

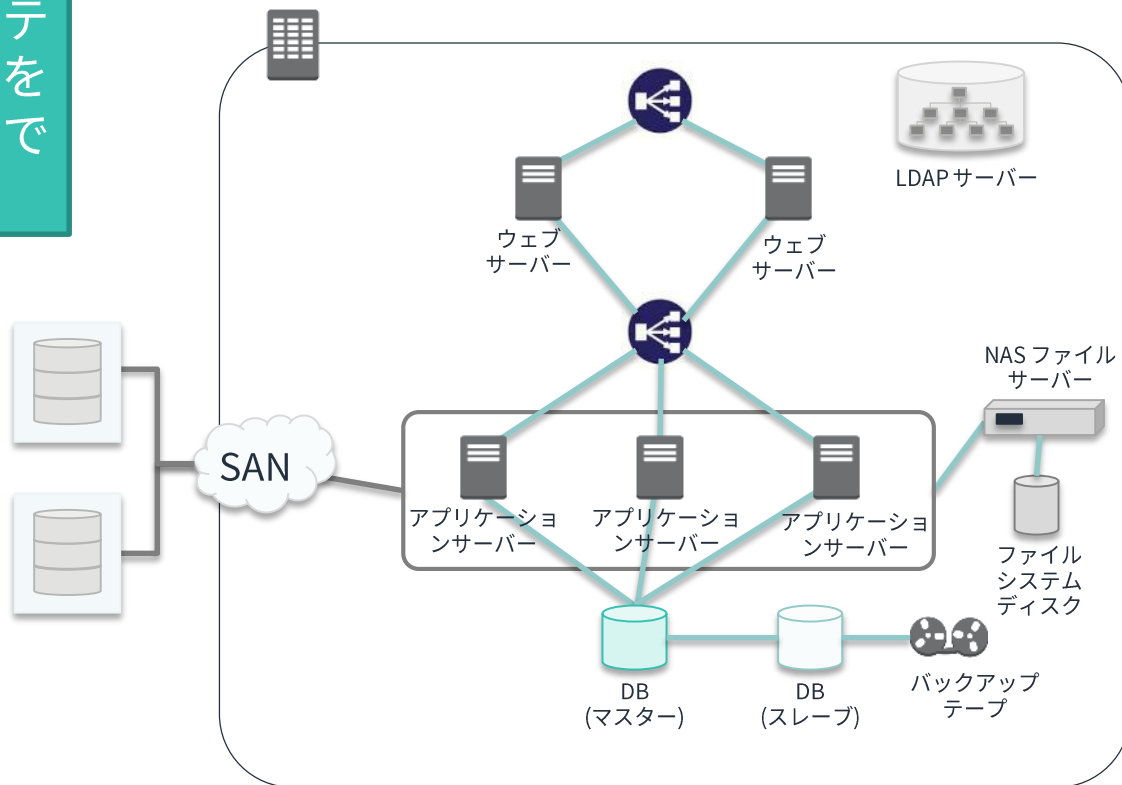


AWSOME DAY

セッション 2: AWS 上でのアプリケーション開発の基礎

オンプレミスからAWS への移行

オンプレミスのシステムをアーキテクチャを変えずに単純に移行できるのか？



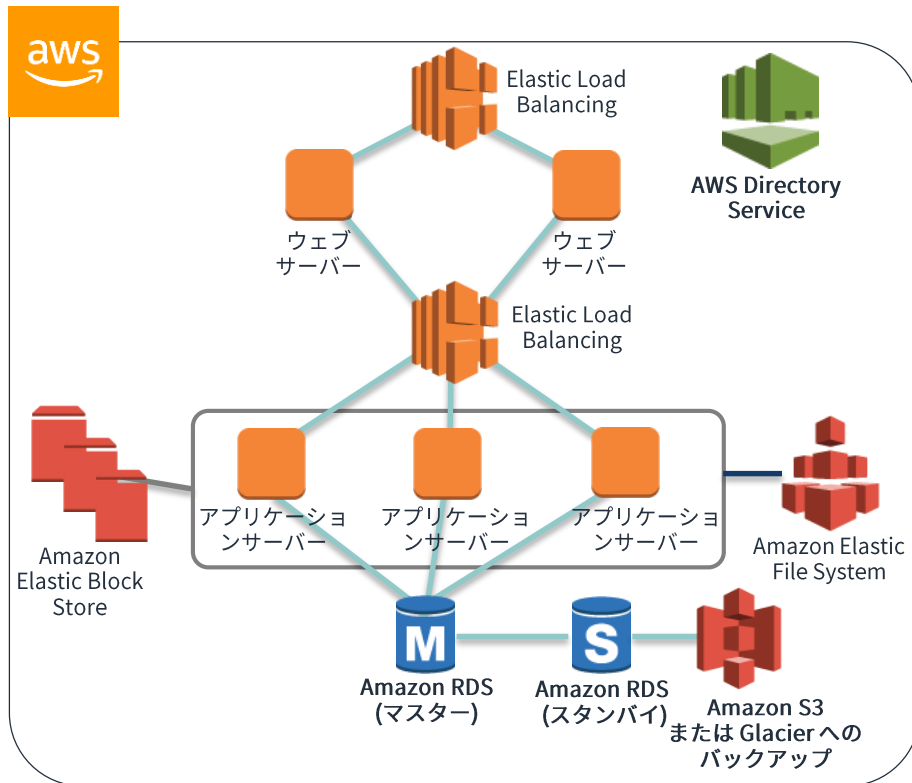
オンプレミスからAWS への移行

コアサービスを利用することで可能



開発者視点

インフラの上で動くアプリケーションはオンプレミスと変わらない



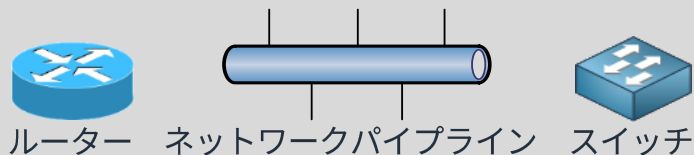
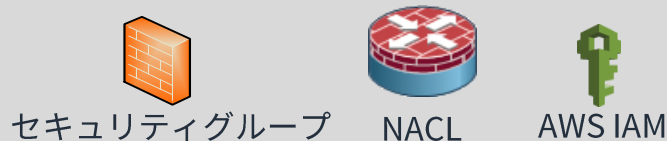
AWS のコアインフラストラクチャとサービス

従来のインフラストラクチャ

アマゾンウェブサービス



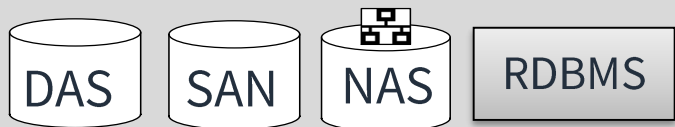
セキュリティ



ネットワー
キング



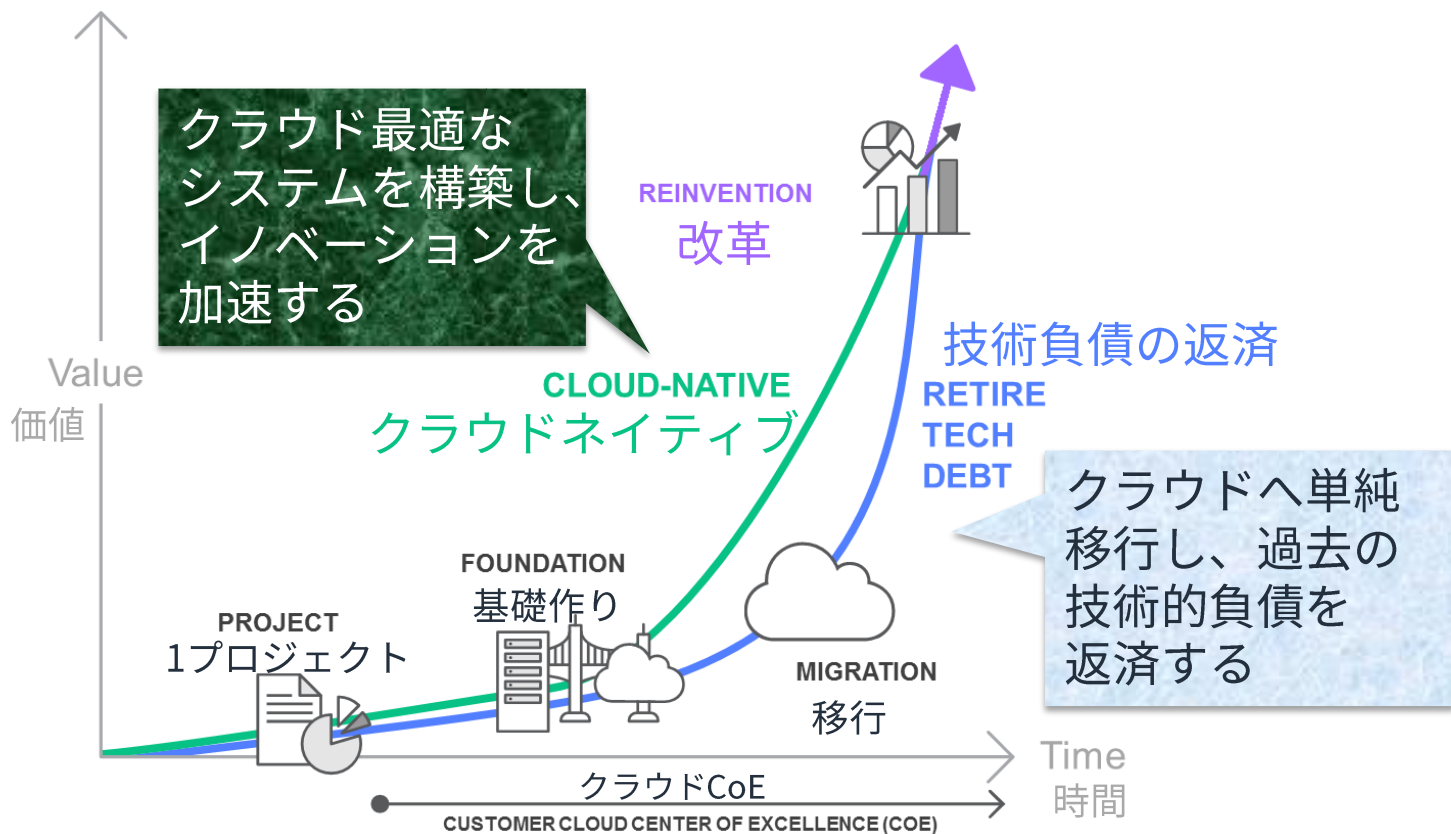
サーバー



ストレージ
データベース



クラウドジャーニーは改革に向けて2つの道を巡る



100 以上のAWS サービス群

お客様のアプリケーション

| | | | | | | |
|---|---|---|---|--|--|---|
|  ライブラリ & SDKs Java, PHP, .NET, Python, Ruby |  管理インター フェイス Management Console, CLI |  認証とログ IAM, Cloud Trail, Cloud HSM, Config |  ディレクトリ Directory Service |  モニタリング Cloud Watch, Trusted Advisor |  コード管理 CodeDeploy, CodeCommit, CodePipeline |  デプロイと自動化 Elastic Beanstalk, Cloud Formation, OpsWorks |
|---|---|---|---|--|--|---|


 **エンタープライズアプリ**
WorkSpaces, WorkDocs,
WorkMail

 **モバイルサービス**
Mobile Analytics,
Cognito, SNS, IoT, Pinpoint

 **人工知能(AI)**
Amazon Lex, Machine
Learning, Polly,
Rekognition

 **データベース**
RDS, DynamoDB, Redshift,
ElastiCache

 **アプリケーションサービス**
AppStream, Elasticsearch, SWF,
SQS, SES



 **分析**
Elastic MapReduce,
Kinesis, Athena, Data
Pipeline

 **コンピュー処理**
EC2, Auto Scaling, Lambda
Elastic Load Balancing,
EC2 Container Service

 **ストレージ**
EBS, S3, Glacier, EFS,
Storage Gateway

 **コンテンツ配信**
CloudFront

 **ネットワーク**
VPC, Route 53, Direct Connect

  **グローバルインフラ**
リージョン、アベイラビリティゾーン、エッジロケーション



Webサイトのデータはどこに保存する？

ユーザ
情報

セッション情報

ショッピング
カート



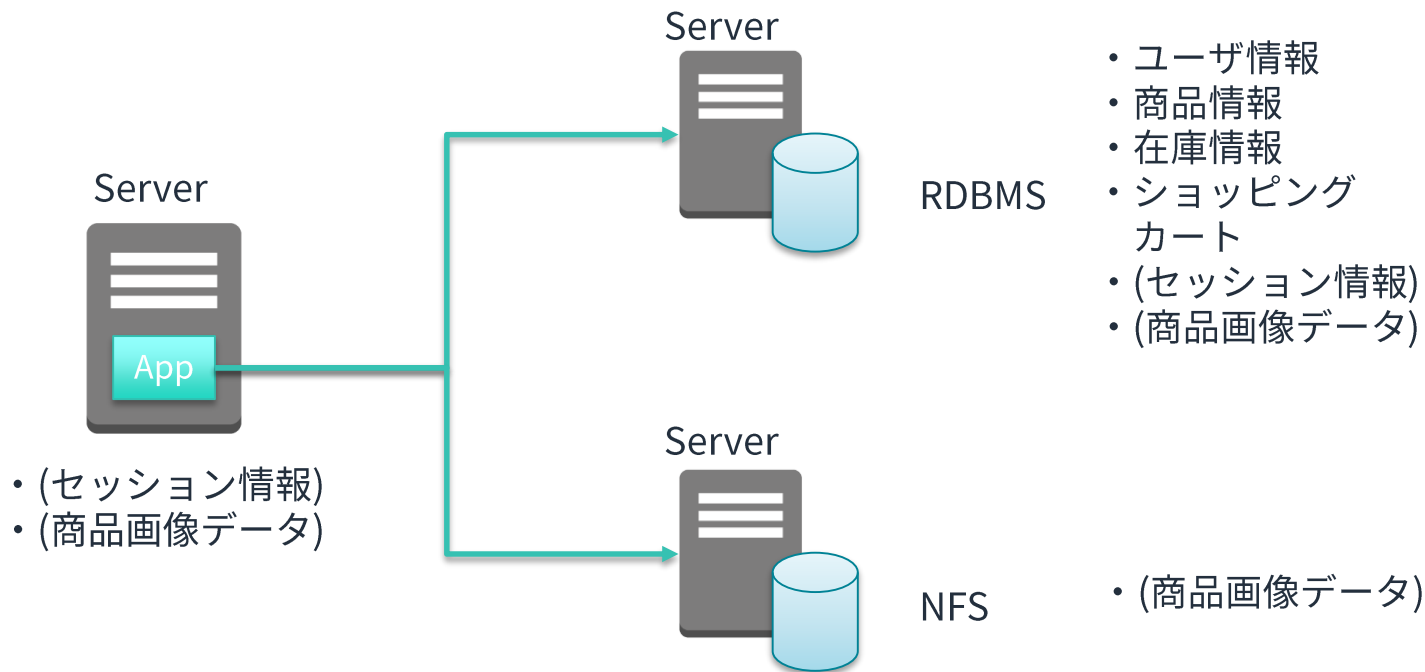
商品画像
データ

商品情報

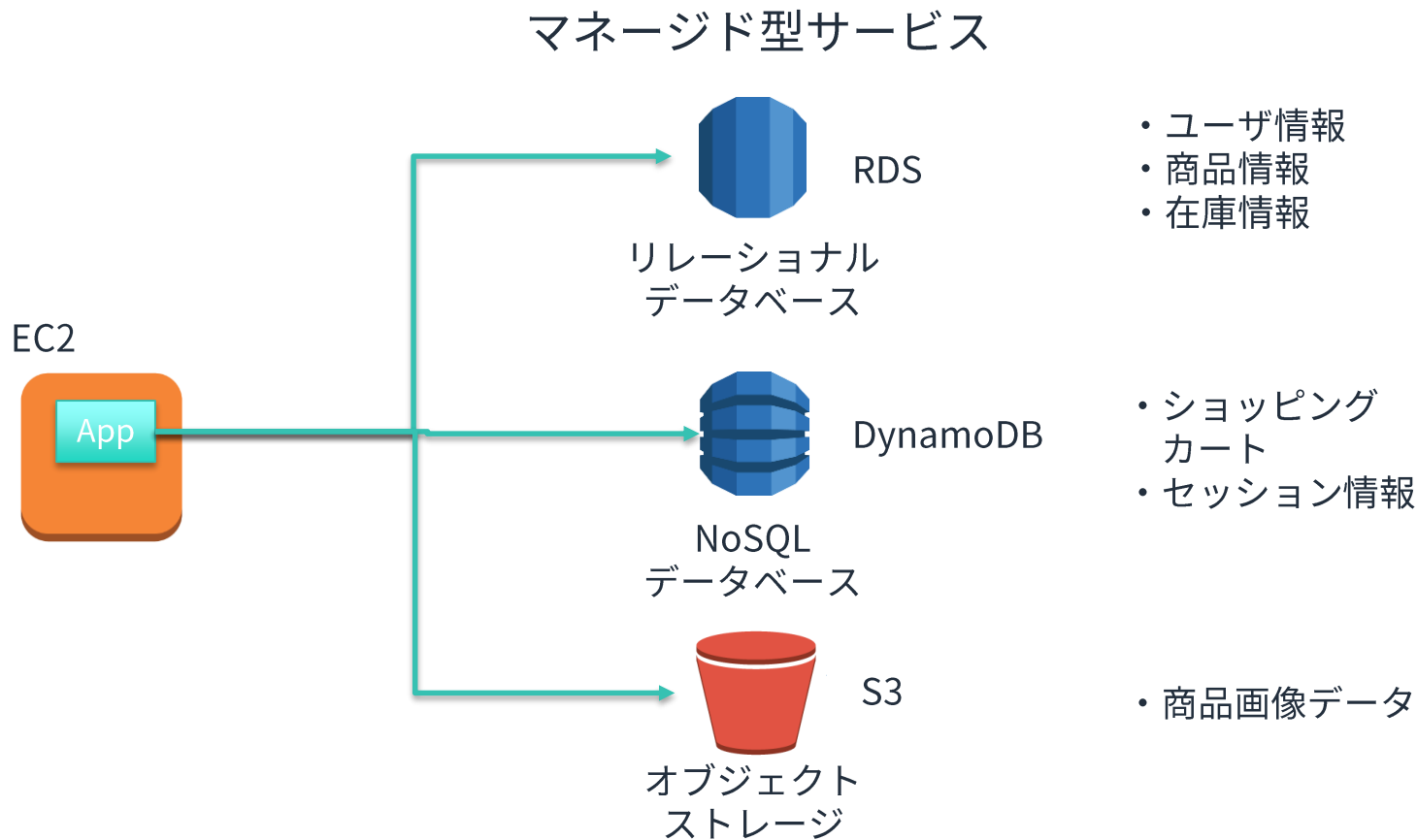
在庫情報



オンプレミスの世界



AWS では最適なマネージド型サービスを選択可能



マネージド型サービスとは？

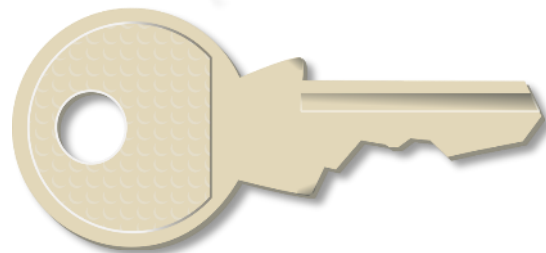
アンマネージド型とは

スケーリング、耐障害性、
および可用性をお客様が管理

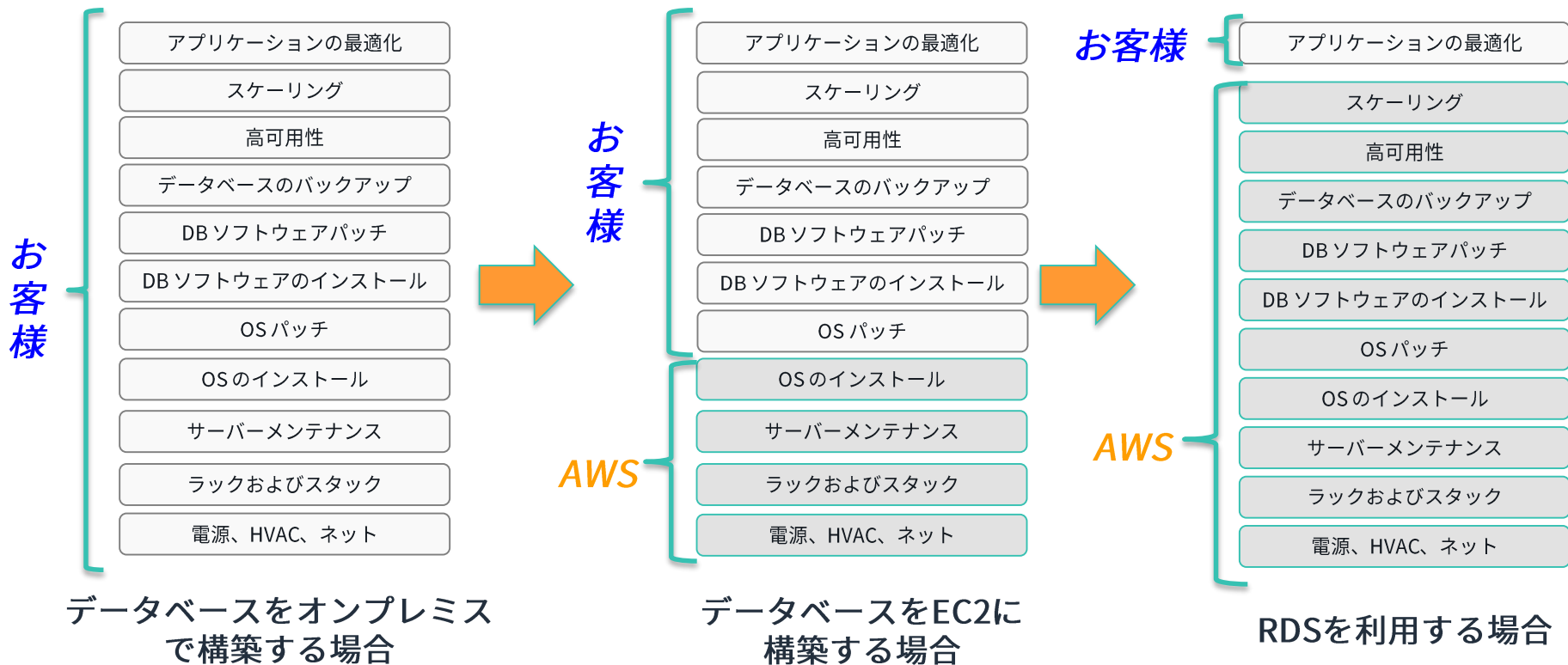


マネージド型とは

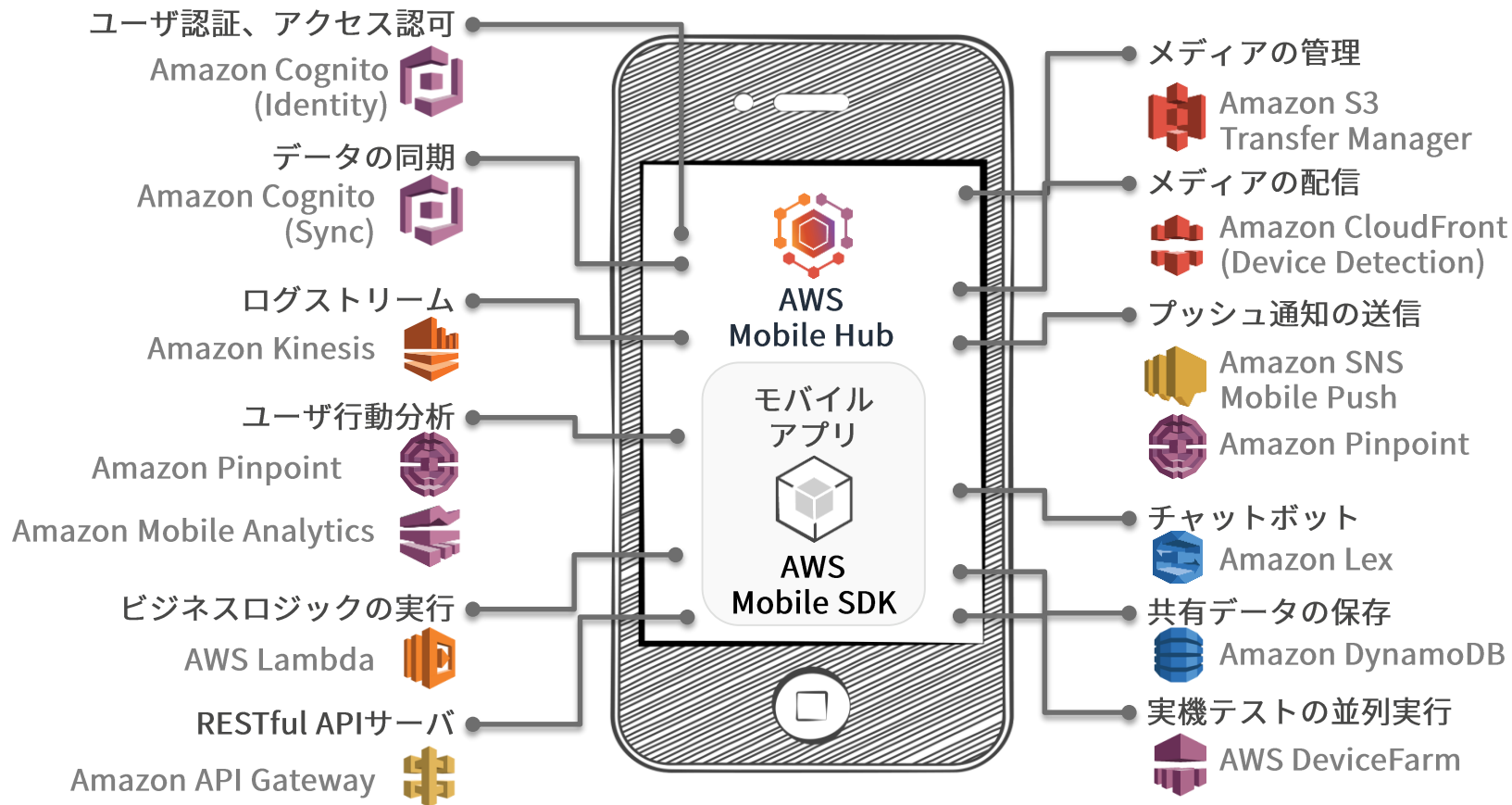
スケーリング、耐障害性、
および可用性は一般的に
サービスに組み込み

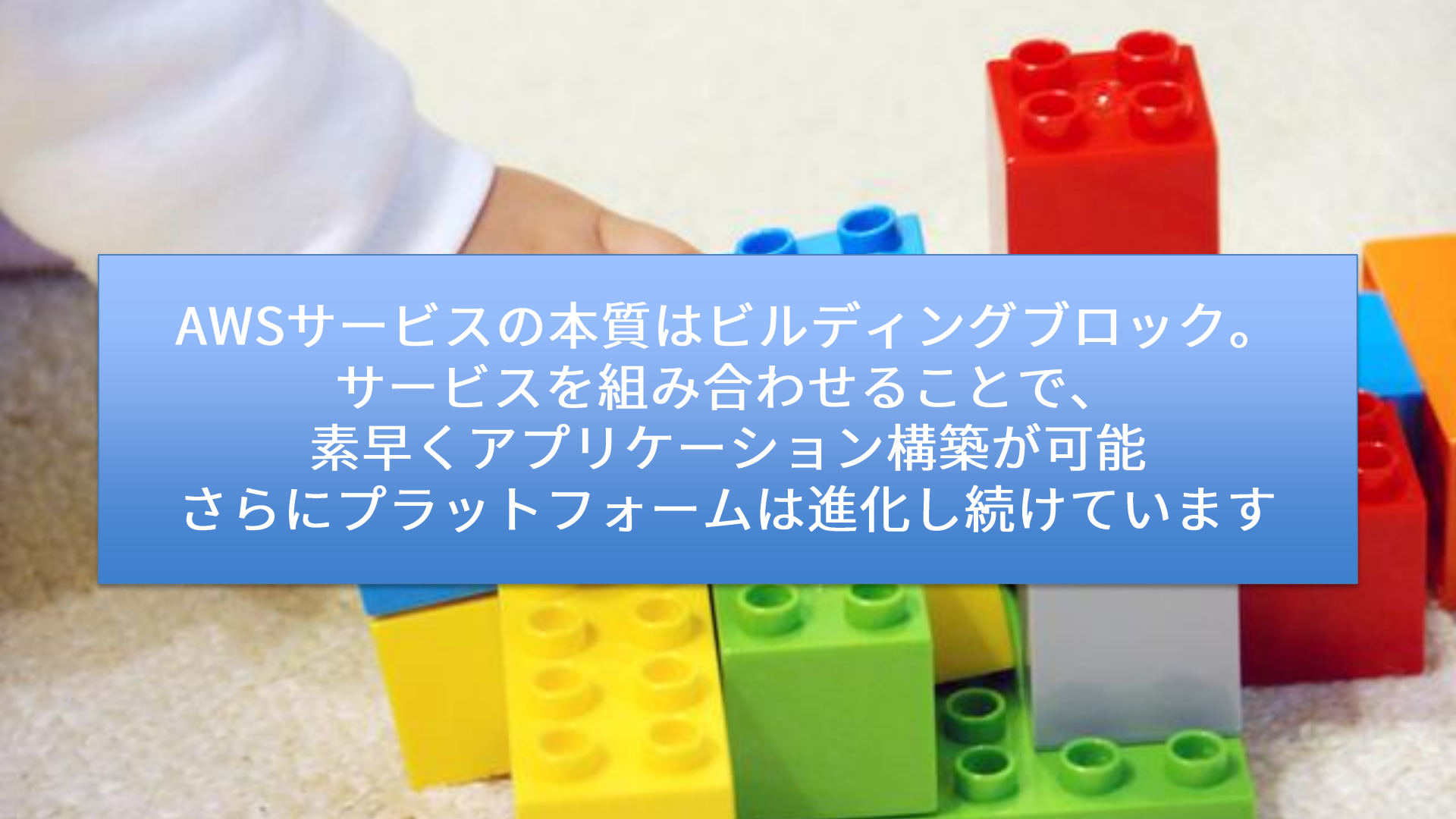


マネージド型サービスの利点とは？



AWS で利用できるモバイルサービス



A close-up photograph of a child's hand, wearing a white sleeve, reaching towards a collection of colorful LEGO bricks. The bricks are in various colors including red, blue, yellow, and green. The child is building a structure, with a red brick being placed on top of a blue one. The background is a light-colored, textured surface.

AWSサービスの本質はビルディングブロック。
サービスを組み合わせることで、
素早くアプリケーション構築が可能
さらにプラットフォームは進化し続けています

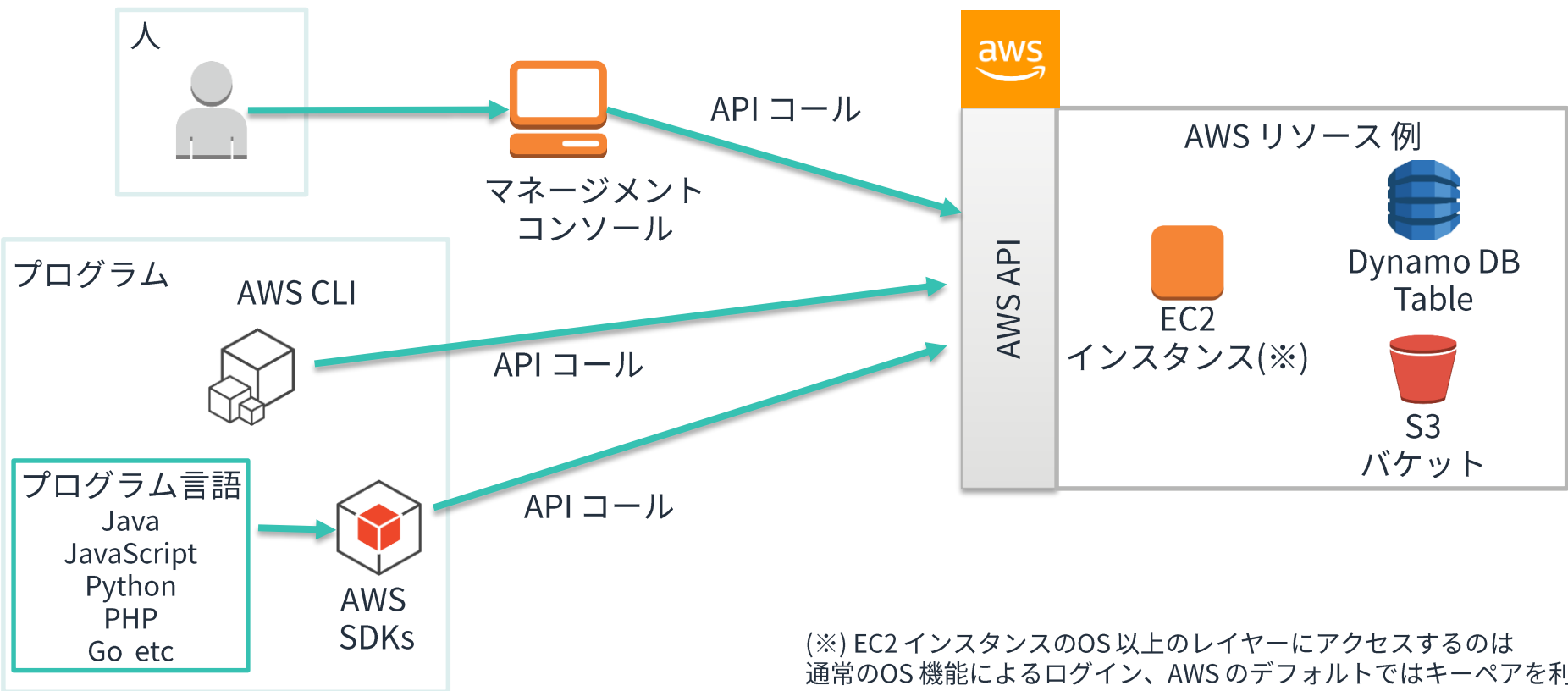
AWS API と AWS SDK

AWS リソースを操作したい場合

- マネージメントコンソールを利用する
- AWS CLI を利用する
- プログラムからAWS SDK を利用する
- プログラムからAWS API を直接利用する

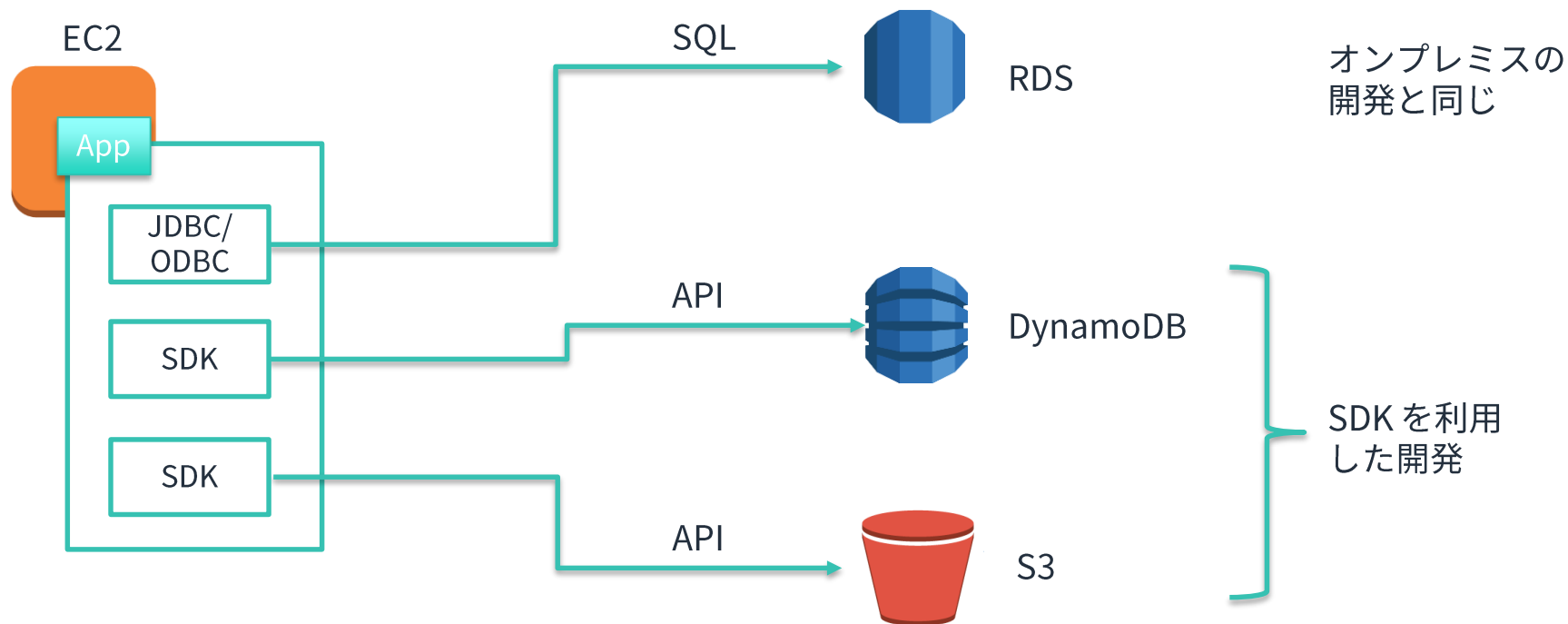


AWS リソースの操作



プログラムからSDK を利用する

マネージド型サービス



AWS SDKs



Android



iOS



Java



JavaScript



.NET



Node.js



PHP



Python (boto)



Ruby



Xamarin



AWS CLI



AWS Toolkit
for Eclipse



AWS Toolkit
for Visual
Studio



AWS Tools
for Windows
PowerShell

開発者が押さえておくべき AWS のセキュリティ

AWS API の認証と認可

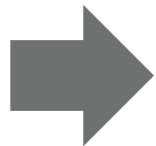
🟡 AWS API の呼び出しには適切なアクセス管理が必要

🟡 認証

- 誰がそのAPI を呼び出したのかを確認する
- 認証情報（ユーザ名／パスワード）を提示することで確認する

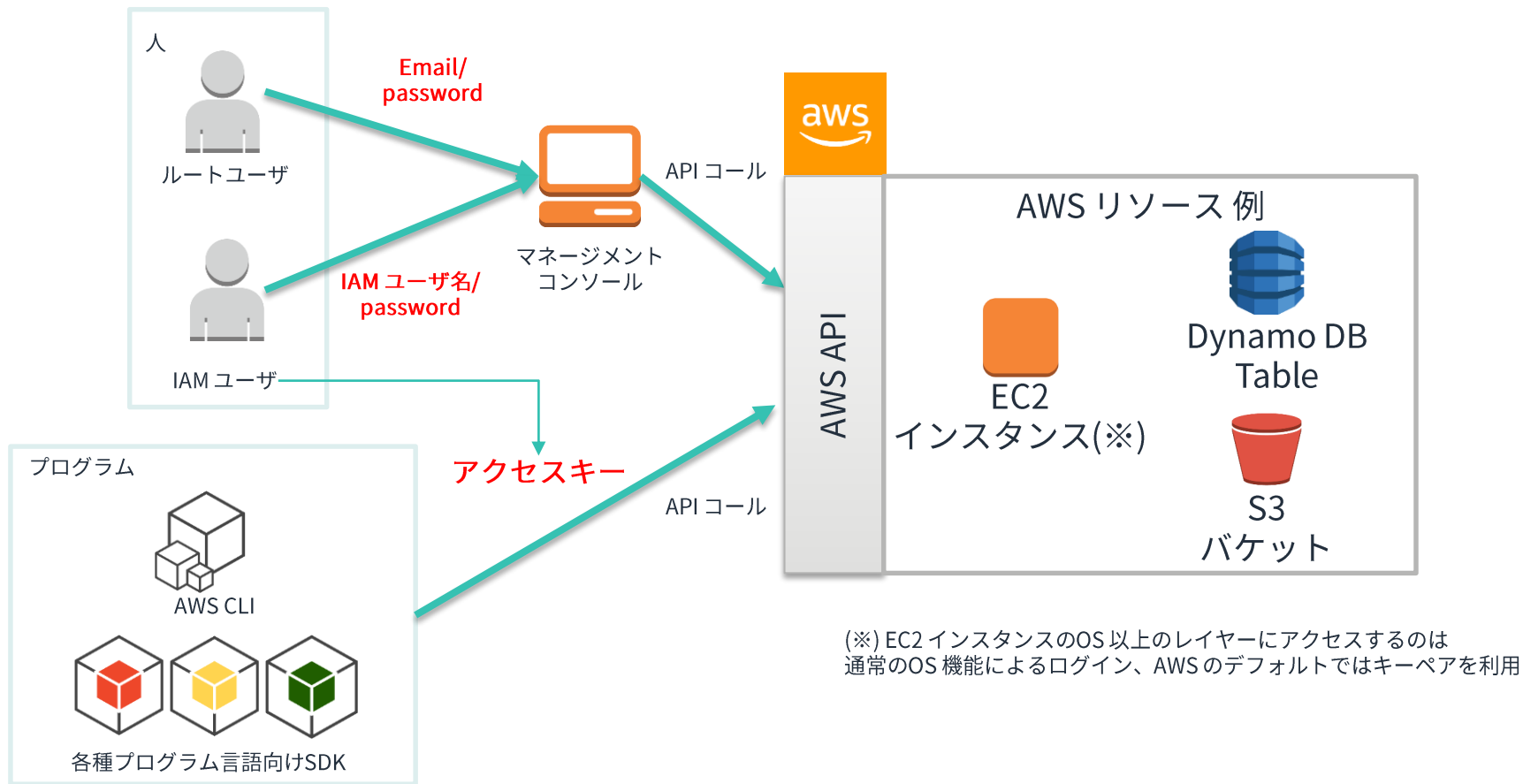
🟡 認可

- API を呼び出した人（もしくはプログラム）が、そのAPIを実行できる権限を持っているかを確認する
- 以下の2つ観点でアクセスできるかをチェックする
 - どのリソースに対して
 - どのような操作（＝API コール）



AWS Identity And Access Management (IAM)

AWS IAM 認証



AWS IAM 認証: 人の認証



認証

AWS マネジメントコンソール

- ユーザー名とパスワード



IAM ユーザー

Account:

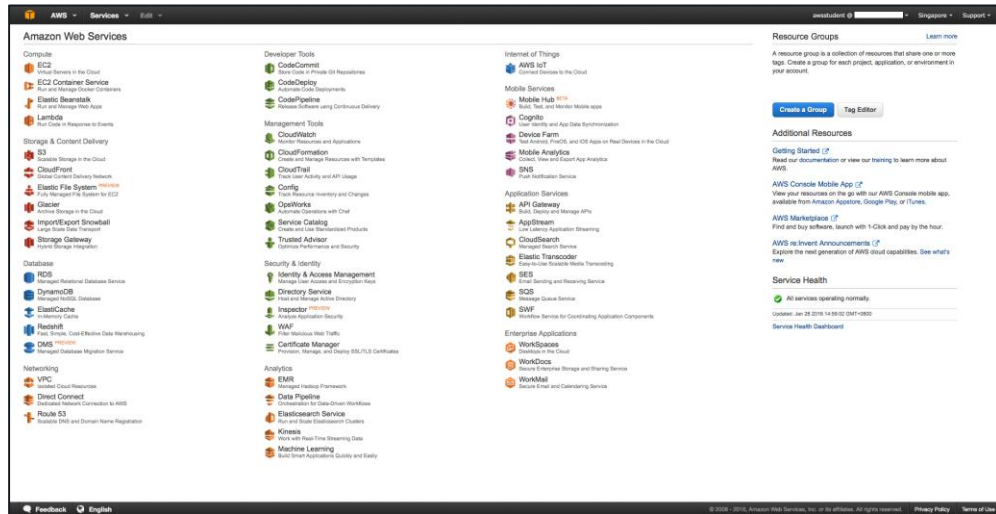


User Name:

Password:

MFA users, enter your code on the next screen.

Sign In



AWS IAM 認証: プログラムの認証



認証

● AWS CLI または SDK API

- アクセスキーID とシークレットアクセスキー



IAM ユーザー

アクセスキー ID: AKIAIOSFODNN7EXAMPLE
シークレットアクセスキー: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY

AWS CLI

```
~$ aws configure
AWS Access Key ID [*****O22A]:
AWS Secret Access Key [*****4m8i]:
Default region name [ap-southeast-1]:
Default output format [json]:
```

AWS SDK および API



Java



Python



.NET

アクセスキーの設定



静的なアクセスキーを保存

- 認証情報プロファイル(ユーザディレクトリの .aws/credentials) に保存

認証情報プロファイルのファイル

Linux、OS X、Unix: ~/.aws/credentials

Windows: C:\Users\USERNAME\.aws\credentials

[default]

aws_access_key_id = your_access_key_id

aws_secret_access_key = your_secret_access_key

- 環境変数に設定
 - AWS_ACCESS_KEY_ID、AWS_SECRET_ACCESS_KEY

一時的認証情報

- IAM ロール + AWS STS (後述)



認可

🟠 ポリシー:

- 権限を記述している JSON ドキュメント
- ユーザー、グループ、
ロールに割り当てられる



IAM ユーザー



IAM グループ



IAM ロール

IAM ポリシー例



```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Stmt1453690971587",
      "Effect": "Allow",
      "Action": [
        "ec2:Describe*",
        "ec2:StartInstances",
        "ec2:StopInstances"
      ],
      "Resource": "*",
      "Condition": {
        "IpAddress": {
          "aws:SourceIp": "54.64.34.65/32"
        }
      }
    }
  ]
}
```

IAM ポリシーにより以下を設定

- どのリソースに対して
- どのような操作を
- 許可する or 拒否する
- (Option) 追加条件

Effect

- Allow or Deny

Resource

- 権限の及ぶ範囲

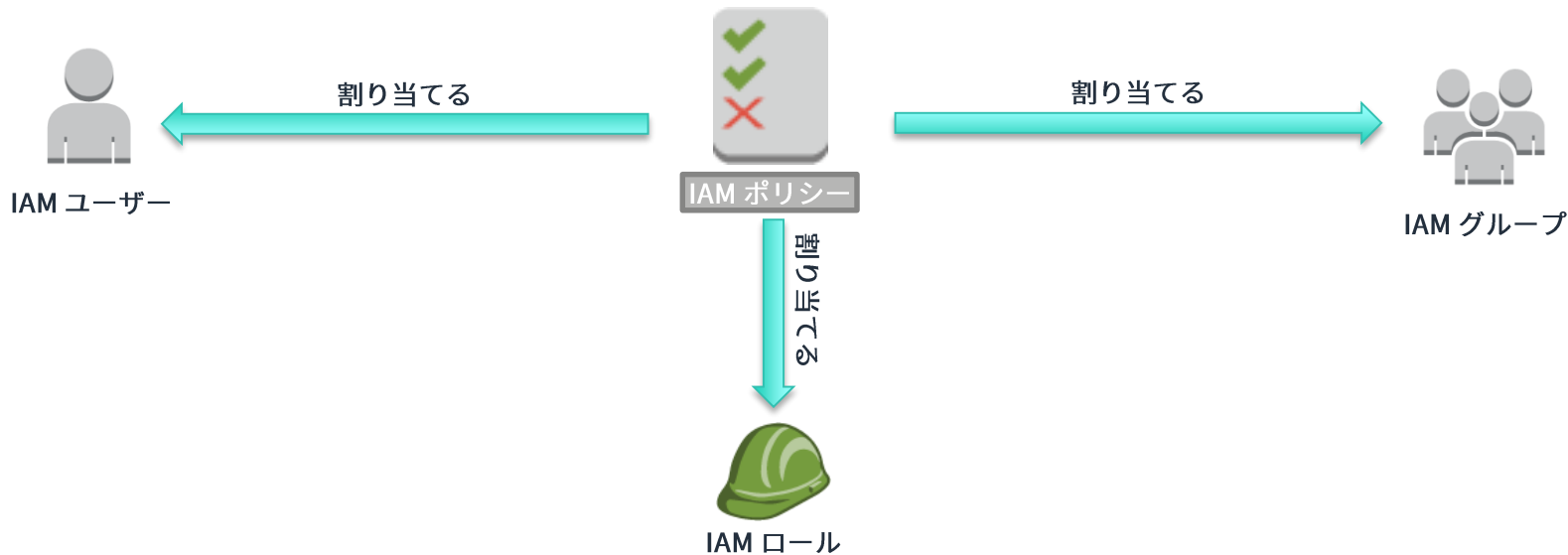
Action

- 権限の及ぶ操作

Condition

- 追加条件を記述可能
- 特定のIP アドレスからのリクエストのみを受け付けるなど

AWS IAM ポリシーを割り当てる



AWS IAM ロール

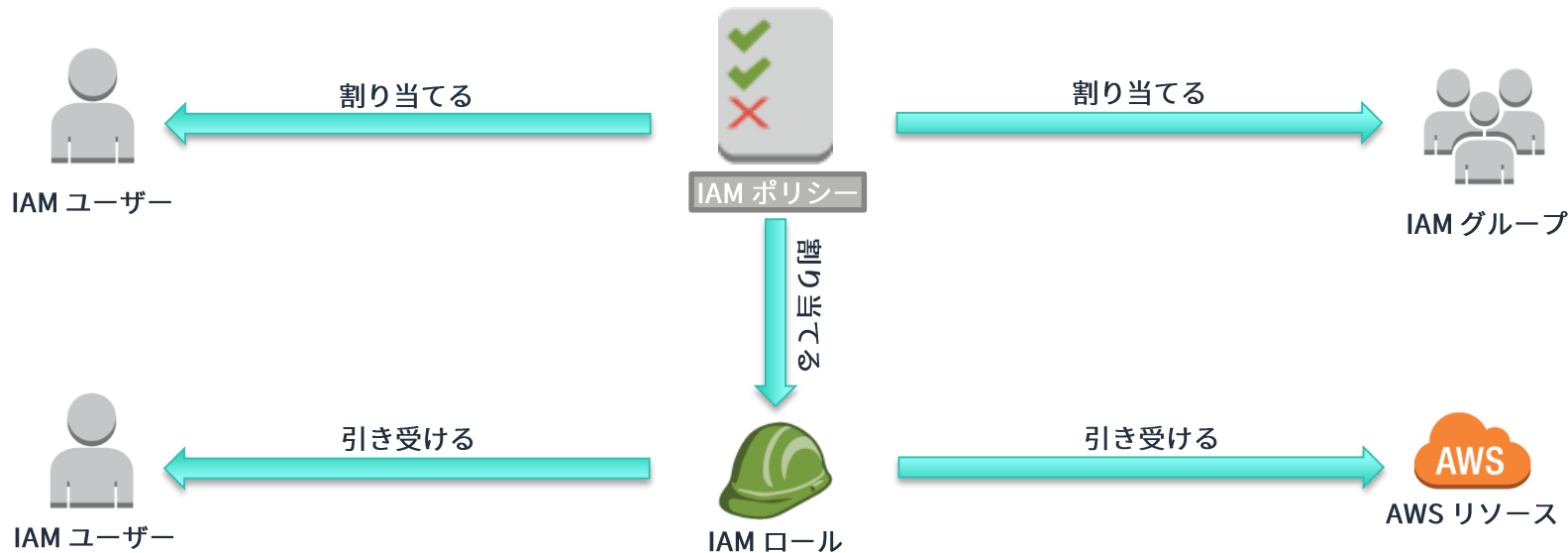


- IAM ロールにポリシーを適用する
- IAM ロールに直接関連付けられる認証情報はない
- IAM ユーザー、アプリケーション、およびサービスが IAM ロールを引き受けられる

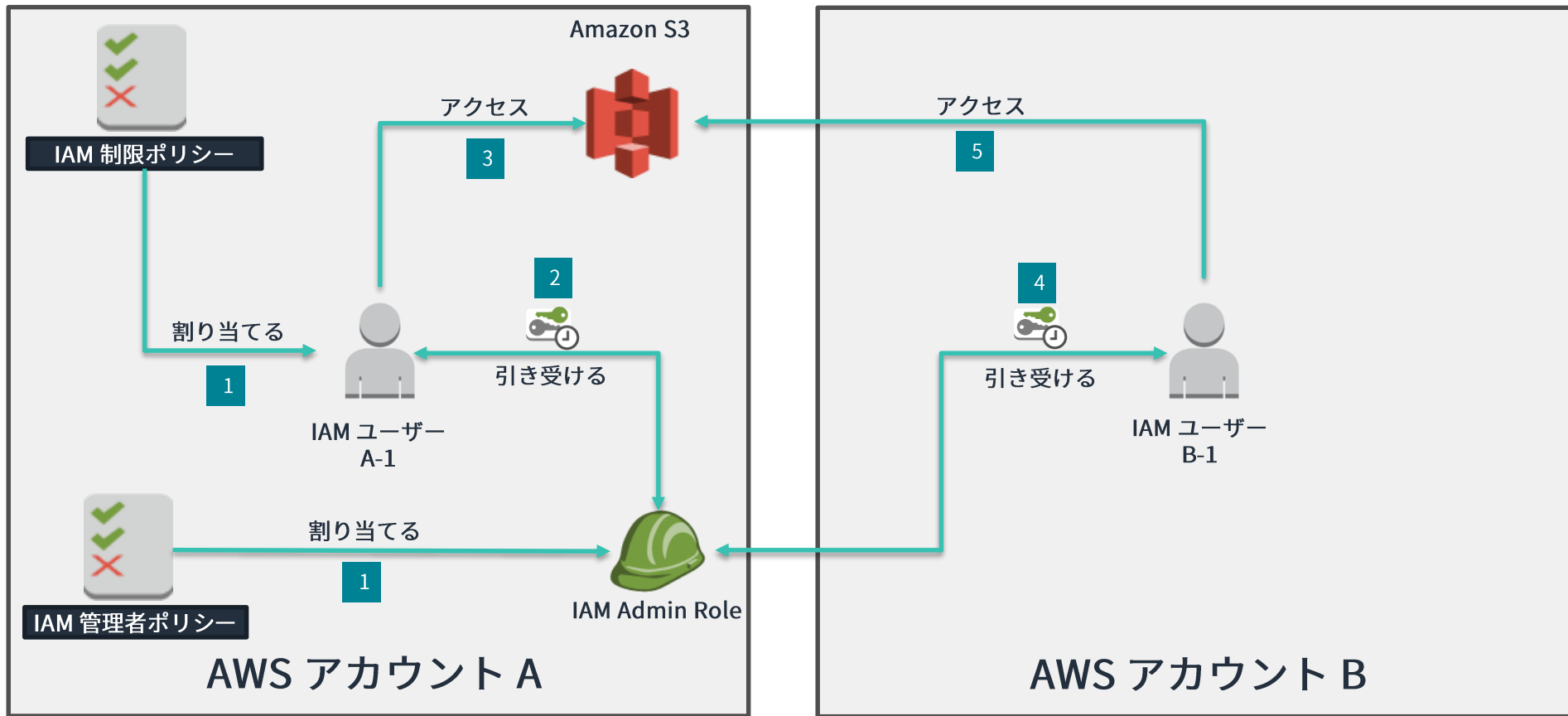


IAM ロール

AWS IAM ポリシーを割り当てる



AWS IAM ロール – ロールを引き受ける



アプリケーションの AWS リソースへのアクセス



- Amazon EC2 インスタンスでホストされている Python アプリケーションでは Amazon S3 とやり取りする必要がある

- AWS 認証情報には以下が必要である

- ~~オプション 1: Amazon EC2 インスタンスに静的な AWS 認証情報(アクセスキー)を保存する~~
- オプション 2: IAM ロールを利用して AWS のサービスおよびアプリケーションに一時的認証情報を配布する



IAM ロール

AWS IAM ロール – インスタンスプロファイル



Amazon EC2



1

インスタ
スの作成

IAM ロールの選択

2

AWS Services Edit

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Tag Instance 6. Configure Security Group 7. Review

Step 3: Configure Instance Details

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take advantage of the lower pricing, assign an access management role to the instance, and more.

Number of instances 1 Launch into Auto Scaling Group

Purchasing option ☐ Request Spot instances

Network vpc-5c (172.31.0.0/16) (default) Create new VPC

Subnet No preference (default subnet in any Availability Z Create new subnet

Auto-assign Public IP Use subnet setting (Enable)

Domain join directory None Create new directory

IAM role **PythonEC2AccessS3** Create new IAM role

Shutdown behavior

Enable termination protection

Monitoring ☐ Enable CloudWatch detailed monitoring Additional charges apply.

Tenancy Shared - Run a shared hardware instance Additional charges will apply for dedicated tenancy.

Advanced Details

Amazon S3



4

S3とやり取りする
アプリケーション



アプリケーション &



3

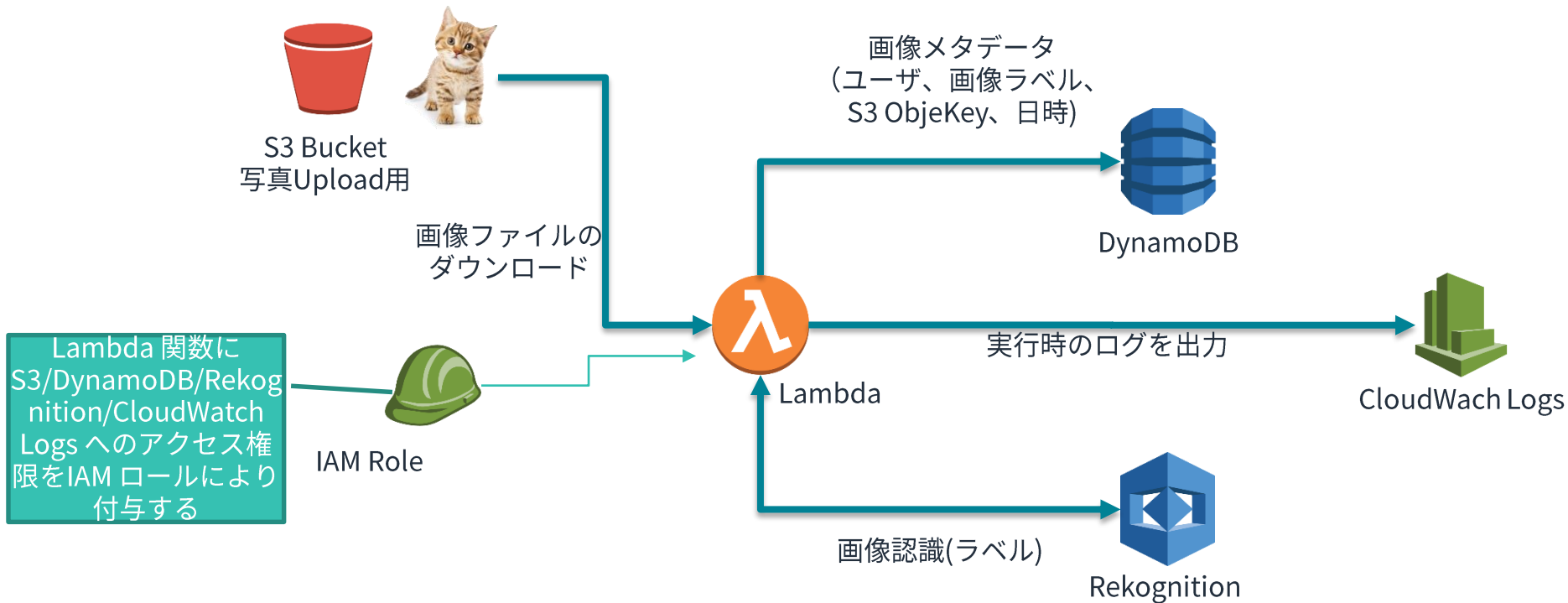
EC2 MetaData Service

<http://169.254.169.254/latest/meta-data/iam/security-credentials/rolename>

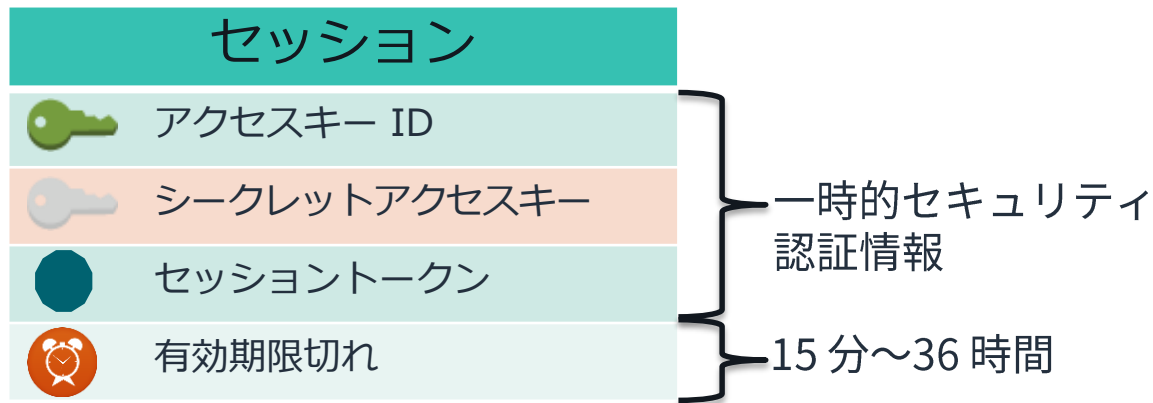
AWS IAM ロール: AWS リソース間の連携



- AWS リソースが別のAWS リソースと連携する場合もIAM ロールが利用される



一時的セキュリティ 認証情報 (AWS STS)



ユースケース

- クロスアカウントアクセス
- フェデレーション
- モバイルユーザー
- Amazon EC2 ベースのアプリケーションのキー更新

Lab01: 開発者IAM ユーザの作成およびIAM ロールの動作確認

ラボの目的

- IAM の操作方法を学び、IAM ユーザとIAM ポリシーを作成する
- EC2 インスタンスを起動しSSH で接続する方法を確認する
- アクセスキーを利用した認証とIAM ロールを利用した認証を実際に動作確認しながら理解する

Thank you