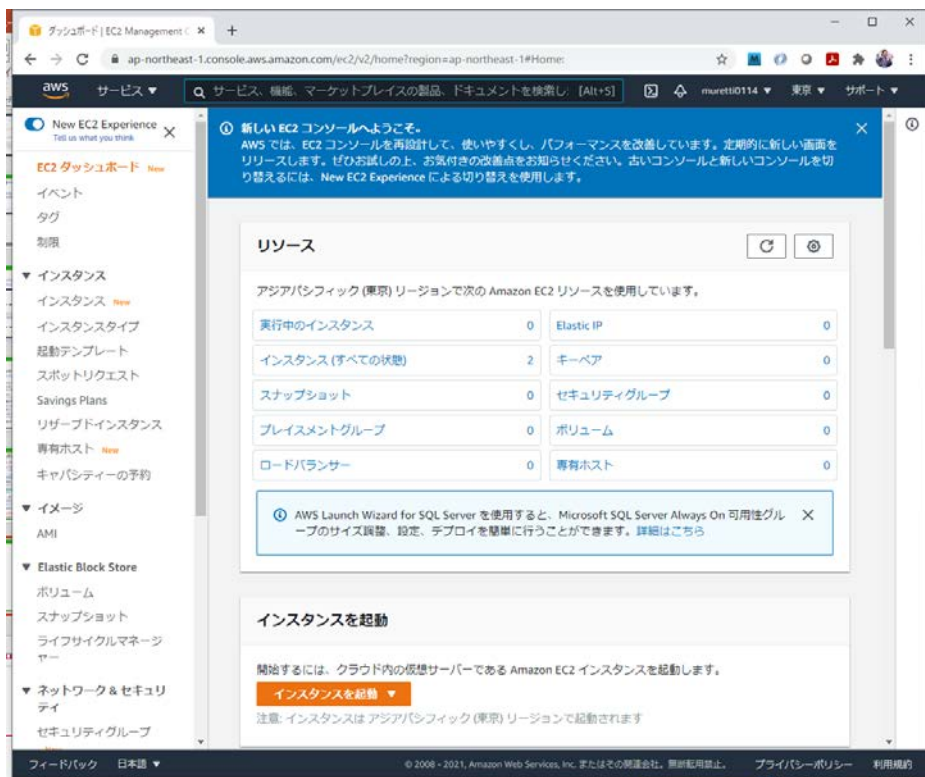
※重要: リソースの後片付けをしよう

- **つけっぱなしだと課金される**ので以下の手順で片付ける
 - ◆ EC2インスタンス:「インスタンスの状態」を変更する
 - 「インスタンスの停止」: サーバを一時停止する. 課金がストップする
 - 「インスタンスの終了」: サーバを消去する. 再起動不可能になる
 - ◆ 停止・終了したらElastic IPを解放(リリース)すること
 - EC2インスタンス実行時は無料. 停止・終了した状態では課金される(1時間0.5円). 使わないリソースを占有してしまうためコストが発生する
 - 参考: <https://dev.classmethod.jp/articles/cost-of-eip/>
 - ◆ ロードバランサーは「アクション」→「削除」で削除. 一時停止できない
 - ◆ VPCは基本的に無料. 全リソースを消したい場合は「VPCダッシュボード」から削除
- AWS 無料枠: 12ヶ月有効. 以下の制限あり
 - ◆ EC2: 750時間まで(つけっぱなしで約1か月)
 - ◆ EBS: 30GBまで. スナップショットは1GB(差分)
 - ◆ ELB: 750時間まで(つけっぱなしで約1か月)
 - ◆ 無料枠ぎりぎりまで楽しむのもアリ(通知等は来ないので忘れないように)
- **コンテンツやアプリは, 手元の環境にバックアップ**しておこう
 - ◆ クラウドは「使い捨て」. さっと立ち上げてさっと消す



ダッシュボードのリソースから確認する

- EC2ダッシュボードおよびVPCダッシュボードからリソースが片付けられたかを確認する



新しい EC2 コンソールへようこそ。
AWS では、EC2 コンソールを再設計して、使いやすくし、パフォーマンスを改善しています。定期的に新しい画面をリリースします。ぜひお試しのし、お気付きの改善点をお知らせください。古いコンソールと新しいコンソールを切り替えるには、New EC2 Experience による切り替えを使用します。

リソース

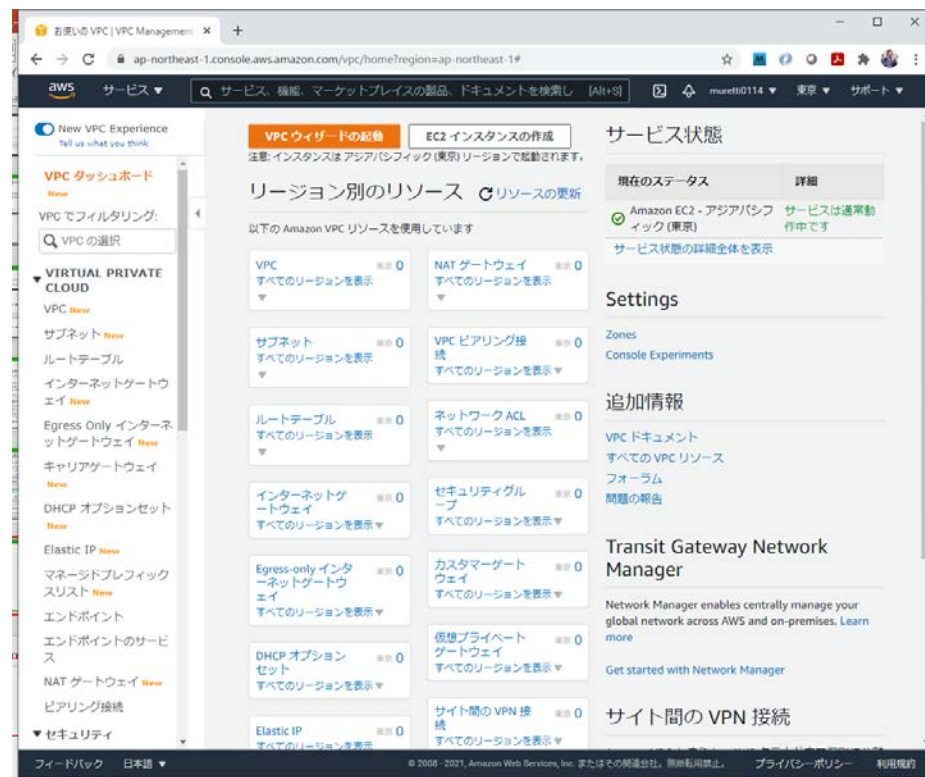
アジアパシフィック (東京) リージョンで次の Amazon EC2 リソースを使用しています。

リソース	数	リソース	数
実行中のインスタンス	0	Elastic IP	0
インスタンス (すべての状態)	2	キーペア	0
スナップショット	0	セキュリティグループ	0
プレースメントグループ	0	ボリューム	0
ロードバランサー	0	専用ホスト	0

インスタンスを起動

開始するには、クラウド内の仮想サーバーである Amazon EC2 インスタンスを起動します。
注意: インスタンスはアジアパシフィック (東京) リージョンで起動されます

ボタン: **インスタンスを起動**



リージョン別のリソース

以下の Amazon VPC リソースを使用しています

リソース	数	リソース	数
VPC	0	NAT ゲートウェイ	0
サブネット	0	VPC ピアリング接続	0
ルートテーブル	0	インターネット ACL	0
Egress Only インターネットゲートウェイ	0	セキュリティグループ	0
インターネットゲートウェイ	0	カスタマーゲートウェイ	0
DHCP オプションセット	0	仮想プライベートゲートウェイ	0
Elastic IP	0	サイト間の VPN 接続	0

- お疲れさまでした！！