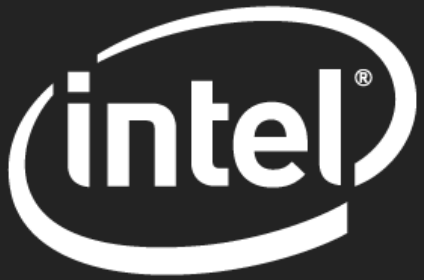

aws
DEVDAY
GLOBAL SERIES

THANKS TO OUR FRIENDS AT:



DevSecOps on AWS

- Policy in Code -

Tomoaki Sakatoku, Partner Solutions Architect

2017/06/01

amazon web services

Who I am…

酒徳 知明 (Tomoaki Sakatoku)

Partner Solutions Architect

- Ecosystem
- DevSecOps
- AWS Gameday

アジェンダ

お話**する**こと

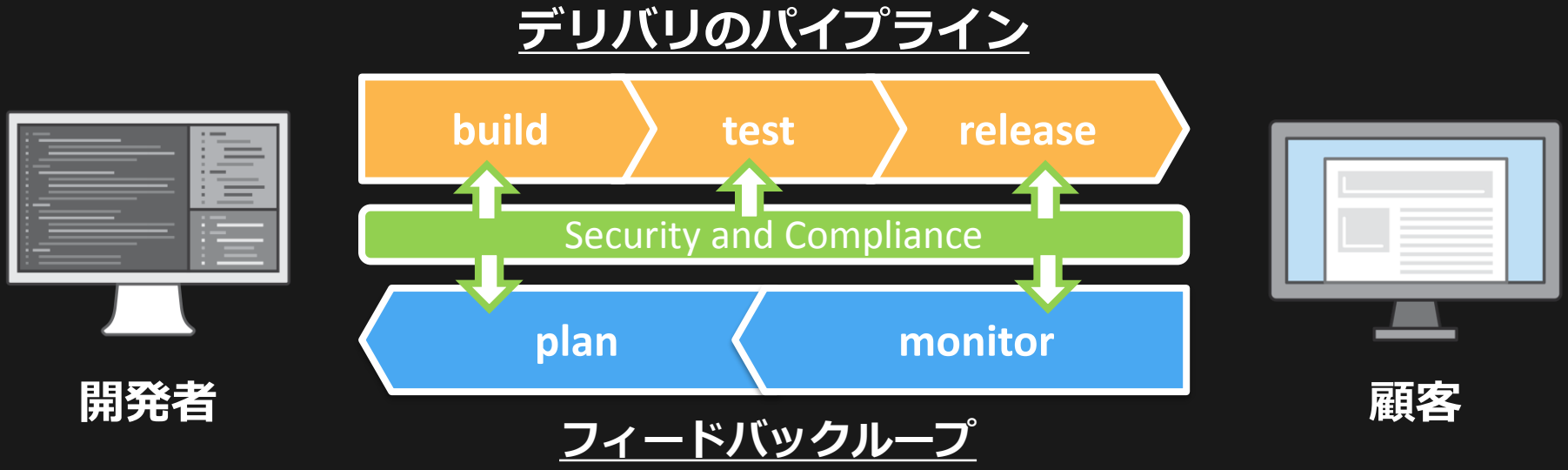
- DevSecOps について...
- DevSecOps を支えるAWS サービス インテグレーション
- コードリンクのご紹介は最後にまとめて

お話**しない**こと

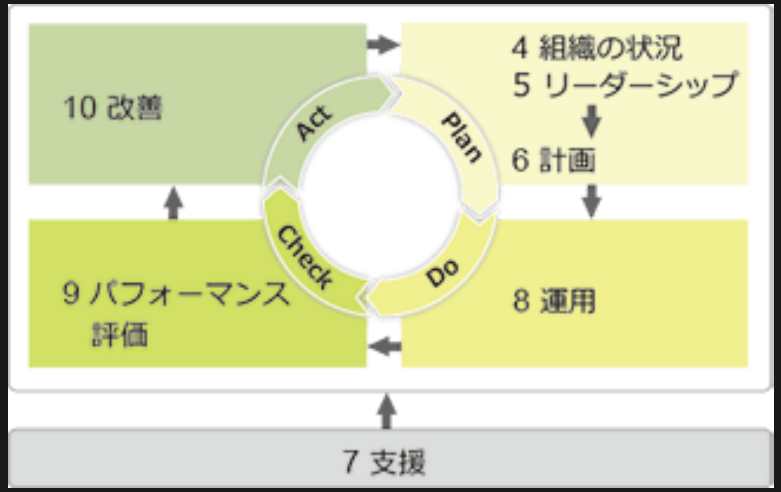
- DevOps ついて...
- DevSecOps を支えるAWS サービス単体の説明

DevOps … and DevSecOps

ソフトウェア開発のライフサイクル



DevSecOps 導入へのブロッカー

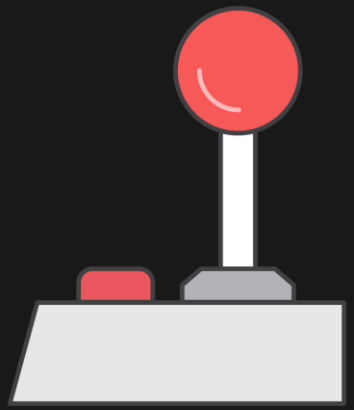


今までの常識 ≠ クラウドの常識

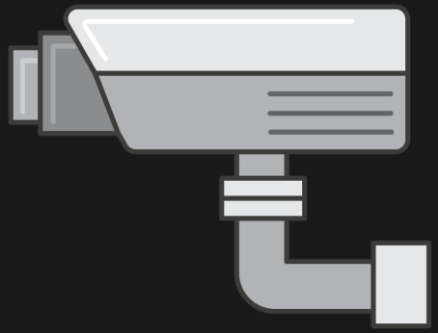
https://www.jqa.jp/service_list/management/service/iso27001/

Security Automation

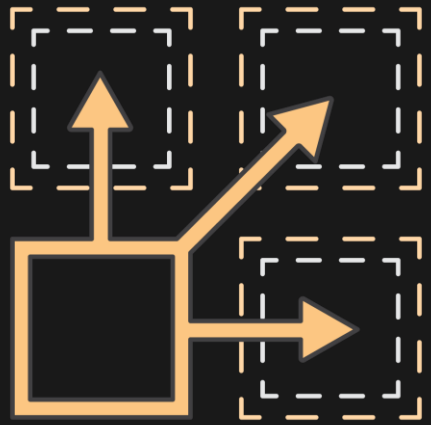
Security Control



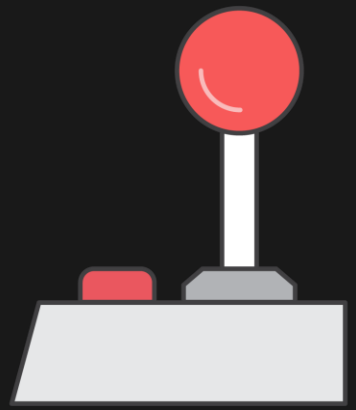
Proactive Monitoring



Security at Scale
Infrastructure as a code



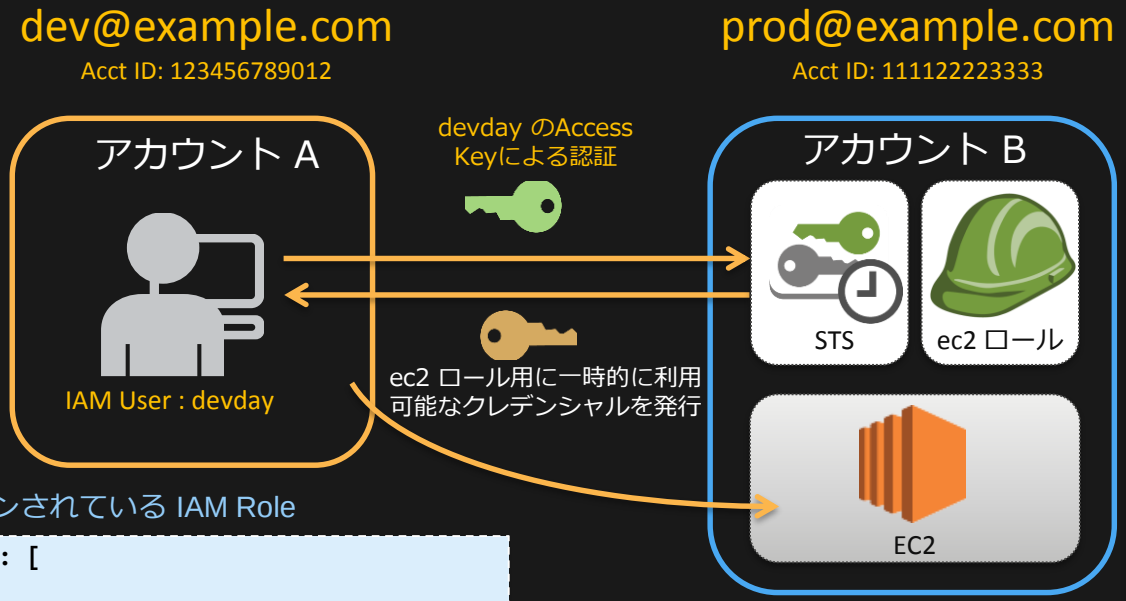
Security Control



Identity Access Management - IAM

- ユーザ/クレデンシャル管理
  - IAMユーザ / パスワード
  - MFA (多要素認証)
  - クレデンシャルのローテーション
- アクセス権限管理
  - IAMグループ
  - IAMポリシー
- 権限の委任と監査
  - IAMロール
  - Security Temporary Token

Identity Access Management - IAM Role & STS



ec2 ロールの権限

```
{ "Statement": [
  {"Effect": "Allow",
   "Action": "ec2:RunInstances",
   "Resource": [
      "arn:aws:ec2:*::image/ami-*",
      "arn:aws:ec2:*:*:network-interfac
      "arn:aws:ec2:*:*:instance/*",
      "arn:aws:ec2:*:*:subnet/*",
      "arn:aws:ec2:*:*:key-pair/*",
      "arn:aws:ec2:*:*:security-group/*
      ]
  },
}]}
```

Jeff にアサインされている IAM Role

```
{ "Statement": [
  {
   "Effect": "Allow",
   "Action": "sts:AssumeRole",
   "Resource":
       "arn:aws:iam::111122223333:role/ec2-role"
}]}
```

```
{ "Statement": [
  {
   "Effect":"Allow",
   "Principal":{"AWS":"123456789012"},
   "Action":"sts:AssumeRole"
}]}
```

ec2 ロールの信頼関係

IAM - AssumeRole によるクレデンシャルの生成

```
sts = boto3.client('sts')
assumedRoleObject = sts.assume_role(
        RoleArn = 'arn:aws:iam::123456789012:role/myRole_prod',
        RoleSessionName = 'MySessionName'
        )

# From the response that contains the assumed role, get the temporary
# credentials that can be used to make subsequent API calls
credentials = assumedRoleObject['Credentials']

ec2 = boto3.resource(
    'ec2',
    aws_access_key_id = credentials['AccessKeyId'],
    aws_secret_access_key = credentials['SecretAccessKey'],
    aws_session_token = credentials['SessionToken'],
)
```

IAM によるタグ管理 – Tag on Creation

```
{
    "Effect": "Allow",
    "Action": "ec2:RunInstances",
    "Resource": [
        "arn:aws:ec2:*:*:instance/*",
        "arn:aws:ec2:*:*:volume/*"
    ],
    "Condition": {
        "StringEquals": {
            "aws:RequestTag/CostCenter": "sa4005"
        }
    }
},
```

https://aws.amazon.com/jp/blogs/aws/new-tag-ec2-instances-ebs-volumes-on-creation/

git-secrets を使ったクレデンシャルの管理

- コミットの防止

```
$ git secrets --register-aws
$ git add git-secret.py
$ git commit -m "This is a test commit for git-secret"
git-secret.py:1:AWSAccessKeyId = "AKIAIOSFODNN1EXAMPLE"
git-secret.py:2:AWSSecretKey = "wJalrXUtnFEMI/K1MDENG/bPxRfiCYEXAMPLEK
[ERROR] Matched one or more prohibited patterns
```

- 既存レポジトリのスキャン

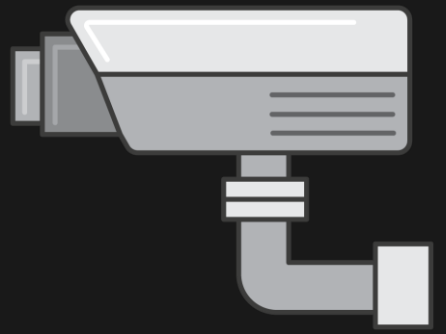
```
$ git secrets --scan
```

https://github.com/awslabs/git-secrets

You don't have permissions to use the AWS CloudTrail Console

If you need assistance, contact your System Administrator

Proactive Monitoring



Proactive Monitoring Lifecycle

- イベントトリガーをベースとした監視
- 修復を最終優先としたアクション
- マニュアル作業を極力減らした自動化
- 必要なログの収集・保存

DevSecOps を支えるAWS Services

AWS CloudTrail

- AWS Platform上で呼び出されたAPIに関するイベントを継続的にロギング&監視
- コンプライアンスの簡素化
- セキュリティの自動化

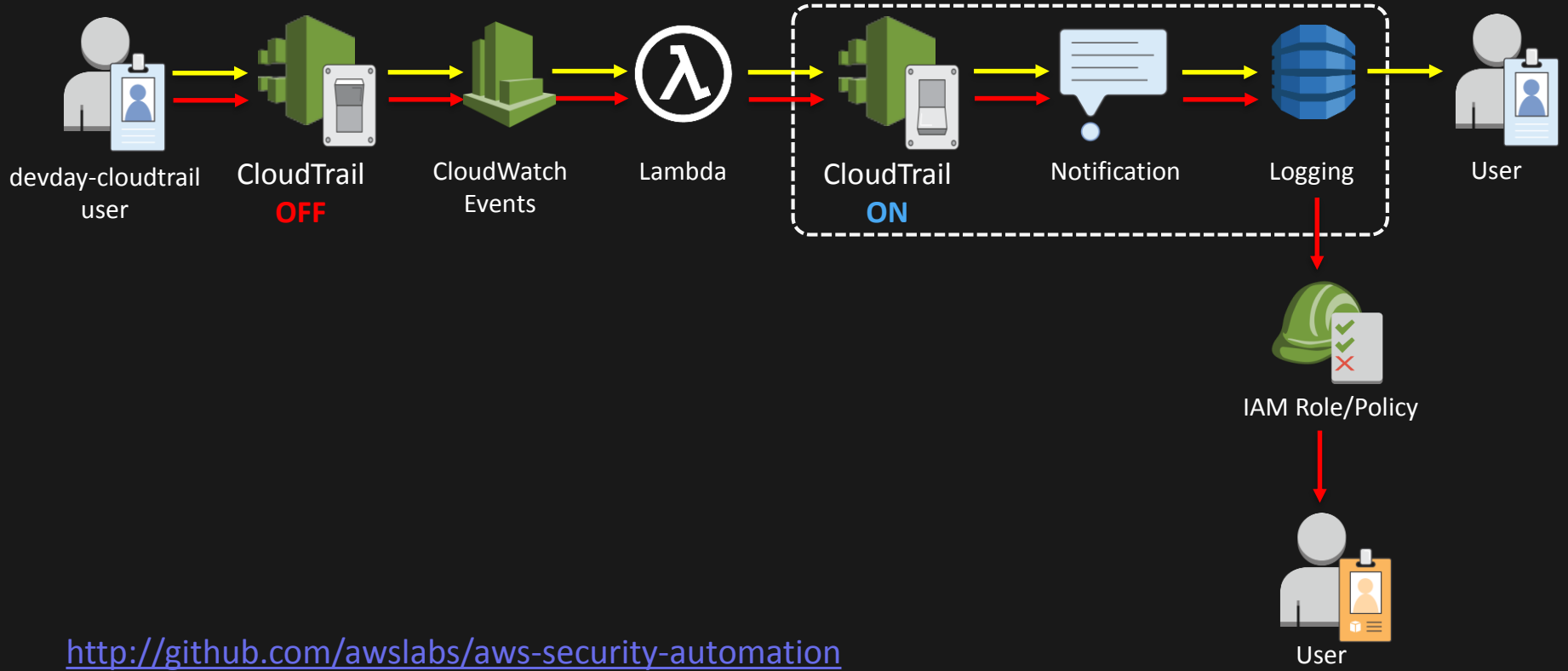
Amazon CloudWatch

- AWSサービスのリソース モニタリング
- ログモニタリング (CloudWatch Logs)
- プロアクティブ モニタリング (CloudWatch Events)

AWS Config / Config rules

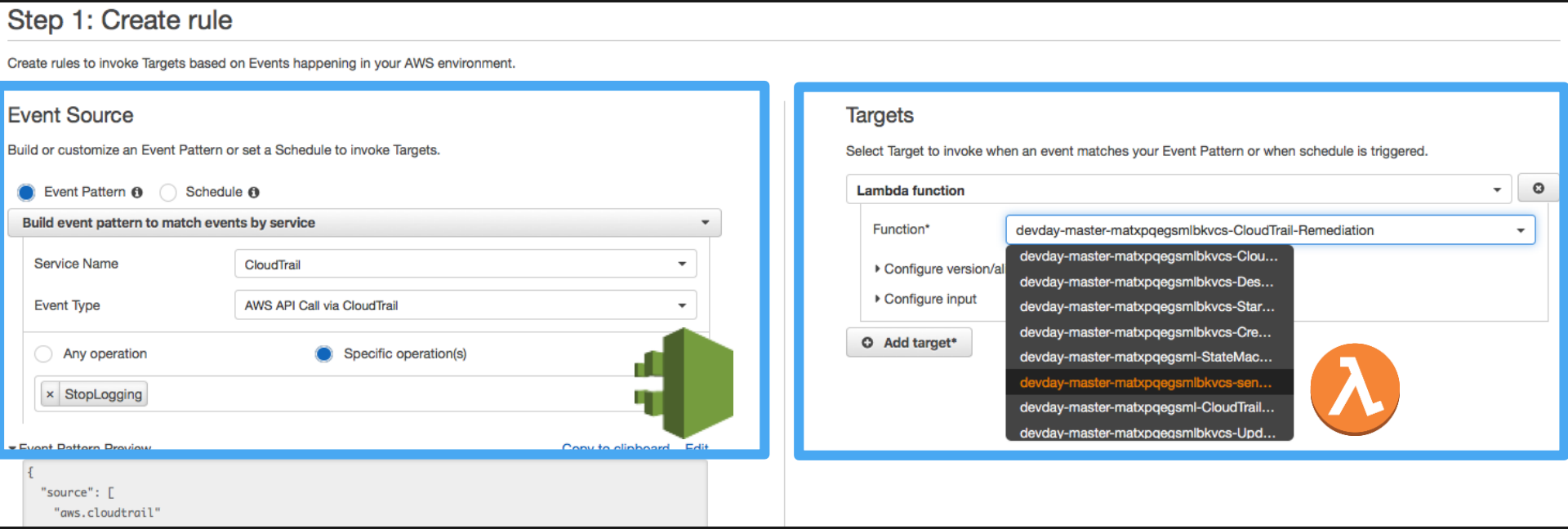
- 変更管理
- 継続的モニタリング・評価
- Compliance as Code

CloudTrail / CloudWatch Events



http://github.com/awslabs/aws-security-automation

CloudWatch Events Rule の作成



Code highlights – Lambda Functionのアクション

```
# Priority action
startTrail(trailArn)

# Alerting
result = sendAlert(logData)

# Forensics
realTable = verifyLogTable()
result = forensic(logData, realTable)

# Logging
result = logEvent(logData, realTable)
return result
```

1. CloudTrail を ON

2. 通知、アラート

3. フォレンジック

4. ロギング

Code highlights – CloudTrail Logの展開

```
trailArn = event['detail']['requestParameters']['name']
try:
    userName = event['detail']['userIdentity']['userName']
except KeyError:
    # User is federated/assumeRole
    userName = event['detail']['userIdentity']['sessionContext']['sessionIssuer']['userName']
userArn = event['detail']['userIdentity']['arn']
accessKeyId = event['detail']['userIdentity']['accessKeyId']
region = event['region']
account = event['account']
eventTime = event['detail']['eventTime']
userAgent = event['detail']['userAgent']
sourceIP = event['detail']['sourceIPAddress']
logData = {'trailArn': trailArn, 'userName': userName, 'userArn': userArn, 'accessKeyId': accessKeyId,
```

Code highlights – CloudTrail Restart !!!

```
def startTrail(trailArn):
    client = boto3.client('cloudtrail')
    response = client.get_trail_status(
        Name=trailArn
    )
    # Check if someone already started the trail
    if response['IsLogging']:
        print "Logging already started"
        return "No Action Needed"
    else:
        print "Starting trail: ", trailArn
        response = client.start_logging(
            Name=trailArn
        )
        return "Trail started"
```

Code highlights – フォレンジック

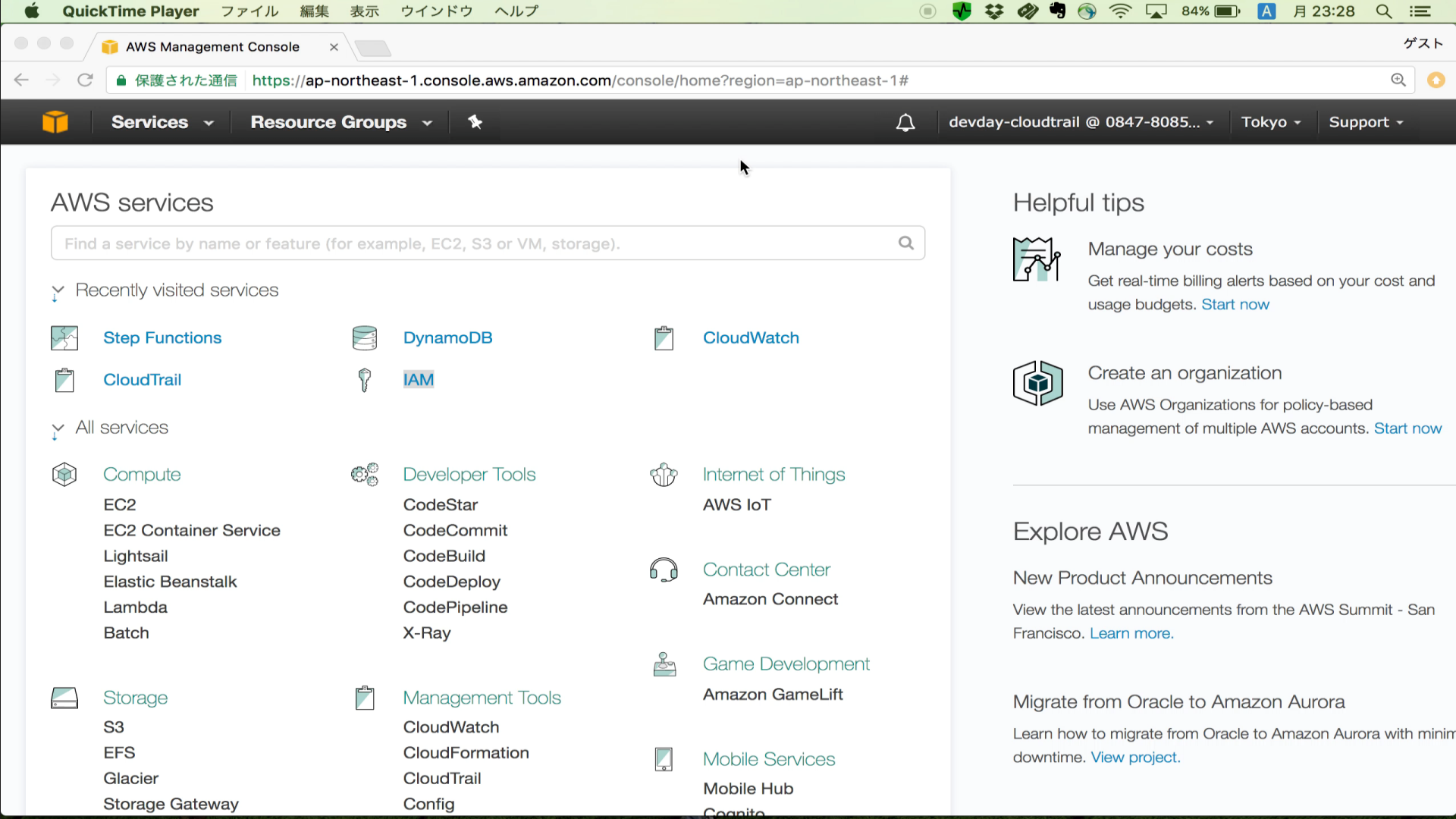
```
def forensic(logData, table):

        try:
            if response['Item']:
                # If not first time, trigger countermeasures.
                result = disableAccount(logData['userName'])
                print("Remediation")
        except Exception as e:
            # First time incident, Register it into DDB Table.
            print(e.message)
            client = boto3.client('dynamodb')
            response = client.put_item(
                TableName=table,
                    Item={
                        'userName': {'S': logData['userName']}
                    }
            )
            print("This is a firtst entry. No Remediation Needed.")
```

Code highlights – 対策アクション

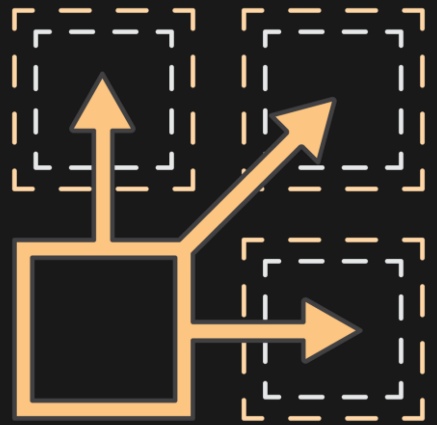
```
def disableAccount(userName):

    client = boto3.client('iam')
    response = client.put_user_policy(
        UserName=userName,
        PolicyName='BlockPolicy',
        PolicyDocument='{
            "Version":"2012-10-17",
            "Statement":{
                "Effect":"Deny",
                "Action":"*",
                "Resource":"*"
                }
        }'
    )
    return 0
```

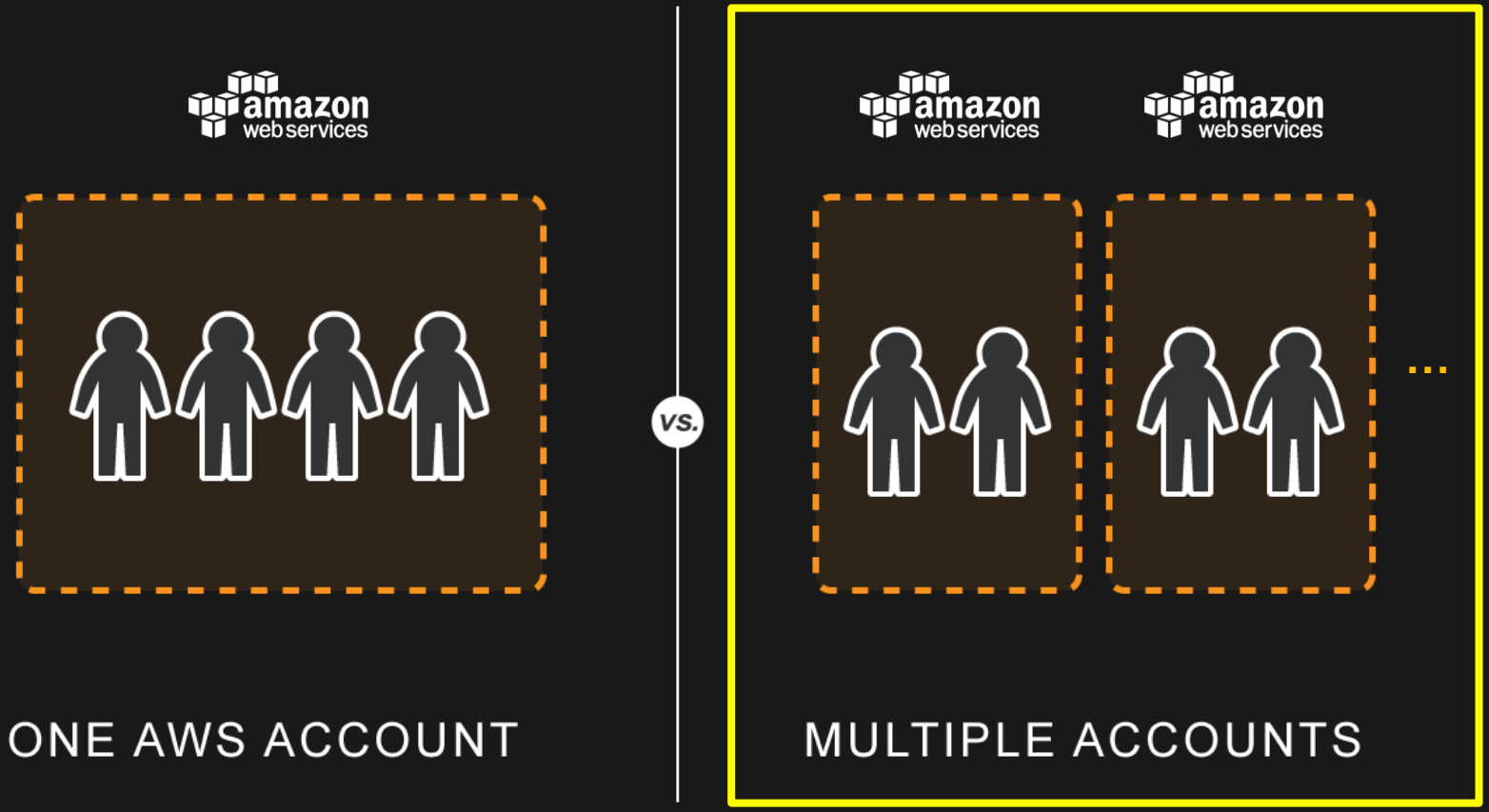
DEMO 01

Proactive Monitoring


QuickTime Player ファイル 編集 表示 ウインドウ ヘルプ
84% 月 23:28
AWS Management Console
ゲスト
保護された通信 https://ap-northeast-1.console.aws.amazon.com/console/home?region=ap-northeast-1#
Services
Resource Groups
devday-cloudtrail @ 0847-8085...
Tokyo
Support
AWS services
Find a service by name or feature (for example, EC2, S3 or VM, storage).
Recently visited services
Step Functions
DynamoDB
CloudWatch
CloudTrail
IAM
All services
Compute
EC2
EC2 Container Service
Lightsail
Elastic Beanstalk
Lambda
Batch
Developer Tools
CodeStar
CodeCommit
CodeBuild
CodeDeploy
CodePipeline
X-Ray
Internet of Things
AWS IoT
Contact Center
Amazon Connect
Game Development
Amazon GameLift
Storage
S3
EFS
Glacier
Storage Gateway
Management Tools
CloudWatch
CloudFormation
CloudTrail
Config
Mobile Services
Mobile Hub
Cognito
Helpful tips
Manage your costs
Get real-time billing alerts based on your cost and usage budgets. Start now
Create an organization
Use AWS Organizations for policy-based management of multiple AWS accounts. Start now
Explore AWS
New Product Announcements
View the latest announcements from the AWS Summit - San Francisco. Learn more.
Migrate from Oracle to Amazon Aurora
Learn how to migrate from Oracle to Amazon Aurora with minim
downtime. View project.

Security at Scale
Infrastructure as a code



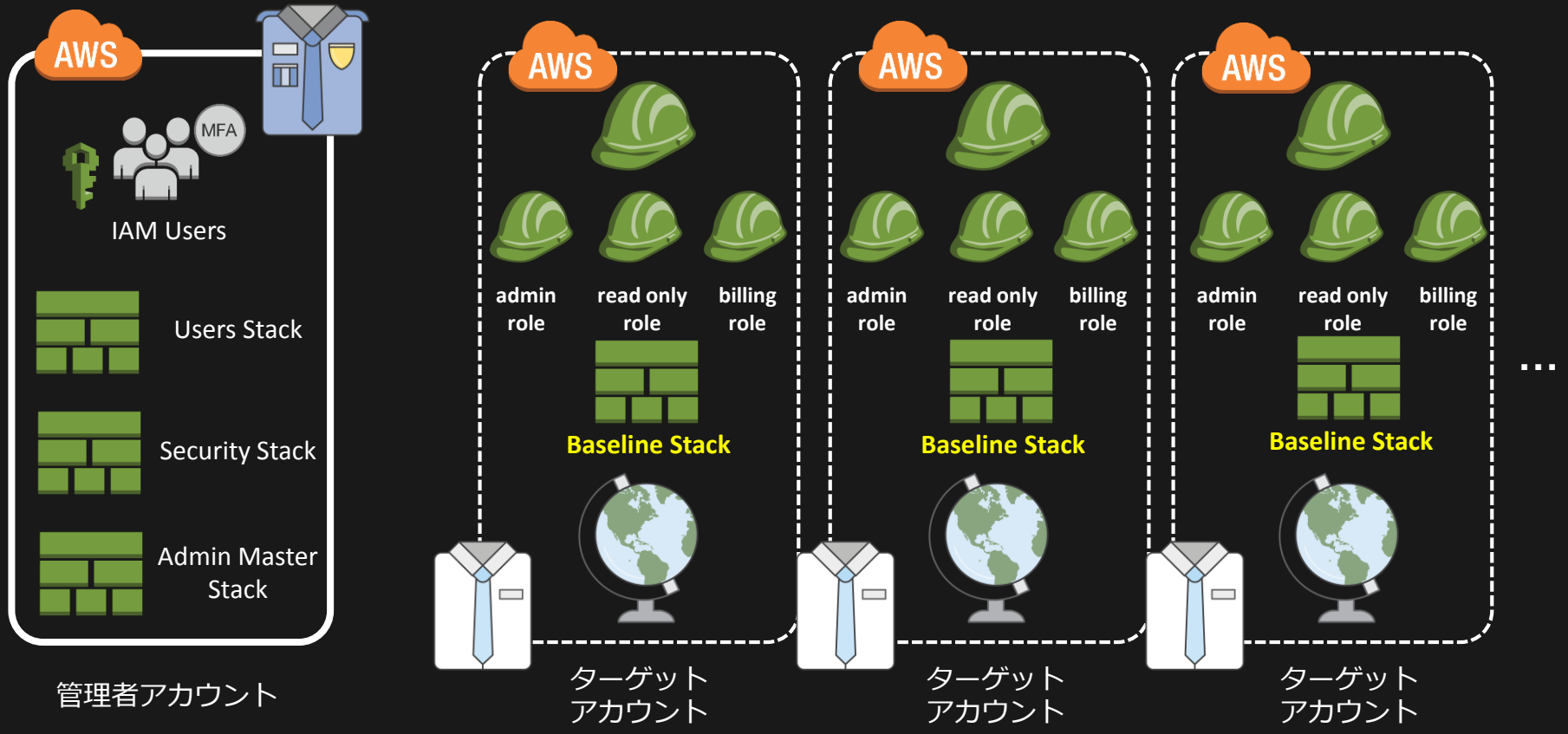

amazon web services
amazon web services
amazon web services
vs.
...
ONE AWS ACCOUNT
MULTIPLE ACCOUNTS

Multi Region

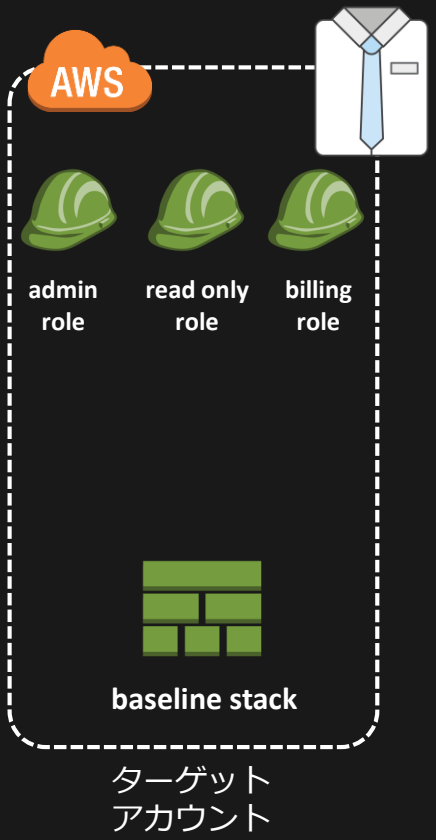
+

Multi Account

Overview of Baseline Design



Target Account Baseline Template



- IAM Role の作成
  - Assume Role に利用
- 設定サービス
  - CloudTrail (global and local)
  - Config
  - S3 Access Logging
  - CloudTrail to CloudWatch Logs
  - SNS For Slack Integration(APIGW Endpoint)

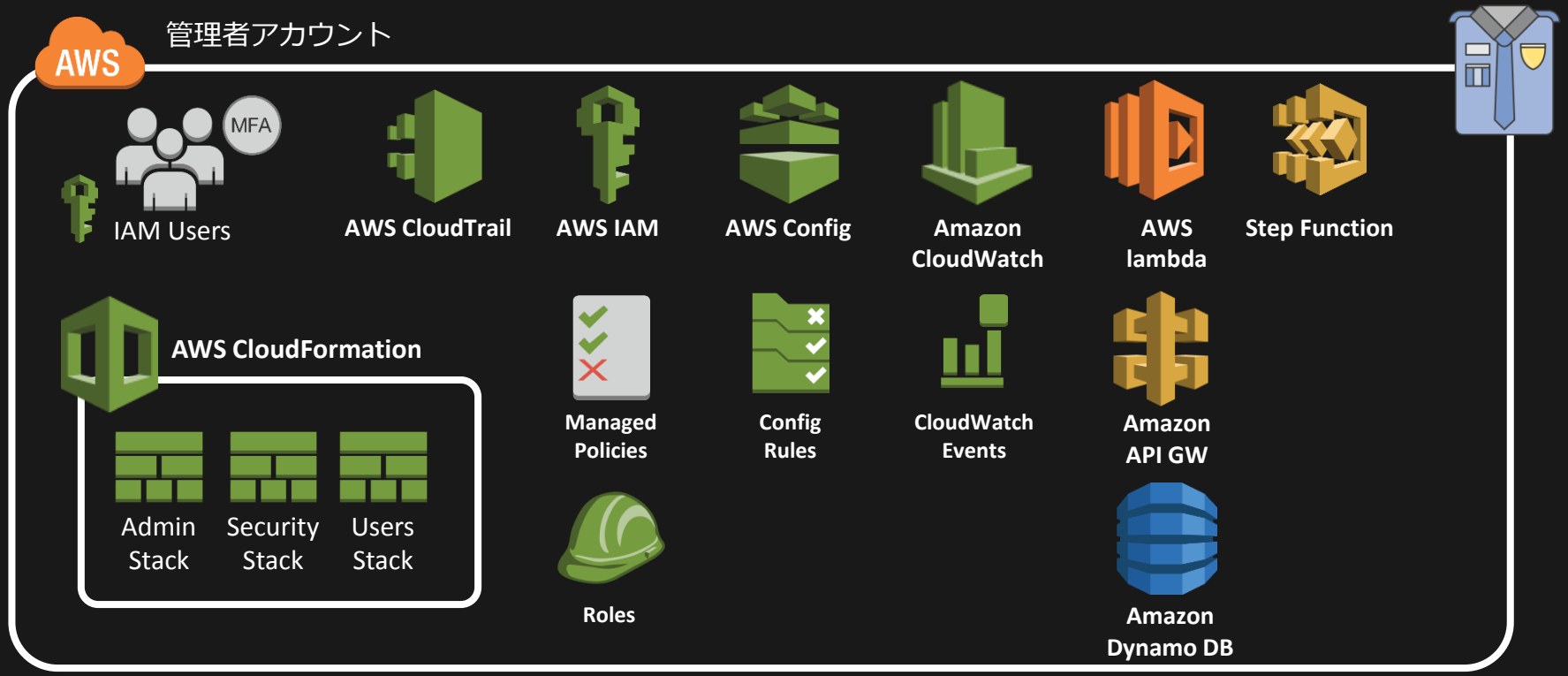
Code Highlights - IAM Role の作成

```
Conditions:
  IAMRegion: !Equals [ !Ref 'AWS::Region', 'us-east-1' ]
  SupportsCloudWatchEvents:
    'Fn::Not':
      - 'Fn::Or':
          - !Equals [ !Ref 'AWS::Region', 'ca-central-1' ]
          - !Equals [ !Ref 'AWS::Region', 'eu-west-2' ]

Resources:
  AdministratorRole:
    Type: AWS::IAM::Role
    Condition: IAMRegion
    Properties:
      RoleName: Administrator
      Path: '/dso/'
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/AdministratorAccess
        - !Ref ContainmentPolicy
      AssumeRolePolicyDocument:
        Version: 2012-10-17
        Statement:
          -
```

Code Highlights - プロアクティブ モニタリング

```
IAMActivityRule:
  Type: AWS::Events::Rule
  Condition: SupportsCloudWatchEvents
  Properties:
    Description: Notify when IAM resources are created or updated
    EventPattern:
      source: [aws.iam]
      detail-type: ['AWS API Call via CloudTrail']
      detail:
        eventSource: [iam.amazonaws.com]
        eventName:
          - CreateUser
          - CreateGroup
          - CreateRole # etc etc etc
    State: ENABLED
    Targets:
      - Arn: !Ref EventsSNSTopic
        Id: PublishToSNS
```

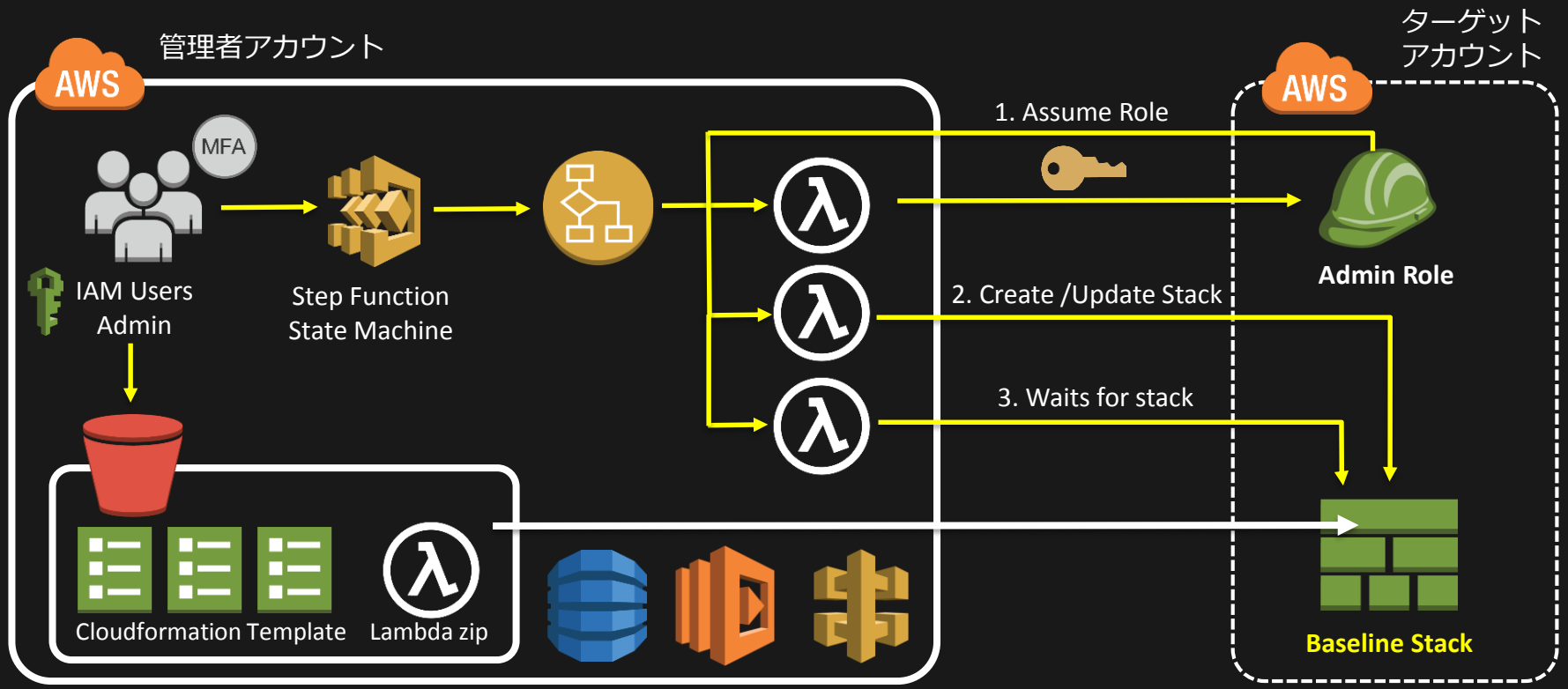
Admin Account Baseline Design



Code Highlights – Lambda Function のデプロイ

```
CreateStackFn:
  Type: AWS::Lambda::Function
  Properties:
    Handler: create_stack.lambda_handler
    Runtime: python2.7
    Role: !GetAtt CfnFnExecutionRole.Arn
    Timeout: 60
    Code:
      S3Bucket: !Ref DistBucket
      S3Key: create_stack.zip
      S3ObjectVersion: !Ref CreateStackFnVersion
    Environment:
      Variables:
        KMS_KEY_ID: !Ref KMSKeyAlias
```

Cross Account Deployment



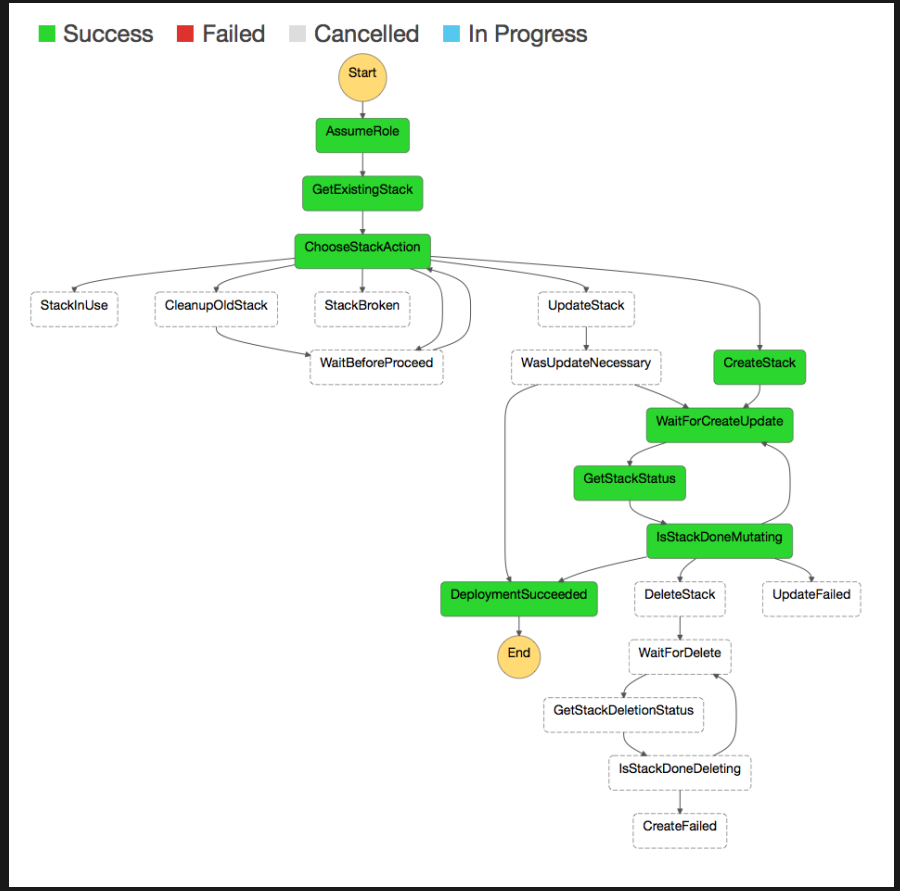
CloudFormation Template Lambda zip

```
for dir in lambda/*; do
    func_name=$(basename $dir)
    func_src=lambda/${func_name}/${func_name}.py
    func_pkg=dist/${func_name}.zip

    pyflakes $func_src

    if [ ! -f $func_pkg ] || [ $func_src -nt $func_pkg ]; then
        ...
        pushd $build_dir
        pip install -r requirements.txt -t .
        rm requirements.txt
        zip -X -r ../../$func_pkg .
        popd
    else
        echo "$func_name has not been updated, skipping"
    fi
done

for tpl_src in cloudformation/*; do…

pushd dist
$AWS s3 sync . s3://$DIST_BUCKET
popd

$AWS cloudformation validate-template \
   --template-url https://s3.amazonaws.com/$DIST_BUCKET/$TEMPLATE
```

```
resp = ec2.describe_regions()
for r in resp['Regions']:
    region = r['RegionName']
    workflow = {
        'ExecutionName': 'Deploy_{}_{}_{}_{}'.format(
            STACK_NAME, account_id, region, rand_token),
        'RoleARN': role_arn,
        'Region': region,
        'TemplateURL': posixpath.join(
            'https://s3.amazonaws.com/', DIST_BUCKET, TEMPLATE_FILE_NAME),
        'Parameters': {…
        },
        'Capabilities': ['CAPABILITY_NAMED_IAM'],
        'Stack': {
            'StackName': STACK_NAME
        }
    }
    sm_input['Workflows'].append(workflow)

resp = sfn.start_execution(
    stateMachineArn=PARALLEL_EXECUTION_STATE_MACHINE_ARN,
    name='Deploy_{}_{}_{}'.format(STACK_NAME, account_id, rand_token),
    input=json.dumps(sm_input))
```

Amazon Step Function Cross Region

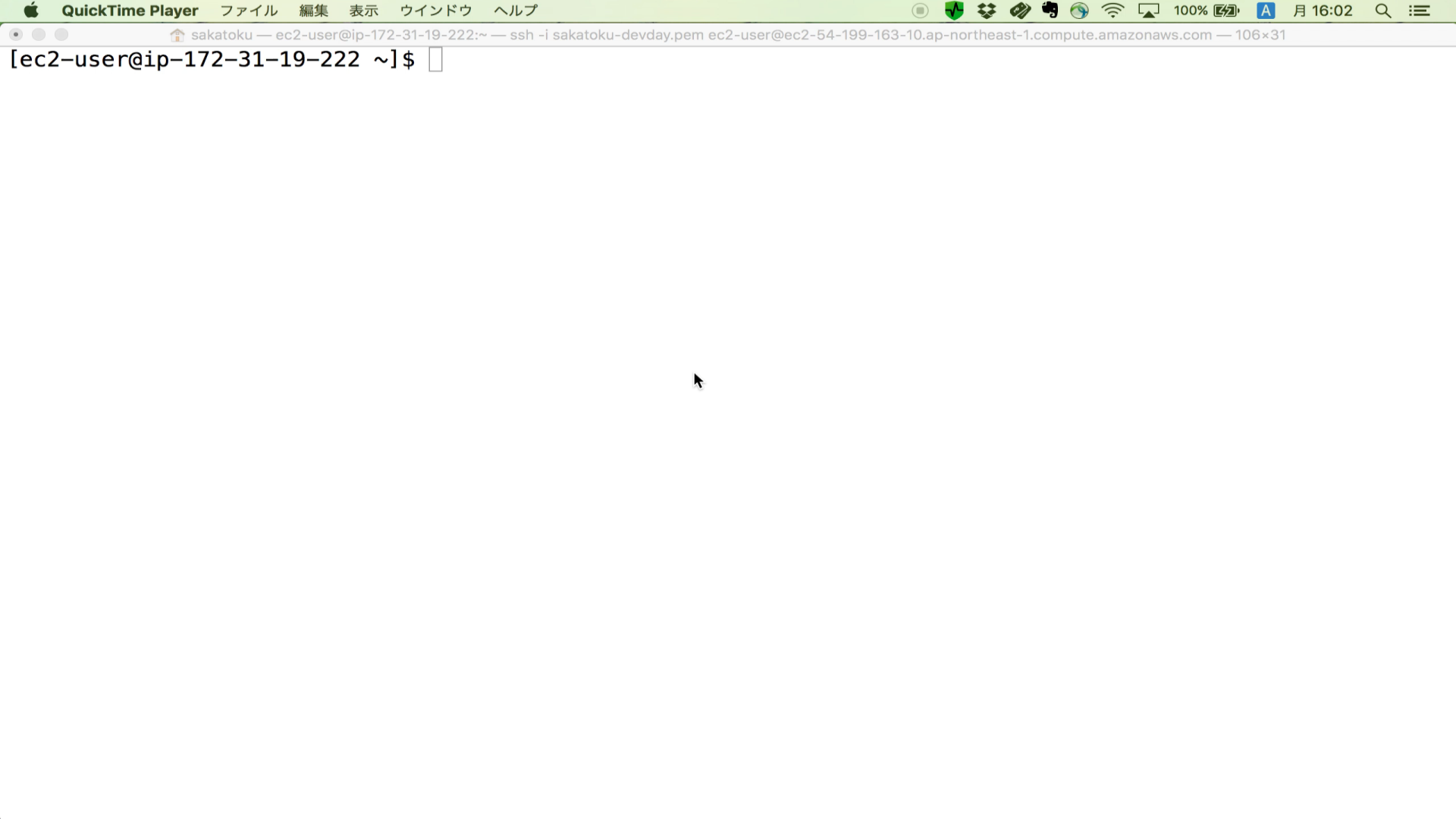
Amazon Step Function
- State Machine -




```
{
   "States":{
      "WaitForWorkflows":{
         "Seconds":30,
         "Type":"Wait",
         "Next":"GetWorkflowStatuses"
      },
      "StartWorkflows":{
         "Resource":"arn:aws:lambda:ap-northeast-1:084780857413:function:
         "ResultPath":"$.Workflows",
         "Type":"Task",
         "Next":"WaitForWorkflows"
      },
      "DeploymentSucceeded":{
         "End":true,
         "Type":"Pass"
      },
      "HaveWorkflowsCompleted":{
         "Default":"WaitForWorkflows",
         "Type":"Choice",
         "Choices":[
            {
               "Variable":"$.Status",
               "StringEquals":"SUCCEEDED",
               "Next":"DeploymentSucceeded"
            },
            {
               "Variable":"$.Status",
               "StringEquals":"FAILED",
               "Next":"DeploymentFailed"
            }
         ]
      },
      "DeploymentFailed":{
         "Cause":"The deployment failed",
         "Type":"Fail",
         "Error":"DeploymentFailed"
```

DEMO 02

Security CI/CD


QuickTime Player ファイル 編集 表示 ウインドウ ヘルプ
100% 月 16:02
sakatoku — ec2-user@ip-172-31-19-222:~ — ssh -i sakatoku-devday.pem ec2-user@ec2-54-199-163-10.ap-northeast-1.compute.amazonaws.com — 106×31
[ec2-user@ip-172-31-19-222 ~]$

まとめ

- プラットフォームがAWSになることで、今までとは全く異なるSecurity Automationが実現可能

- Security Automationを広範囲に実現することで、DevSecOpsを回すことが可能になる

- Security Baseline = Dev + Ops +Sec

Happy Coding ☺

- get-secrets

https://github.com/awslabs/git-secrets

- CloudTrail Remediation

http://github.com/awslabs/aws-security-automation

Happy Policy Coding ☺

Don’t Forget Evaluations!

How to Centrally Manage AWS Config Rules across Multiple AWS Accounts

https://aws.amazon.com/blogs/devops/how-to-centrally-manage-aws-config-rules-across-multiple-aws-accounts/

Keep your TAG

- IAM Role – Tag restriction
- https://aws.amazon.com/jp/blogs/aws/new-tag-ec2-instances-ebs-volumes-on-creation/
- http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ExamplePolicies_EC2.html#iam-example-instances

- https://aws.amazon.com/jp/answers/account-management/aws-tagging-strategies/

本セッションのFeedbackをお願いします

受付でお配りしたアンケートに本セッションの満足度やご感想などをご記入ください
アンケートをご提出いただきました方には、もれなく素敵なAWSオリジナルグッズをプレゼントさせていただきます

アンケートは各会場出口、パミール3FのEXPO展示会場内にて回収させて頂きます