aws DEV DAY
ONLINE JAPAN

C - 4

「そのコンテナ、安全ですか？」

～AWS x DevSecOpsで実践するコンテナセキュリティ～

新井 雅也 - Masaya ARAI
株式会社野村総合研究所 – Nomura Research Institute, Ltd.

In Partnership with intel

新井 雅也 – Masaya ARAI

野村総合研究所 - NRI

フルスタックディベロッパー

I ♡AWS, GCP, k8s, Golang, IaC and UI/UX

2020 APN AWS Top Engineers

@msy78

JAWS-UG コンテナ支部
(コミュニティ運営)

In Partnership with intel

Agenda

なぜ DevSecOps が求められるのか？

DevSecOps を実現するためのプラクティスとツール

AWS × DevSecOps の歩み寄り方

よりセキュアなコンテナ運用を目指して

まとめ

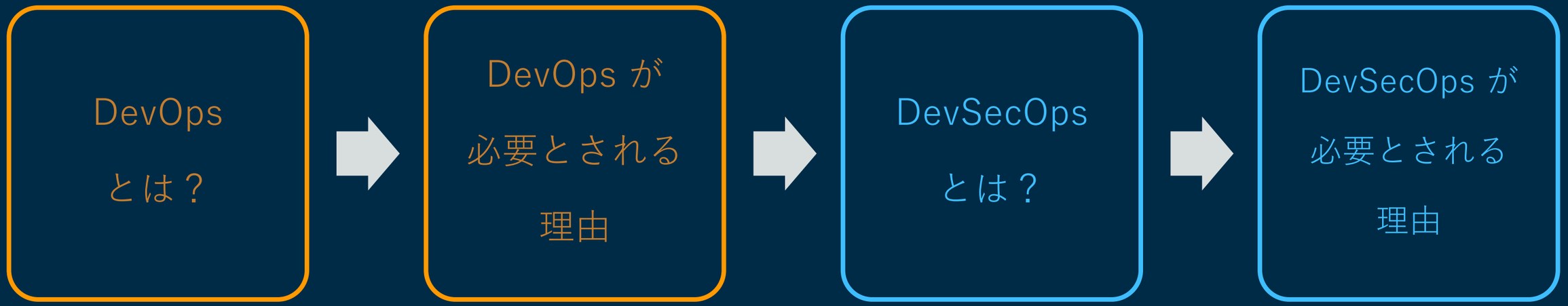
In Partnership with intel

aws DEV DAY
ONLINE JAPAN

なぜ DevSecOps が求められるのか？

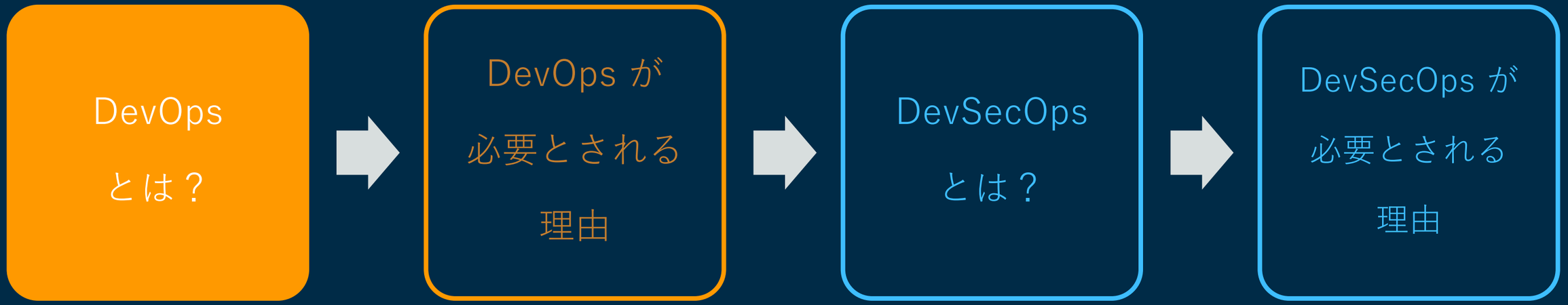
In Partnership with intel

DevSecOps の必要性をお話する前に、前提を整理していく



In Partnership with intel

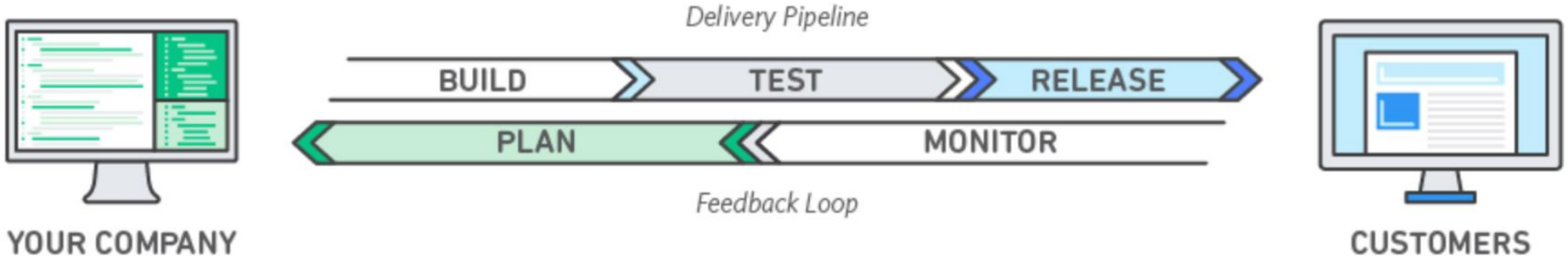
DevSecOps の必要性をお話する前に、前提を整理していく



In Partnership with intel

DevOps は開発チームと運用チームが協力してビジネス価値を高める活動の概念

厳密な定義は存在しないが、DevOps は**組織の文化、プラクティス、ツール**で構成



※図の一部を引用
https://aws.amazon.com/jp/devops/what-is-devops/

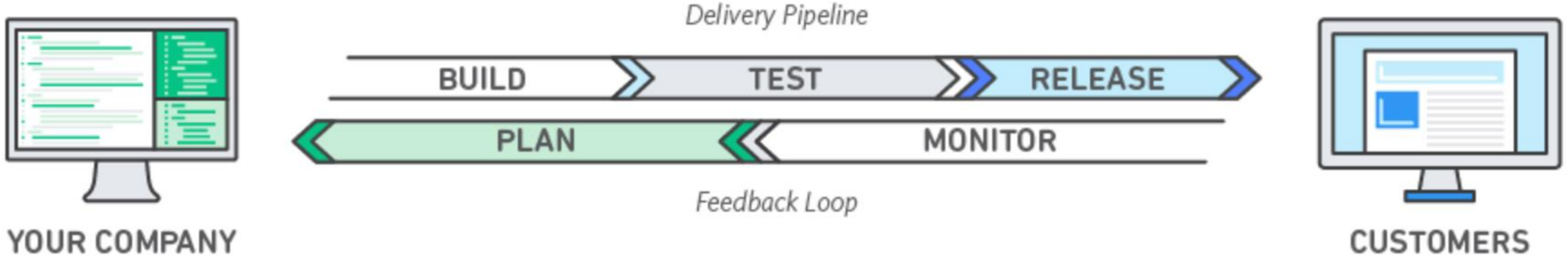
DevOps は開発チームと運用チームが協力してビジネス価値を高める活動の概念

厳密な定義は存在しないが、DevOps は**組織の文化、プラクティス、ツール**で構成

NOT 縦割り

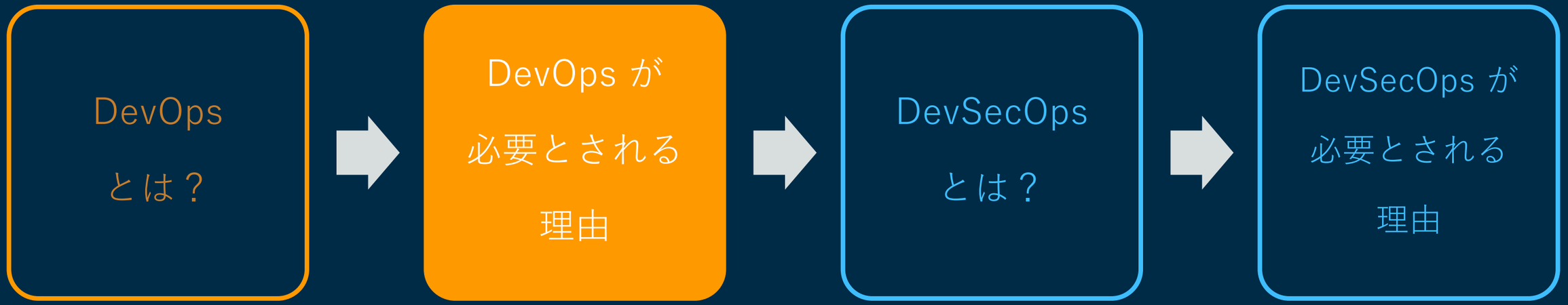
アジャイル,
CI/CD 等

Git, コンテナ,etc



※図の一部を引用
https://aws.amazon.com/jp/devops/what-is-devops/

DevSecOps の必要性をお話する前に、前提を整理していく



In Partnership with intel

変化に対して柔軟かつ品質の良いサービスを提供するために DevOps は必要

IT は効率化の手段だけではなく差別化の手段(= DX)

金融機関、
Fintech 企業など

※登壇者は主に金融業界のお客様を担当しており、
ここでは Fintech サービスを例にご紹介しています。

In Partnership with intel

変化に対して柔軟かつ品質の良いサービスを提供するために DevOps は必要

IT は効率化の手段だけではなく差別化の手段(= DX)

金融機関、
Fintech 企業など

開発/提供

Fintech サービス

※登壇者は主に金融業界のお客様を担当しており、
ここでは Fintech サービスを例にご紹介しています。

In Partnership with intel

変化に対して柔軟かつ品質の良いサービスを提供するために DevOps は必要

IT は効率化の手段だけではなく差別化の手段(= DX)

※登壇者は主に金融業界のお客様を担当しており、
ここでは Fintech サービスを例にご紹介しています。

In Partnership with intel

変化に対して柔軟かつ品質の良いサービスを提供するために DevOps は必要

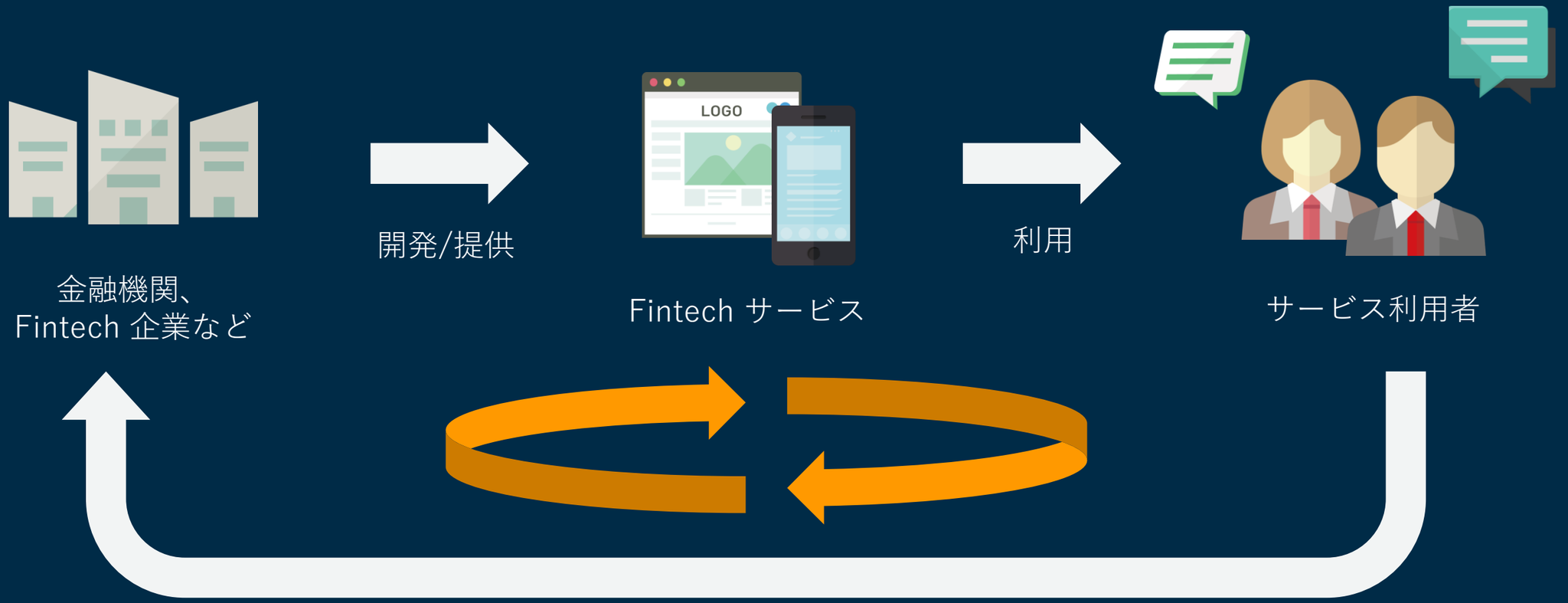
IT は効率化の手段だけではなく差別化の手段(= DX)

※登壇者は主に金融業界のお客様を担当しており、
ここでは Fintech サービスを例にご紹介しています。

In Partnership with intel

変化に対して柔軟かつ品質の良いサービスを提供するために DevOps は必要

IT は効率化の手段だけではなく差別化の手段(= DX)



※登壇者は主に金融業界のお客様を担当しており、
ここでは Fintech サービスを例にご紹介しています。

In Partnership with intel

変化に対して柔軟かつ品質の良いサービスを提供するために DevOps は必要

IT は効率化の手段だけではなく差別化の手段(= DX)

※登壇者は主に金融業界のお客様を担当しており、
ここでは Fintech サービスを例にご紹介しています。

In Partnership with intel

変化に対して柔軟かつ品質の良いサービスを提供するために DevOps は必要

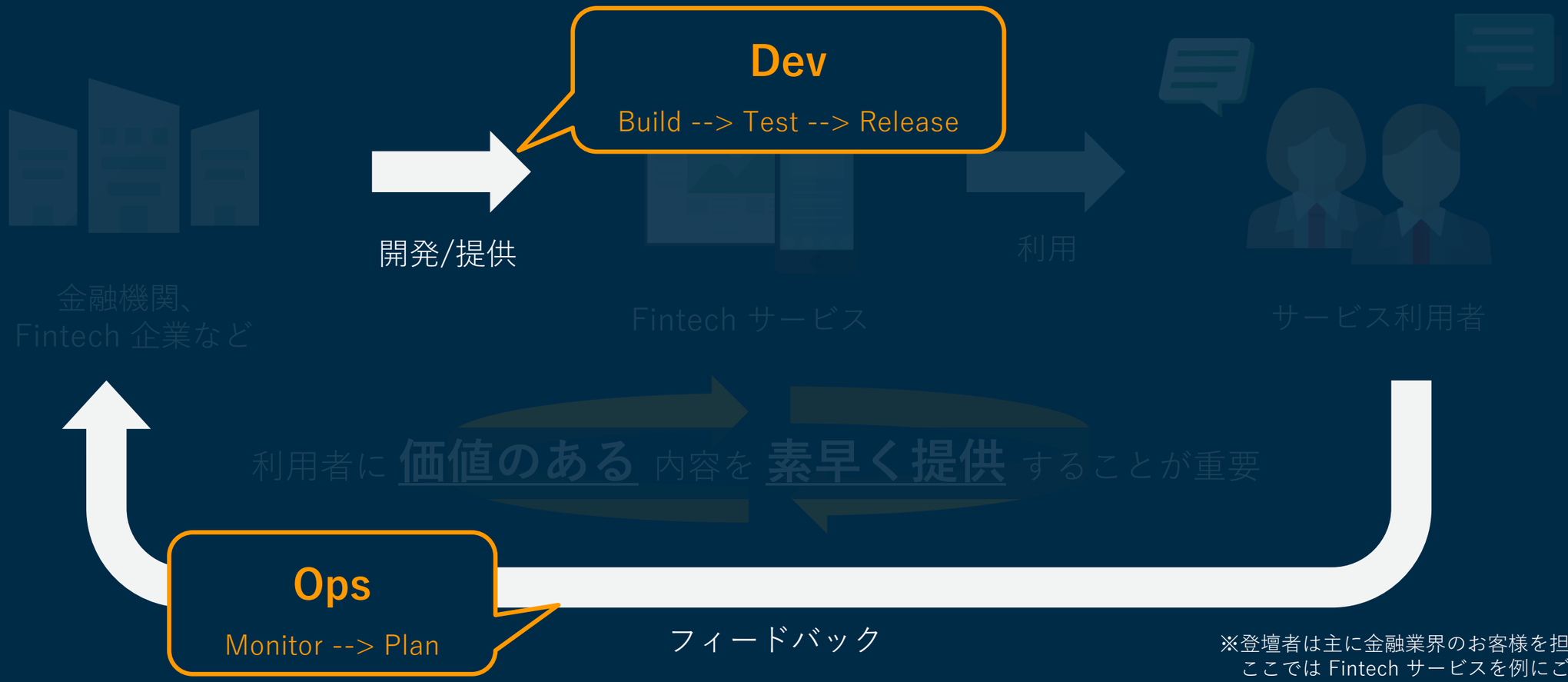
IT は効率化の手段だけではなく差別化の手段(= DX)

※登壇者は主に金融業界のお客様を担当しており、
ここでは Fintech サービスを例にご紹介しています。

In Partnership with intel

変化に対して柔軟かつ品質の良いサービスを提供するために DevOps は必要

IT は効率化の手段だけではなく差別化の手段(= DX)



※登壇者は主に金融業界のお客様を担当しており、
ここでは Fintech サービスを例にご紹介しています。

変化に対して柔軟かつ品質の良いサービスを提供するために DevOps は必要

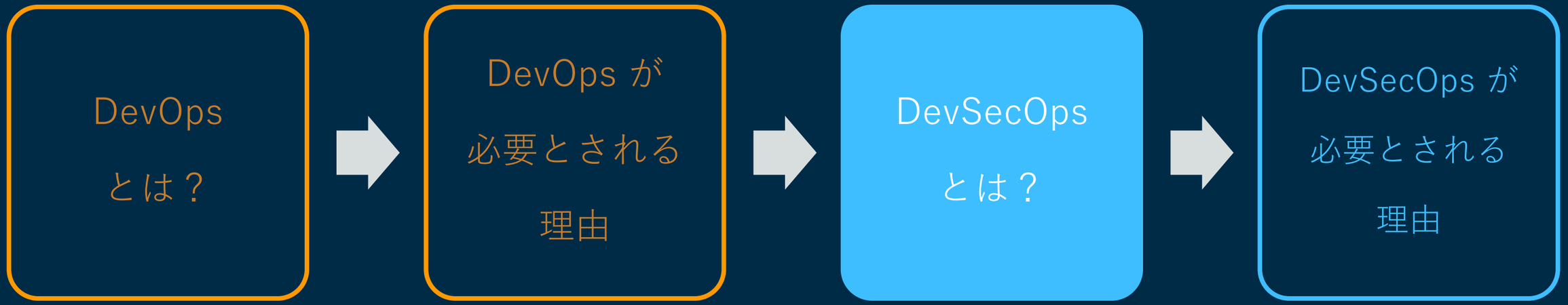
IT は効率化の手段だけではなく差別化の手段(= DX)

DevOps は DX のアジリティを高めるための重要な要素

※登壇者は主に金融業界のお客様を担当しており、
ここでは Fintech サービスを例にご紹介しています。

In Partnership with intel

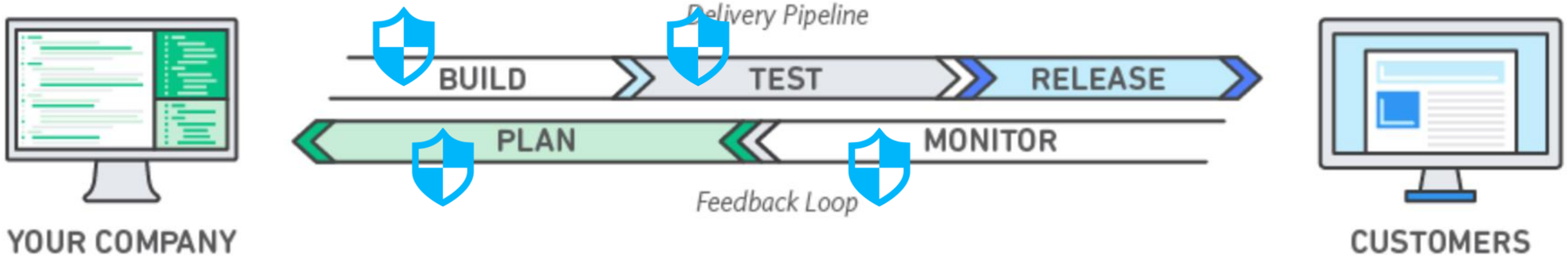
DevSecOps の必要性をお話する前に、前提を整理していく



In Partnership with intel

DevOps にセキュリティを協調させるのが DevSecOps

DevOps と同様に組織の文化、プラクティス、ツールをセキュリティの目線で取り入れる



※図の一部を引用
https://aws.amazon.com/jp/devops/what-is-devops/

In Partnership with intel

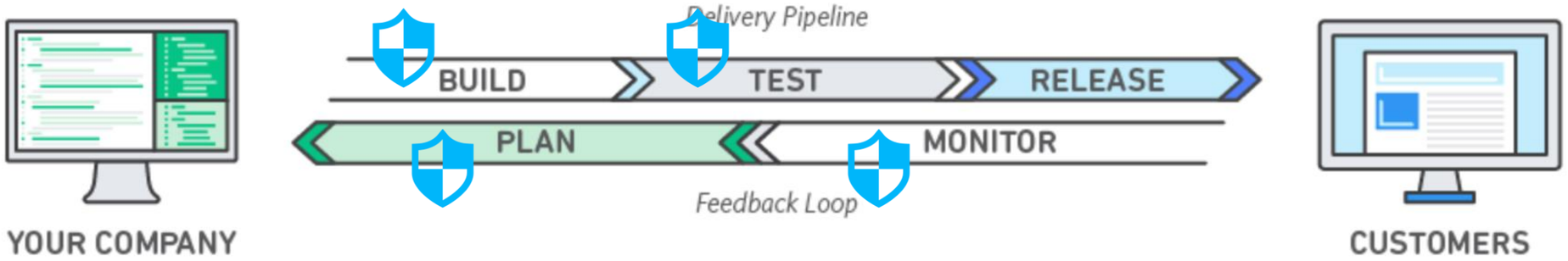
DevOps にセキュリティを協調させるのが DevSecOps

DevOps と同様に組織の文化、プラクティス、ツールをセキュリティの目線で取り入れる

全員がセキュリティに責務を持つ

シフトレフト、CI/CD 等

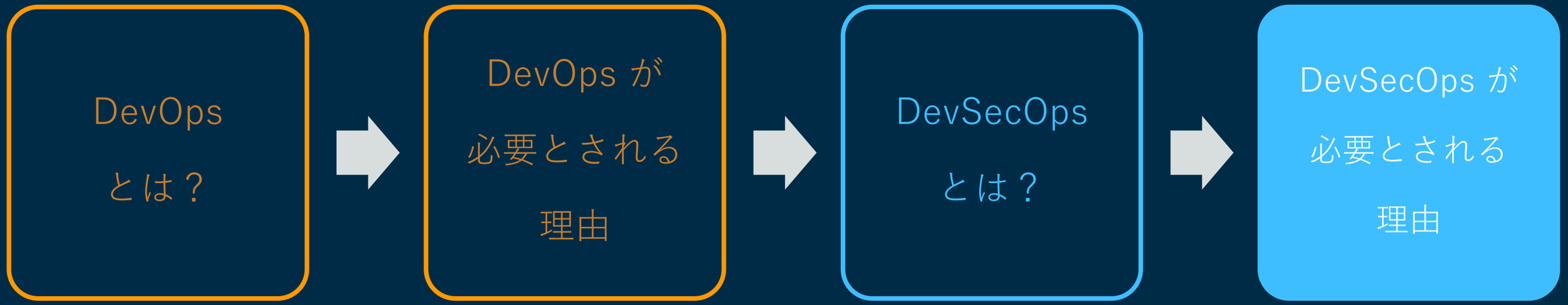
OSS 等



※図の一部を引用

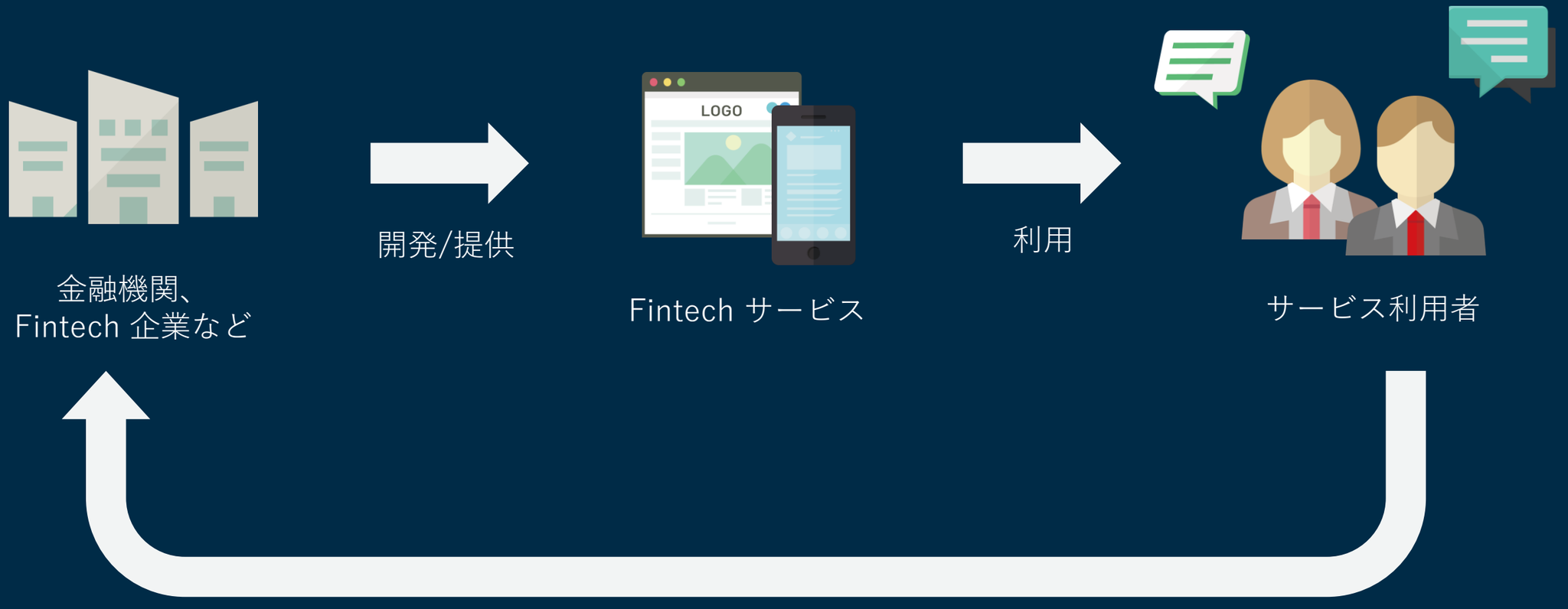
https://aws.amazon.com/jp/devops/what-is-devops/

DevSecOps の必要性をお話する前に、前提を整理していく



In Partnership with intel

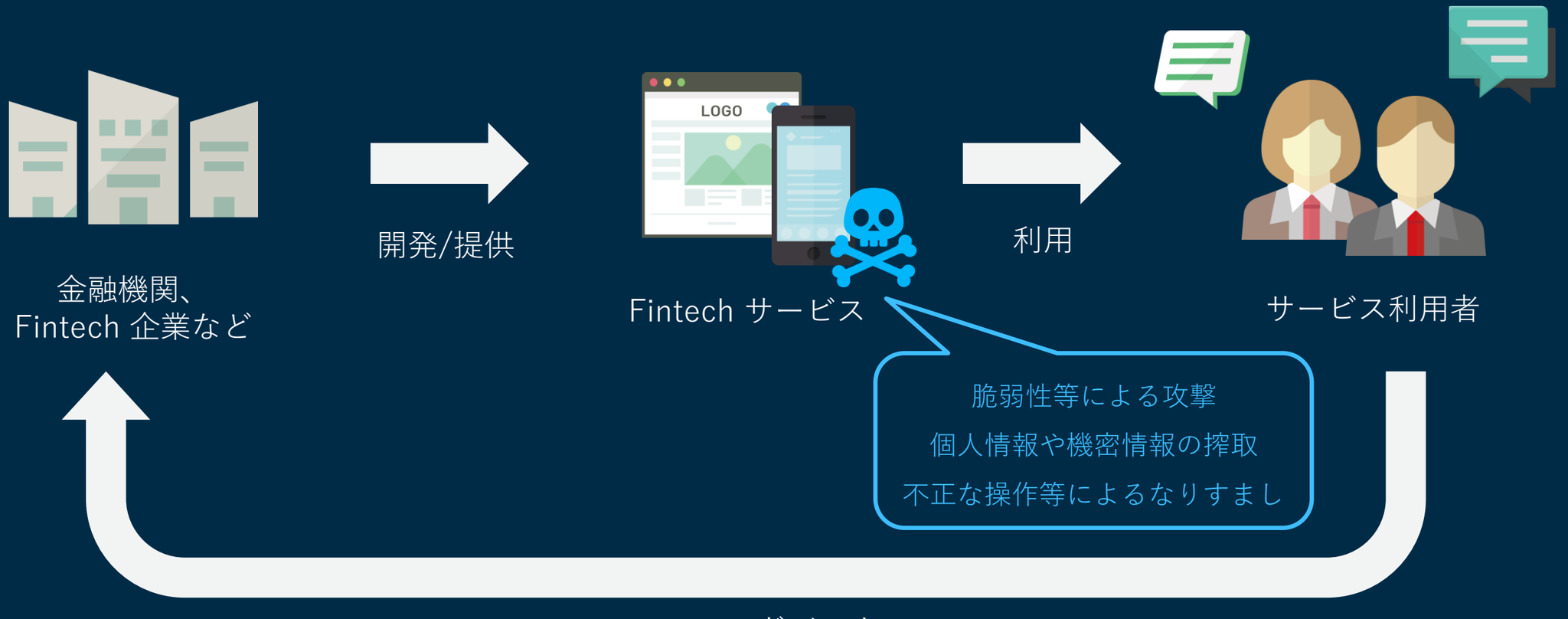
セキュリティインシデントはビジネスの根幹を揺るがすことになりかねない



※登壇者は主に金融業界のお客様を担当しており、
ここでは Fintech サービスを例にご紹介しています。

In Partnership with intel

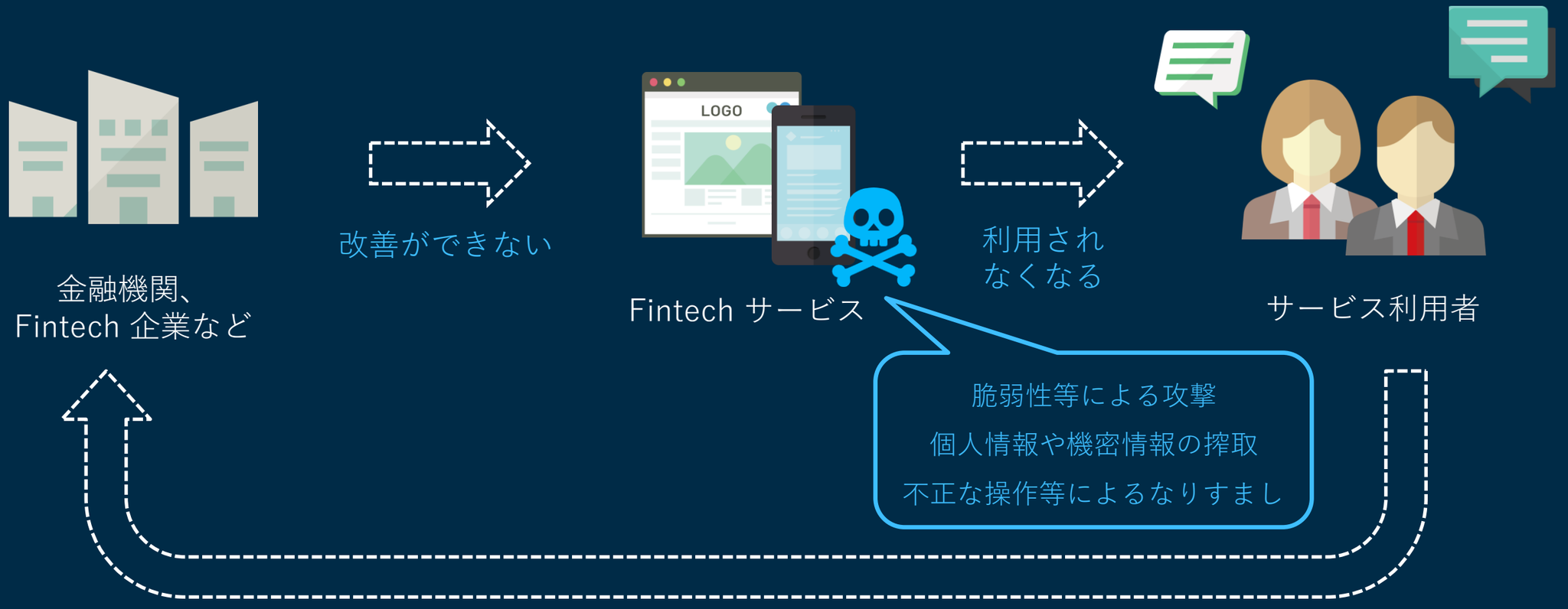
セキュリティインシデントはビジネスの根幹を揺るがすことになりかねない



※登壇者は主に金融業界のお客様を担当しており、
ここでは Fintech サービスを例にご紹介しています。

In Partnership with intel

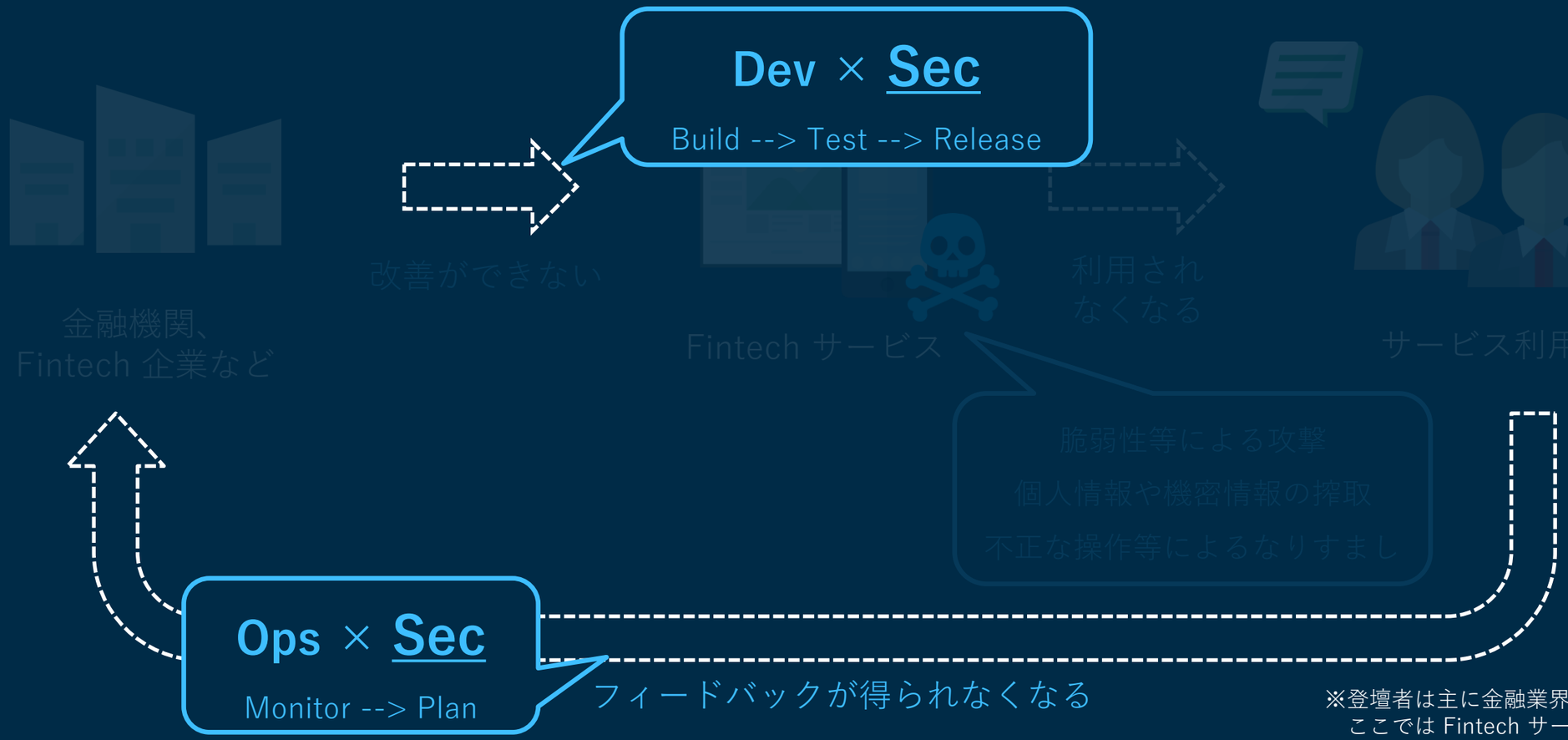
セキュリティインシデントはビジネスの根幹を揺るがすことになりかねない



※登壇者は主に金融業界のお客様を担当しており、
ここでは Fintech サービスを例にご紹介しています。

In Partnership with intel

セキュリティインシデントはビジネスの根幹を揺るがすことになりかねない



※登壇者は主に金融業界のお客様を担当しており、
ここでは Fintech サービスを例にご紹介しています。

In Partnership with intel

セキュリティインシデントはビジネスの根幹を揺るがすことになりかねない

Dev × Sec

Build --> Test --> Release

スピードと品質が優れていれば良いだけでなく、
セキュリティとの両立(= DevSecOps)は DX 実現の大前提

Ops × Sec

Monitor --> Plan

※登壇者は主に金融業界のお客様を担当しており、
ここでは Fintech サービスを例にご紹介しています。

In Partnership with intel

aws DEV DAY
ONLINE JAPAN

DevSecOps を実現するためのプラクティスとツール

In Partnership with intel

DevOps にセキュリティを協調させるのが DevSecOps

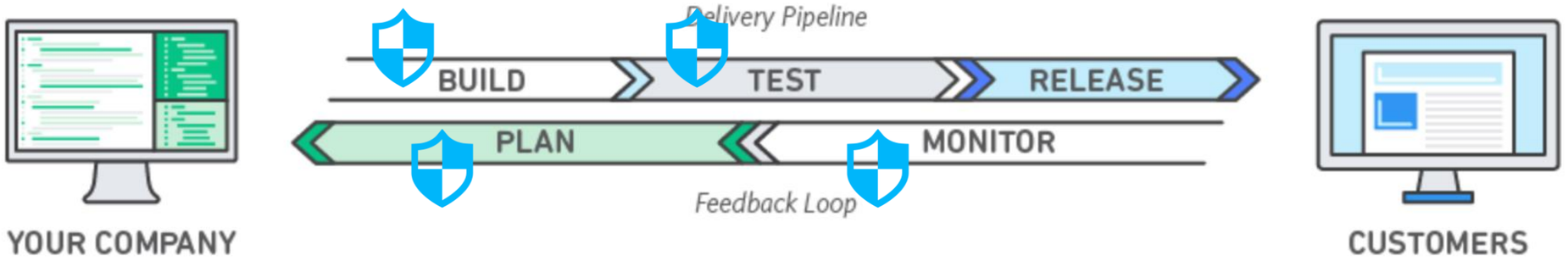
再掲

DevOps と同様に組織の文化、プラクティス、ツールをセキュリティの目線で取り入れる

全員がセキュリティに責務を持つ

シフトレフト、CI/CD 等

OSS 等



※図の一部を引用

https://aws.amazon.com/jp/devops/what-is-devops/

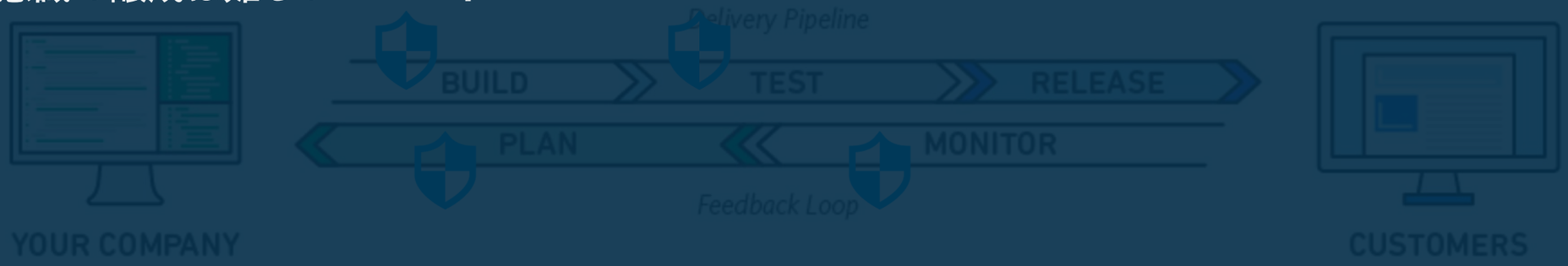
DevOps にセキュリティを協調させるのが DevSecOps

再掲

DevOps と同様に組織の文化、プラクティス、ツールをセキュリティの目線で取り入れる

- 全員がセキュリティに責務を持つ
- シフトレフト、CI/CD 等
- OSS 等

いきなり意識の醸成は難しい・・・？



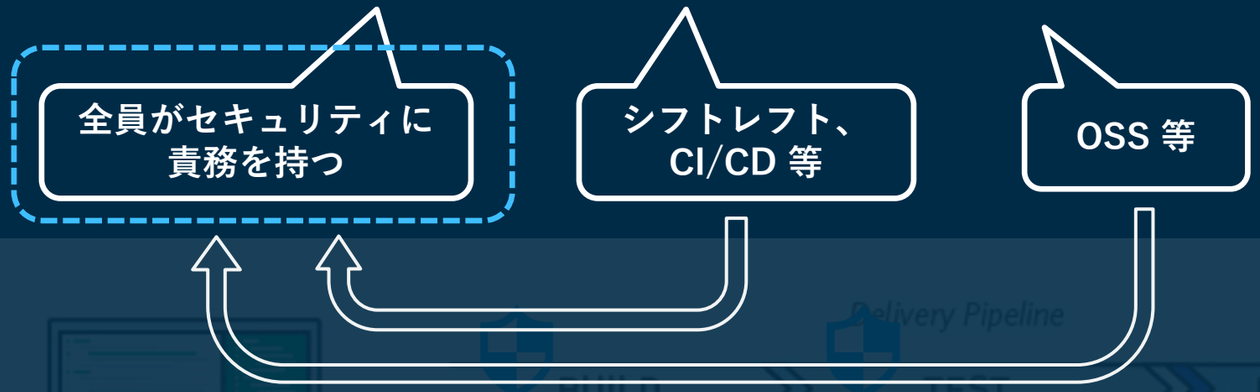
※図の一部を引用
https://aws.amazon.com/jp/devops/what-is-devops/

In Partnership with intel

再掲

DevOps にセキュリティを協調させるのが DevSecOps

DevOps と同様に組織の文化、プラクティス、ツールをセキュリティの目線で取り入れる



プラクティスとツールから

徐々に文化を醸成させていくアプローチが理想

※図の一部を引用

https://aws.amazon.com/jp/devops/what-is-devops/

DevOps にセキュリティを協調させるのが DevSecOps

再掲

DevOps と同様に組織の文化、プラクティス、ツールをセキュリティの目線で取り入れる

コンテナ技術を軸としてセキュリティリスクと対策を検討

https://aws.amazon.com/jp/devops/what-is-devops/

NIST SP800-190 からコンテナイメージに含まれるリスクを俯瞰してみる

アメリカ国立標準技術研究所（NIST）が提供するコンテナのリスクと対策内容をまとめたもの

項番	項目	リスク内容	対策
1.1	イメージの脆弱性	イメージ内コンポーネントのセキュリティアップデート漏れ等により、脆弱性が含まれている	脆弱性を排除する、かつ脆弱性を含んだイメージをデプロイしないようにする
1.2	イメージの設定不備	イメージの設定が不適切である 例) root ユーザによる実行	ベストプラクティスに従ったイメージ作成を行う
1.3	埋め込まれたマルウェア	悪意のあるファイルがイメージに含まれている	信頼されたイメージやレジストリを利用する
1.4	埋め込まれた平文の秘密情報	イメージファイル内に予め秘密情報を平文で含んでしまうことで、セキュリティリスクが生じる	秘密情報をイメージの外部で保存する
1.5	信頼できないイメージの使用	外部の信頼できないイメージを利用してしまう	信頼されたイメージやレジストリを利用する

※コンテナイメージに関する内容を一部引用

https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-190.pdf

NIST SP800-190 からコンテナイメージに含まれるリスクを俯瞰してみる

アメリカ国立標準技術研究所（NIST）が提供するコンテナのリスクと対策内容をまとめたもの

項番	項目	リスク内容	対策
1.1	イメージの脆弱性	イメージ内コンポーネントのセキュリティアップデート漏れ等により、脆弱性が含まれている	脆弱性を排除する、かつ脆弱性を含んだイメージをデプロイしないようにする
1.2	イメージの設定不備	イメージの設定が不適切である 例) root ユーザによる実行	ベストプラクティスに従ったイメージ作成を行う
1.3	埋め込まれたマルウェア	悪意のあるファイルがイメージに含まれている	信頼されたイメージやレジストリを利用する
1.4	埋め込まれた平文の秘密情報	イメージファイル内に予め秘密情報を平文で含んでしまうことで、セキュリティリスクが生じる	秘密情報をイメージの外部で保存する
1.5	信頼できないイメージの使用	外部の信頼できないイメージを利用してしまう	信頼されたイメージやレジストリを利用する

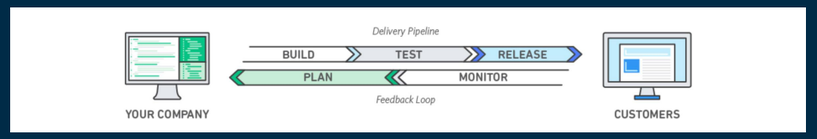
今回対処を目指すリスク

1.3〜1.5 は後述のトピックにて。

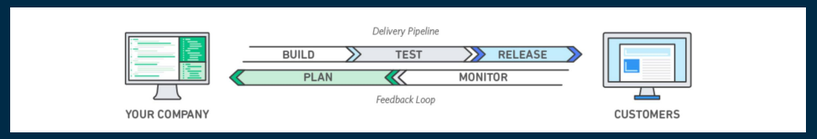
※コンテナイメージに関する内容を一部引用

https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-190.pdf

コンテナ視点で必要なセキュリティ対策を俯瞰してみる

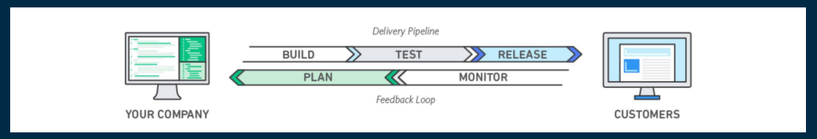


コンテナ視点で必要なセキュリティ対策を俯瞰してみる



In Partnership with intel

コンテナ視点で必要なセキュリティ対策を俯瞰してみる



コンテナ視点で。

▼静的解析 (SAST)

ベストプラクティスチェック

CIS ベンチマーク (コンテナイメージ)

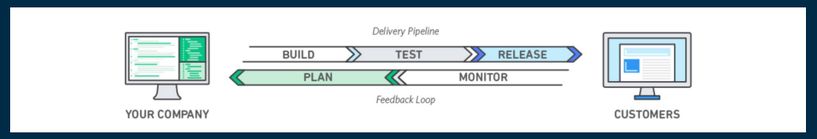
※商用ツールを含めるとさらに細分化されるが、
ここでは基本的な内容のみ列挙している。

※上記は以下内容を参考に登壇者が独自に追記したもの。

https://cloudsecurityalliance.org/artifacts/devsecops-automation/

In Partnership with intel

コンテナ視点で必要なセキュリティ対策を俯瞰してみる



コンテナ視点で。

▼静的解析 (SAST)

ベストプラクティスチェック

CIS ベンチマーク (コンテナイメージ)

▼ ソフトウェアコンポジション解析 (SCA)

CVE 脆弱性チェック / イメージスキャン

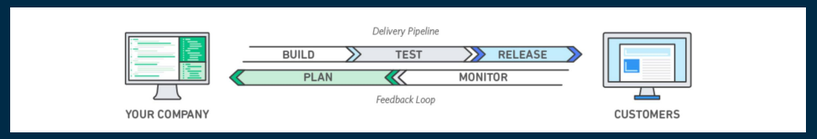
※商用ツールを含めるとさらに細分化されるが、
ここでは基本的な内容のみ列挙している。

※上記は以下内容を参考に登壇者が独自に追記したもの。

https://cloudsecurityalliance.org/artifacts/devsecops-automation/

In Partnership with intel

コンテナ視点で必要なセキュリティ対策を俯瞰してみる



コンテナ視点で。

▼静的解析 (SAST)

ベストプラクティスチェック

CIS ベンチマーク (コンテナイメージ)

▼ ソフトウェアコンポジション解析 (SCA)

CVE 脆弱性チェック / イメージスキャン

▼動的解析 (DAST)

ペネトレーションテスト

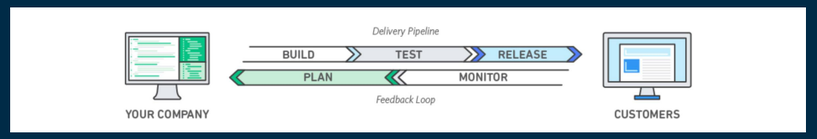
※商用ツールを含めるとさらに細分化されるが、
ここでは基本的な内容のみ列挙している。

※上記は以下内容を参考に登壇者が独自に追記したもの。

https://cloudsecurityalliance.org/artifacts/devsecops-automation/

In Partnership with intel

コンテナ視点で必要なセキュリティ対策を俯瞰してみる



Build Test Release Monitor Plan

コンテナ視点で。

▼静的解析 (SAST)

ベストプラクティスチェック

CIS ベンチマーク (コンテナイメージ)

▼ ソフトウェアコンポジション解析 (SCA)

CVE 脆弱性チェック / イメージスキャン

▼動的解析 (DAST)

ペネトレーションテスト

▼脆弱性管理

▼機能改善要望

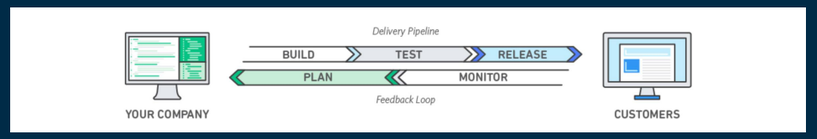
トリアージ等

※商用ツールを含めるとさらに細分化されるが、
ここでは基本的な内容のみ列挙している。

※上記は以下内容を参考に登壇者が独自に追記したもの。

https://cloudsecurityalliance.org/artifacts/devsecops-automation/

コンテナ視点で必要なセキュリティ対策を俯瞰してみる



Build → Test → Release → Monitor → Plan

コンテナ視点で。

▼静的解析 (SAST)

ベストプラクティスチェック

CIS ベンチマーク (コンテナイメージ)

▼ ソフトウェアコンポジション解析 (SCA)

CVE 脆弱性チェック / イメージスキャン

▼脆弱性管理

▼機能改善要望

トリアージ等

自動化可能な観点

▼動的解析 (DAST)

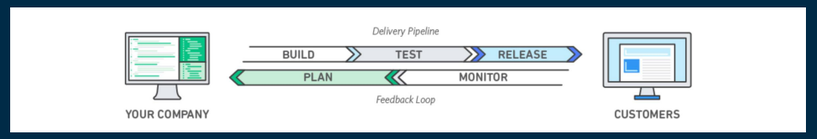
ペネトレーションテスト

※商用ツールを含めるとさらに細分化されるが、
ここでは基本的な内容のみ列挙している。

※上記は以下内容を参考に登壇者が独自に追記したもの。

https://cloudsecurityalliance.org/artifacts/devsecops-automation/

コンテナ視点で必要なセキュリティ対策を俯瞰してみる



Build → Test → Release → Monitor → Plan

コンテナ視点で。

▼静的解析（SAST）

ベストプラクティスチェック

CIS ベンチマーク（コンテナイメージ）

dockle

▼脆弱性管理

Security Hub

▼機能改善要望

トリアージ等

▼ソフトウェアコンポジション解析（SCA）

CVE 脆弱性チェック / イメージスキャン

trivy

自動化可能な観点

▼動的解析（DAST）

ペネトレーションテスト

※商用ツールを含めるとさらに細分化されるが、
ここでは基本的な内容のみ列挙している。

※上記は以下内容を参考に登壇者が独自に追記したもの。
https://cloudsecurityalliance.org/artifacts/devsecops-automation/

hadolint (Haskell Docker Linter)

- ✓ Dockerfile の Lint ツール（コードのチェックを行う）

  イメージ作成プロセスの妥当性チェック

- ✓ OSS で提供されており、Visual Studio Code プラグインでも提供

- ✓ Docker 社が提供するベストプラクティスのうち40個と
  shellcheck (シェルスクリプトの静的解析ツール)のルール33個に従ってチェック

※参考

https://github.com/hadolint/hadolint

hadolint (Haskell Docker Linter)

```
FROM debian

RUN node_version = “0.10”  && apt-get update && apt-get -y install nodejs="$node_version”
COPY package.json usr/src//app
RUN cd /usr/src/app && npm install node-static

EXPOSE 80000
CMD ["npm", "start"]
```

```
❯ docker run --rm -i hadolint/hadolint hadolint - --ignore DL3018 < Dockerfile
/dev/stdin:1 DL3006 Always tag the version of an image explicitly
/dev/stdin:3 SC1068 Don't put spaces around the = in assignments (or quote to make it literal).
/dev/stdin:3 DL3009 Delete the apt-get lists after installing something
/dev/stdin:3 DL3015 Avoid additional packages by specifying `--no-install-recommends`
/dev/stdin:6 DL3003 Use WORKDIR to switch to a directory
/dev/stdin:6 DL3016 Pin versions in npm. Instead of `npm install <package>` use `npm install <package>@<version>`
/dev/stdin:9 DL3011 Valid UNIX ports range from 0 to 65535
```

Dockle

- ✓ OSS で提供されているコンテナイメージチェックツール

  ※ 開発者は 天地知也 氏

- ✓ コンテナイメージのベストプラクティスチェックと
  CIS ベンチマークの準拠チェックを同時に実施してくれる

- ✓ インストールやスキャン方法が非常にシンプル

※参考
https://github.com/goodwithtech/dockle

Dockle


```
❯ dockle trivy_test:v1


WARN    - CIS-DI-0001: Create a user for the container
        * Last user should not be root
INFO    - CIS-DI-0005: Enable Content trust for Docker
        * export DOCKER_CONTENT_TRUST=1 before docker pull/build
INFO    - CIS-DI-0006: Add HEALTHCHECK instruction to the container image
        * not found HEALTHCHECK statement
INFO    - CIS-DI-0008: Confirm safety of setuid/setgid files
        * setuid file: bin/mount urwxr-xr-x
        * setuid file: bin/umount urwxr-xr-x
        * setuid file: usr/bin/newgrp urwxr-xr-x
        * setuid file: usr/bin/chsh urwxr-xr-x
        * setgid file: usr/bin/wall grwxr-xr-x
        * setuid file: usr/bin/chfn urwxr-xr-x
        * setuid file: usr/bin/gpasswd urwxr-xr-x
        * setgid file: usr/bin/chage grwxr-xr-x
        * setuid file: bin/su urwxr-xr-x
        * setgid file: usr/bin/expiry grwxr-xr-x
        * setuid file: usr/bin/passwd urwxr-xr-x
        * setgid file: sbin/pam_extrausers_chkpwd grwxr-xr-x
        * setgid file: sbin/unix_chkpwd grwxr-xr-x
```

CIS ベンチマークに従った
チェックを実施してくれる

Trivy

- ✓ Aqua Security 社提供の OSS イメージスキャンツール (CNCF)

  ※ 開発者は 福田 鉄平 氏

- ✓ アプリケーションパッケージの脆弱性まで踏み込んでスキャンしてくれる

- ✓ インストールやスキャン方法が非常にシンプル

  CI/CD への組み込みを前提に開発されている

※参考

https://github.com/aquasecurity/trivy

Trivy

```
❯ trivy --no-progress  trivy_test:v1
2020-07-05T01:59:46.985+0900    INFO    Detecting Ubuntu vulnerabilities...

trivy_test:v1 (ubuntu 18.04)
============================
Total: 113 (UNKNOWN: 0, LOW: 82, MEDIUM: 31, HIGH: 0, CRITICAL: 0)

+---------------------+------------------+----------+-------------------------+-------------------+---------------------------------+
|       LIBRARY       | VULNERABILITY ID | SEVERITY |    INSTALLED VERSION    |   FIXED VERSION   |              TITLE              |
+---------------------+------------------+----------+-------------------------+-------------------+---------------------------------+
| bash                | CVE-2019-18276   | LOW      | 4.4.18-2ubuntu1.2       |                   | bash: when effective UID is     |
|                     |                  |          |                         |                   | not equal to its real UID       |
|                     |                  |          |                         |                   | the...                          |
+---------------------+------------------+          +-------------------------+-------------------+---------------------------------+
| bsdutils            | CVE-2018-7738    |          | 2.31.1-0.4ubuntu3.6     |                   | util-linux: Shell command       |
|                     |                  |          |                         |                   | injection in unescaped          |
|                     |                  |          |                         |                   | bash-completed mount point      |
|                     |                  |          |                         |                   | names                           |
+---------------------+------------------+          +-------------------------+-------------------+---------------------------------+
| coreutils           | CVE-2016-2781    |          | 8.28-1ubuntu1           |                   | coreutils: Non-privileged       |
|                     |                  |          |                         |                   | session can escape to the       |
|                     |                  |          |                         |                   | parent session in chroot        |
+---------------------+------------------+----------+-------------------------+-------------------+---------------------------------+
| curl                | CVE-2020-8177    | MEDIUM   | 7.58.0-2ubuntu3.8       | 7.58.0-2ubuntu3.9 | curl: command line arguments    |
|                     |                  |          |                         |                   | lead to local file overwrite    |
```

各ライブラリ毎に CVE と
SECERITY（重大度）を一覧で表示してくれる

補足：なぜ ECR イメージスキャンではなく、Trivy を利用するのか？

ECR イメージスキャンと Trivy の大きな差は、検出精度とスキャン対象スコープの違い

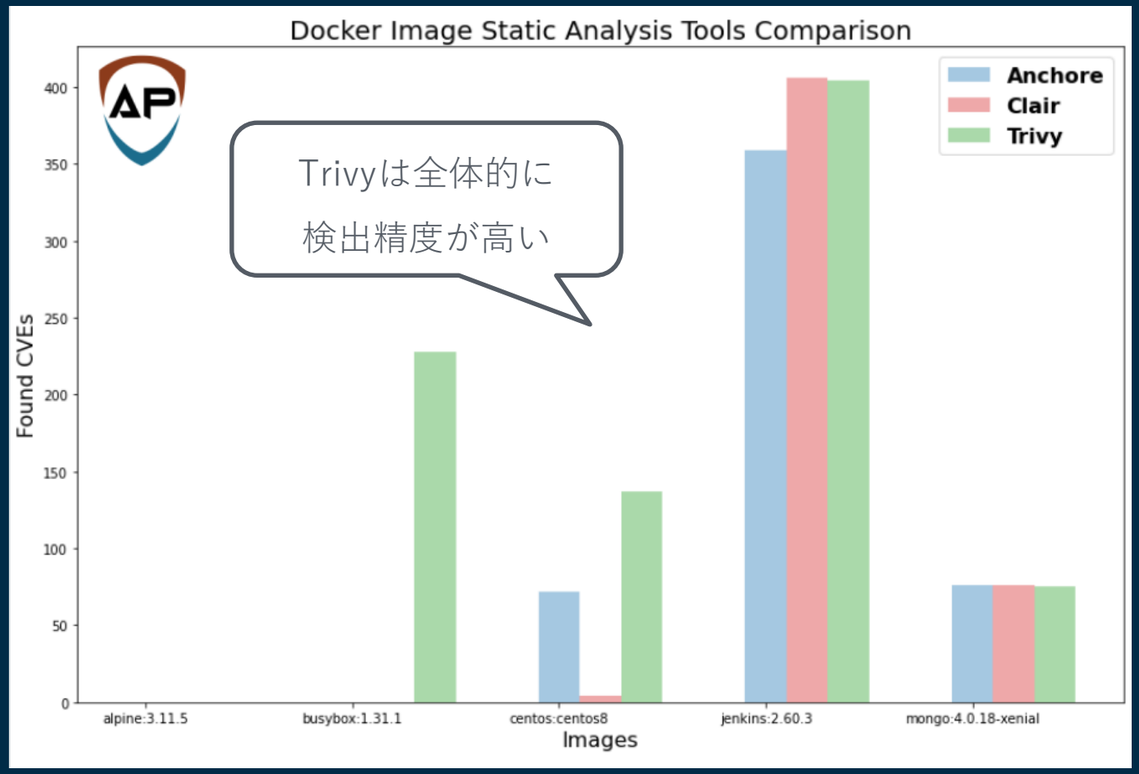
✓ ECR は Clair という OSS がベース

検出精度は2倍程度の差

最新バージョンの alpine や busybox は対象外

✓ ECR は無料なのでとりあえず有効化は◯

ECR イメージスキャンのみで PCIDSS 準拠事例あり



※図の一部を引用
https://www.a10o.net/devsecops/docker-image-security-static-analysis-tool-comparison-anchore-engine-vs-clair-vs-trivy/

AWS Security Hub

✓ マネージドなセキュリティアラート一元管理サービス

✓ 各セキュリティ項目の準拠状況が確認でき、ワークフローとしても利用可能

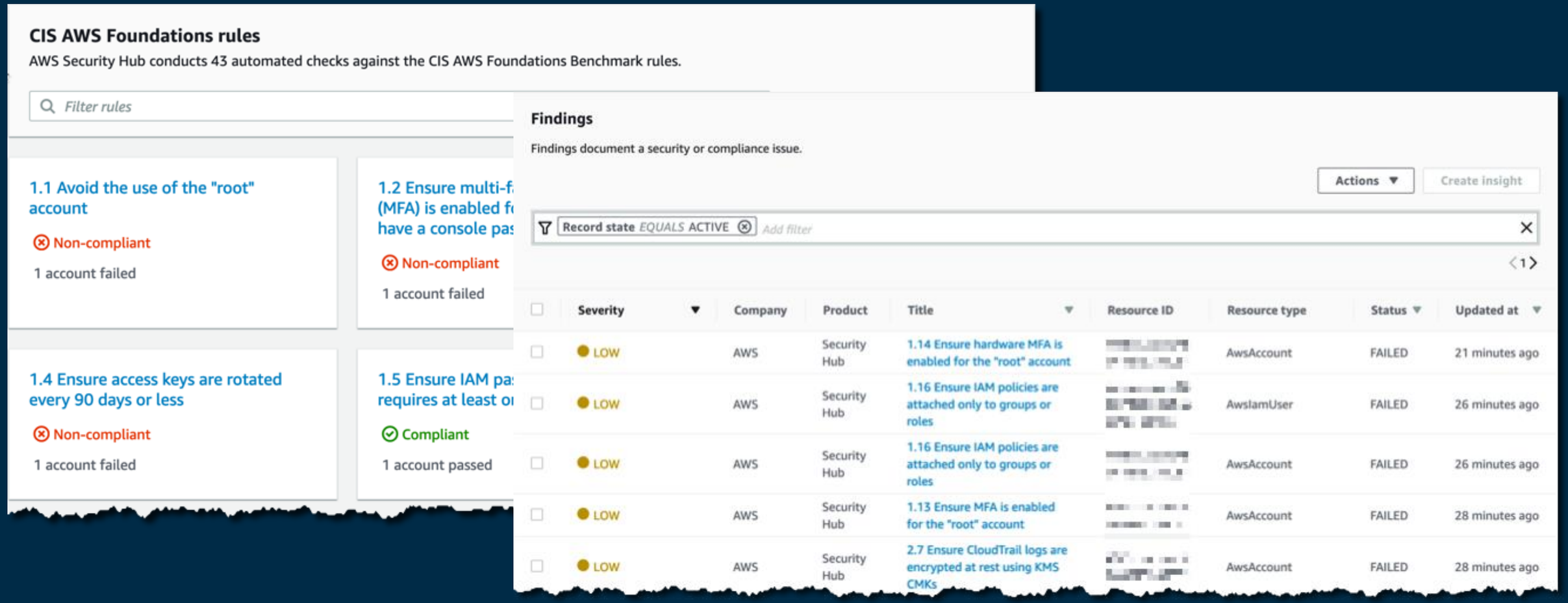
CIS AWS Foundations Benchmark や PCIDSS など各業界標準やベストプラクティスに基づくチェックが可能

✓ サードパーティ製セキュリティツールとの統合が可能

※参考

https://aws.amazon.com/jp/security-hub/

AWS Security Hub



In Partnership with intel

一部リスクに対して各ツールを駆使してコンテナセキュリティを高める

リスク

項番	項目	対策
1.1	イメージの脆弱性	脆弱性を排除する、かつ脆弱性を含んだイメージをデプロイしないようにする
1.2	イメージの設定不備	ベストプラクティスに従ったイメージ作成を行う
1.3	埋め込まれたマルウェア	信頼されたイメージやレジストリを利用する
1.4	埋め込まれた平文の秘密情報	秘密情報をイメージの外部で保存する
1.5	信頼できないイメージの使用	信頼されたイメージやレジストリを利用する

対策

In Partnership with intel

aws DEV DAY
ONLINE JAPAN

AWS × DevSecOps への歩み寄り方

In Partnership with intel

DevOps にセキュリティを協調させるのが DevSecOps

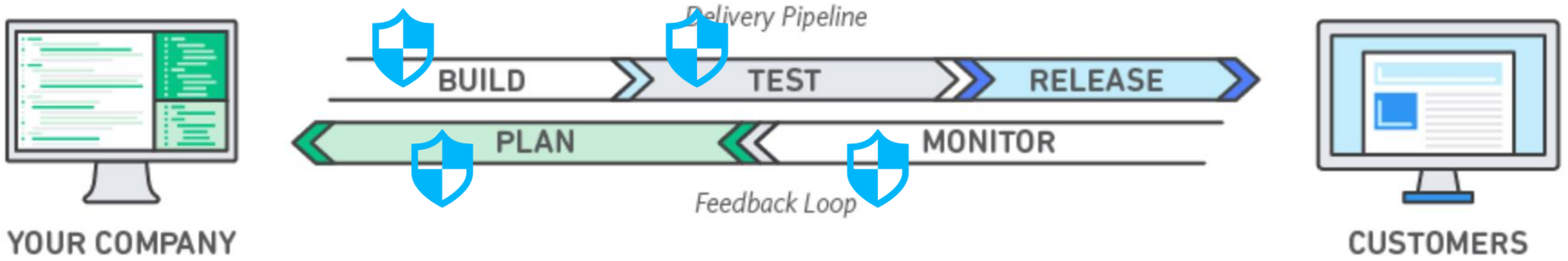
再々掲

DevOps と同様に組織の文化、プラクティス、ツールをセキュリティの目線で取り入れる

全員がセキュリティに責務を持つ

シフトレフト、CI/CD 等

OSS 等



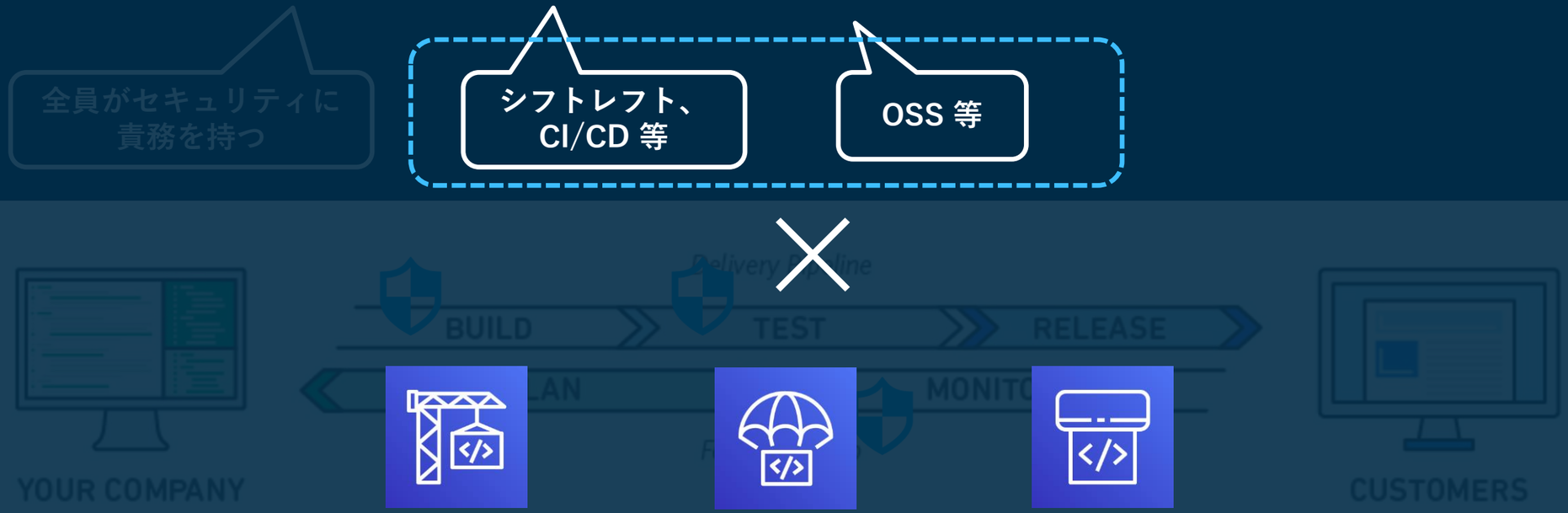
※図の一部を引用

https://aws.amazon.com/jp/devops/what-is-devops/

DevOps にセキュリティを協調させるのが DevSecOps

再々掲

DevOps と同様に組織の文化、プラクティス、ツールをセキュリティの目線で取り入れる

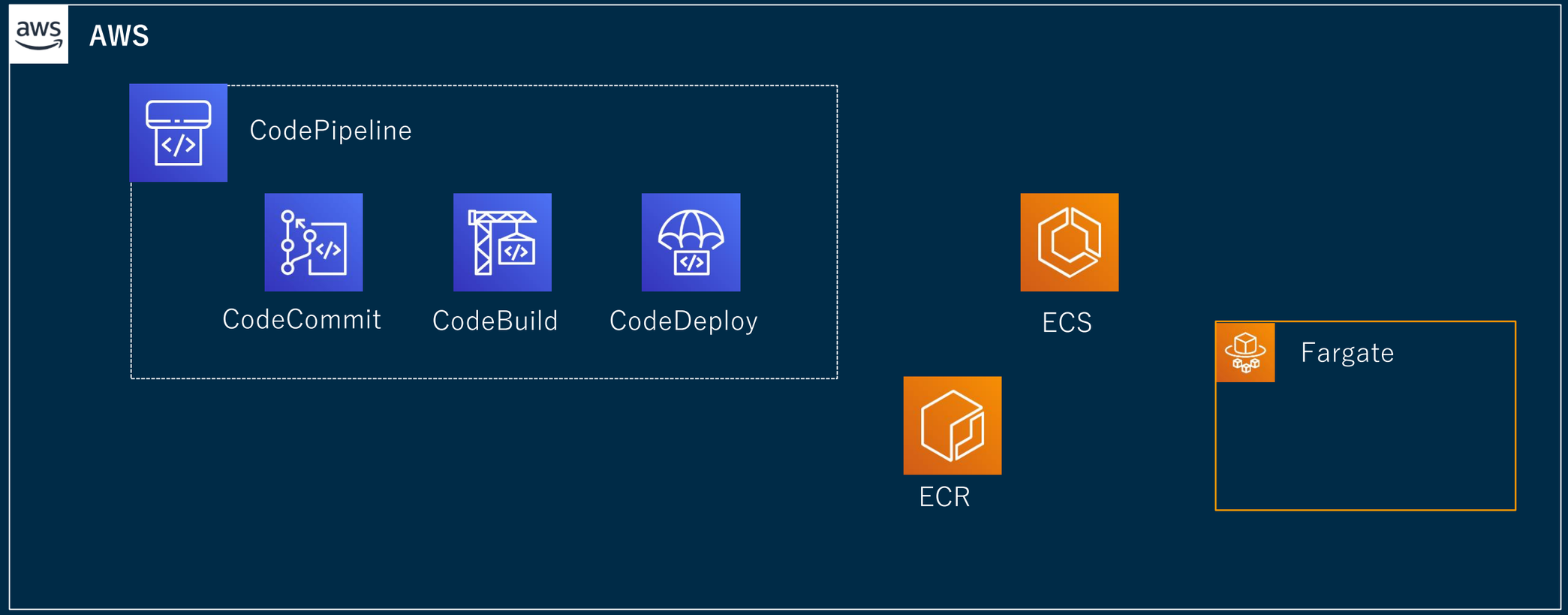


AWS CI/CD 系サービスである

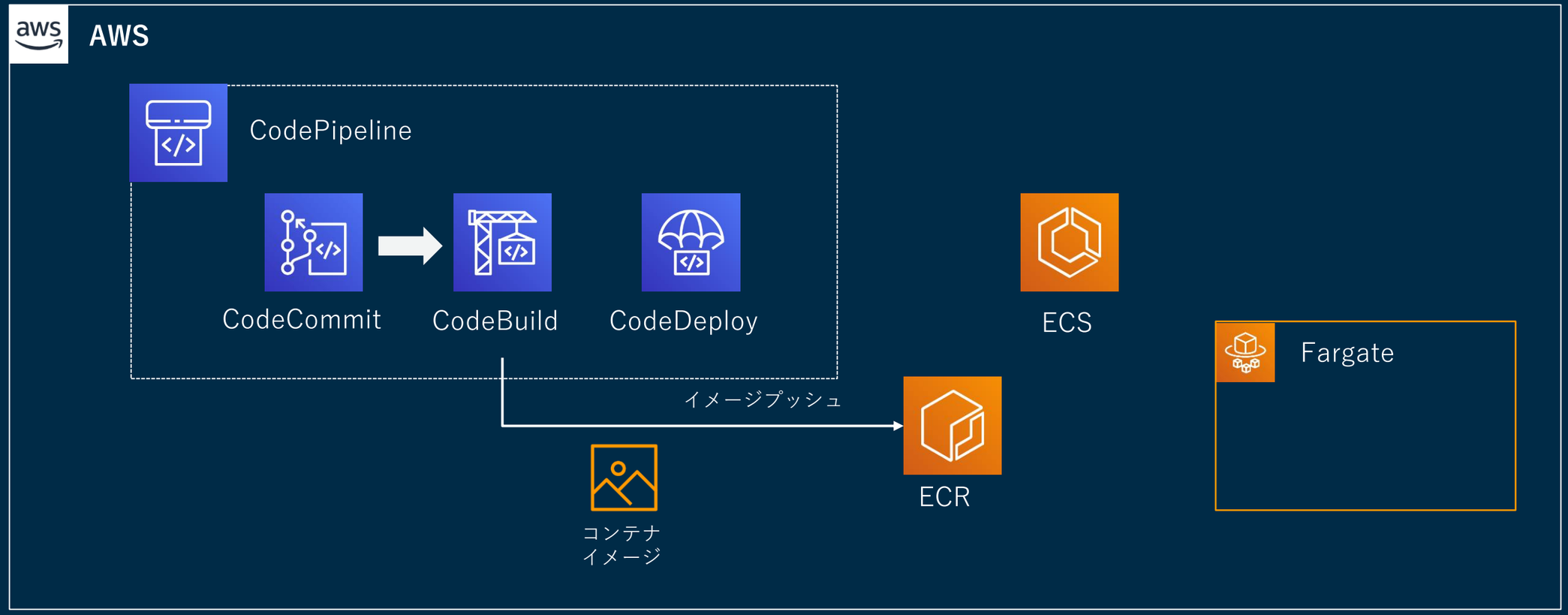
※図の一部を引用
https://aws.amazon.com/jp/devops/what-is-devops/

Code シリーズへの組み込みを検討

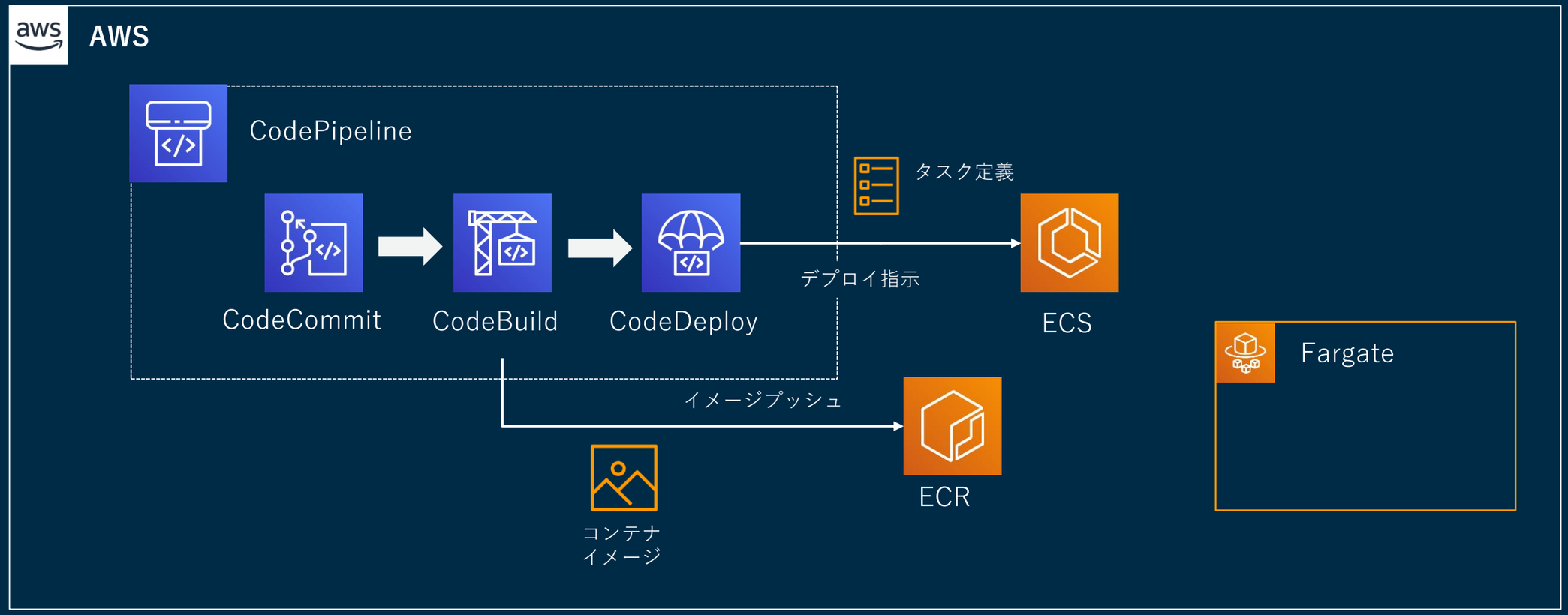
Amazon ECS / Fargate をベースとして CI/CD をインテグレーション



Amazon ECS / Fargate をベースとして CI/CD をインテグレーション

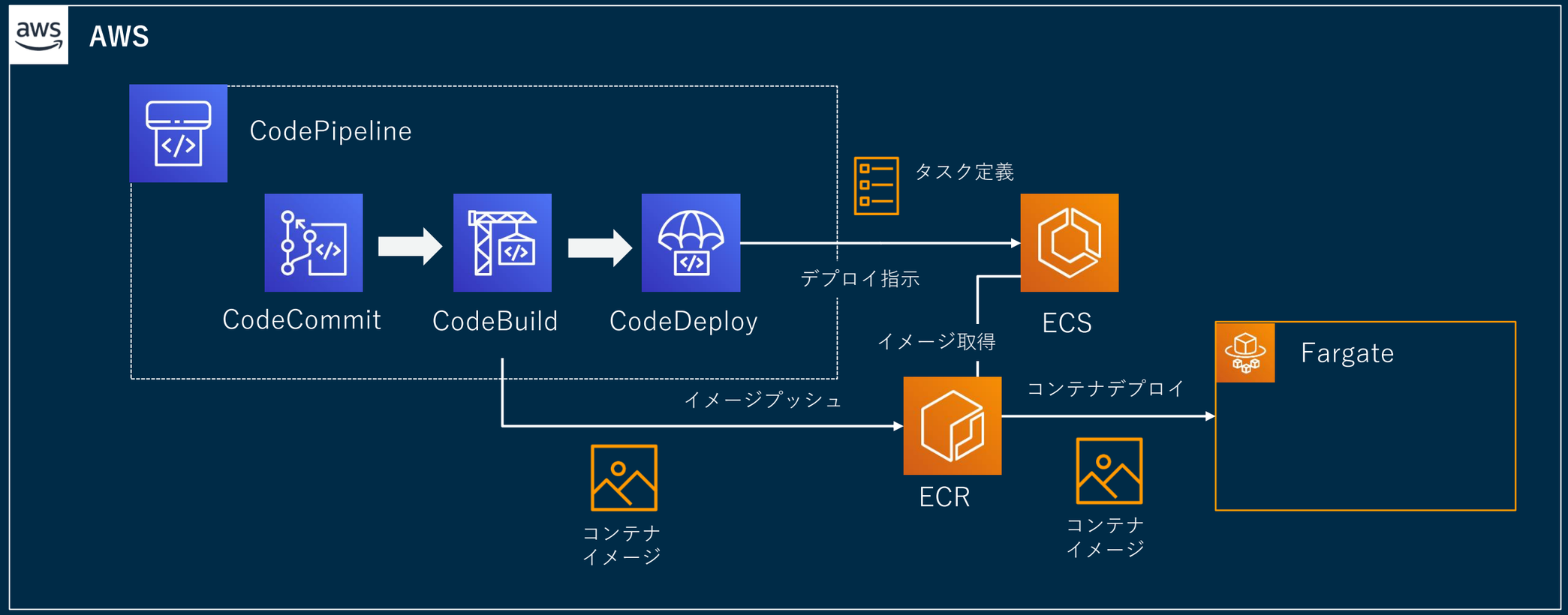


Amazon ECS / Fargate をベースとして CI/CD をインテグレーション

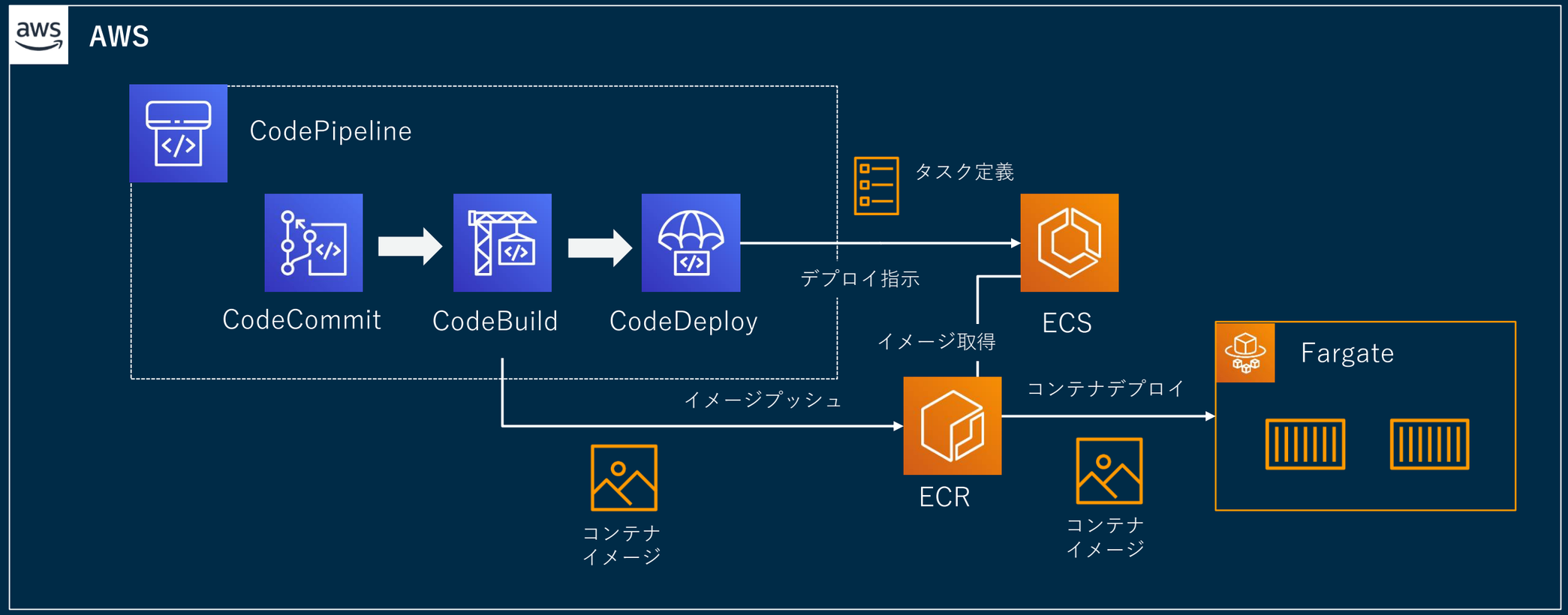


In Partnership with intel

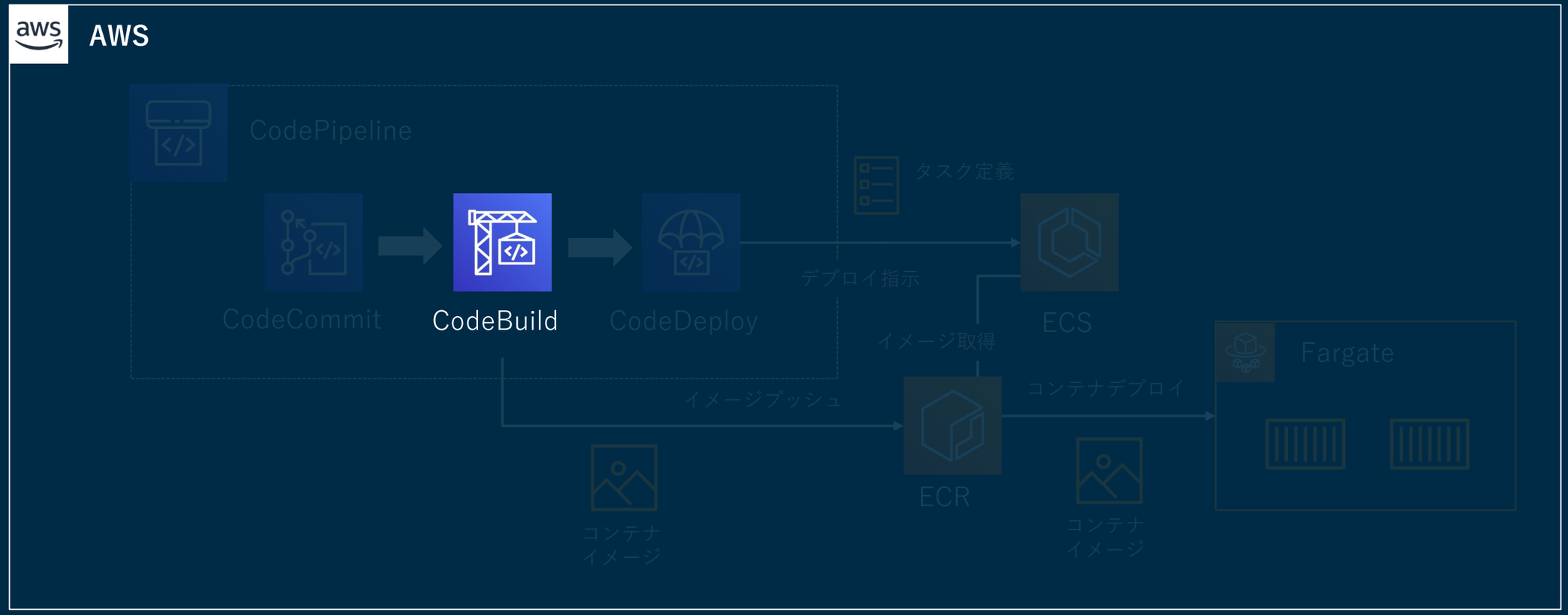
Amazon ECS / Fargate をベースとして CI/CD をインテグレーション



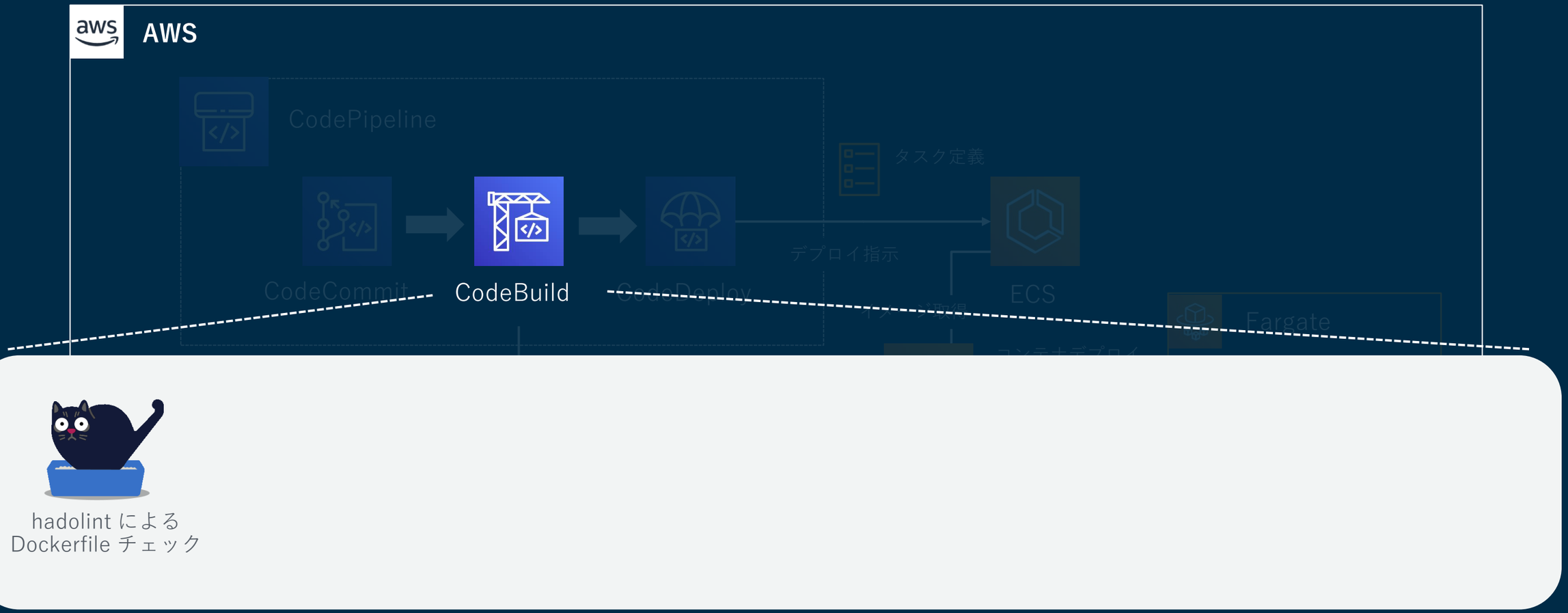
Amazon ECS / Fargate をベースとして CI/CD をインテグレーション



ビルド・テストフェーズを担う CodeBuild に OSS をシフトレフトで組み込む



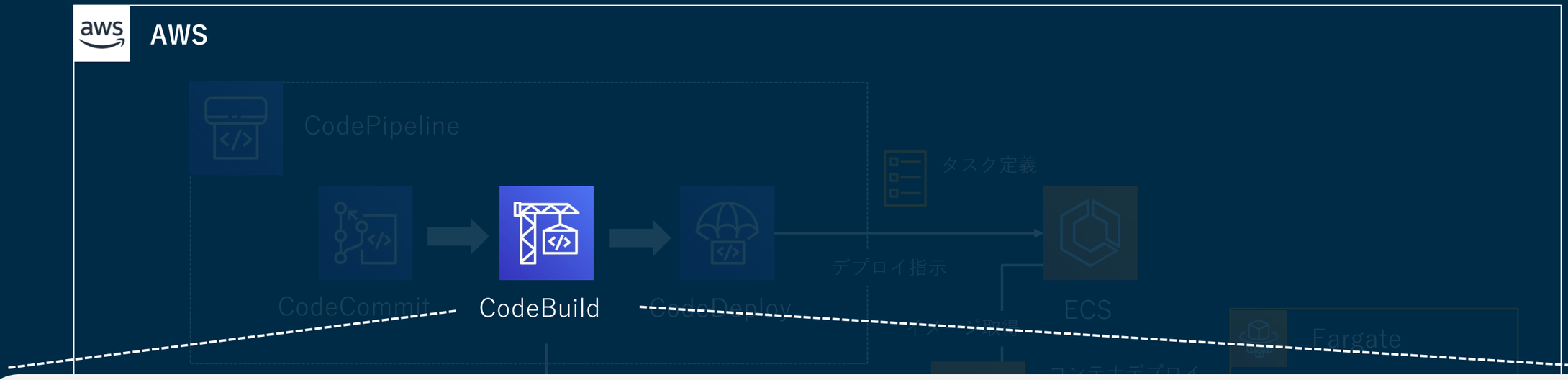
ビルド・テストフェーズを担う CodeBuild に OSS をシフトレフトで組み込む



hadolint による
Dockerfile チェック

In Partnership with intel

ビルド・テストフェーズを担う CodeBuild に OSS をシフトレフトで組み込む

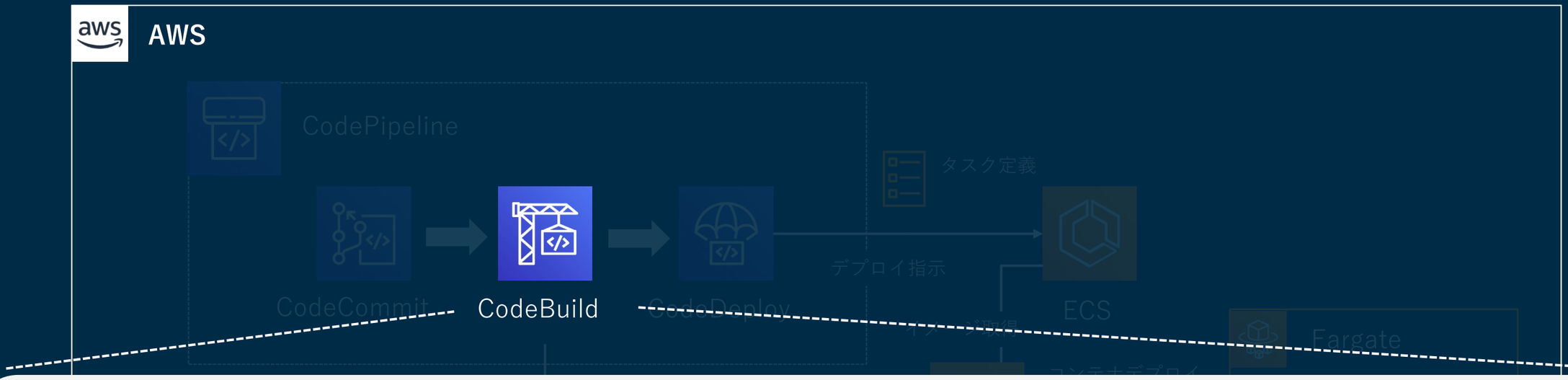


hadolint による
Dockerfile チェック

アプリケーションの
ビルド & テスト

In Partnership with intel

ビルド・テストフェーズを担う CodeBuild に OSS をシフトレフトで組み込む



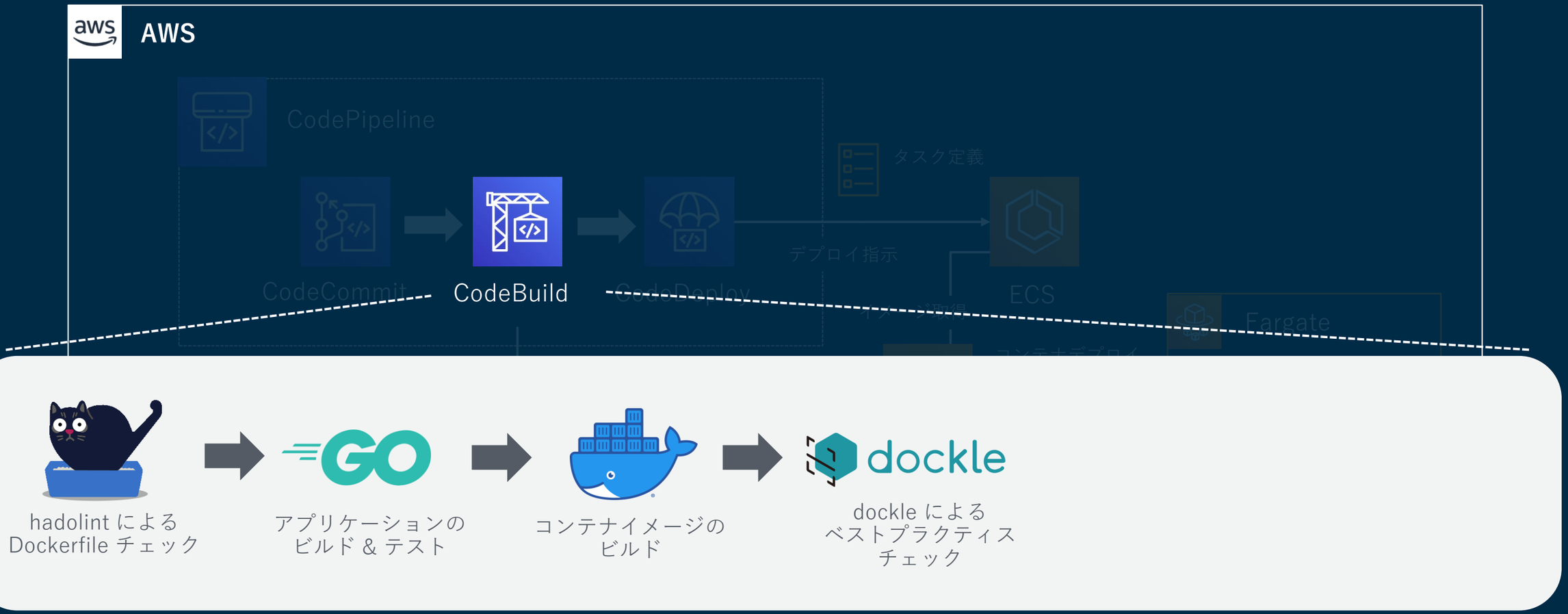
hadolint による
Dockerfile チェック

アプリケーションの
ビルド & テスト

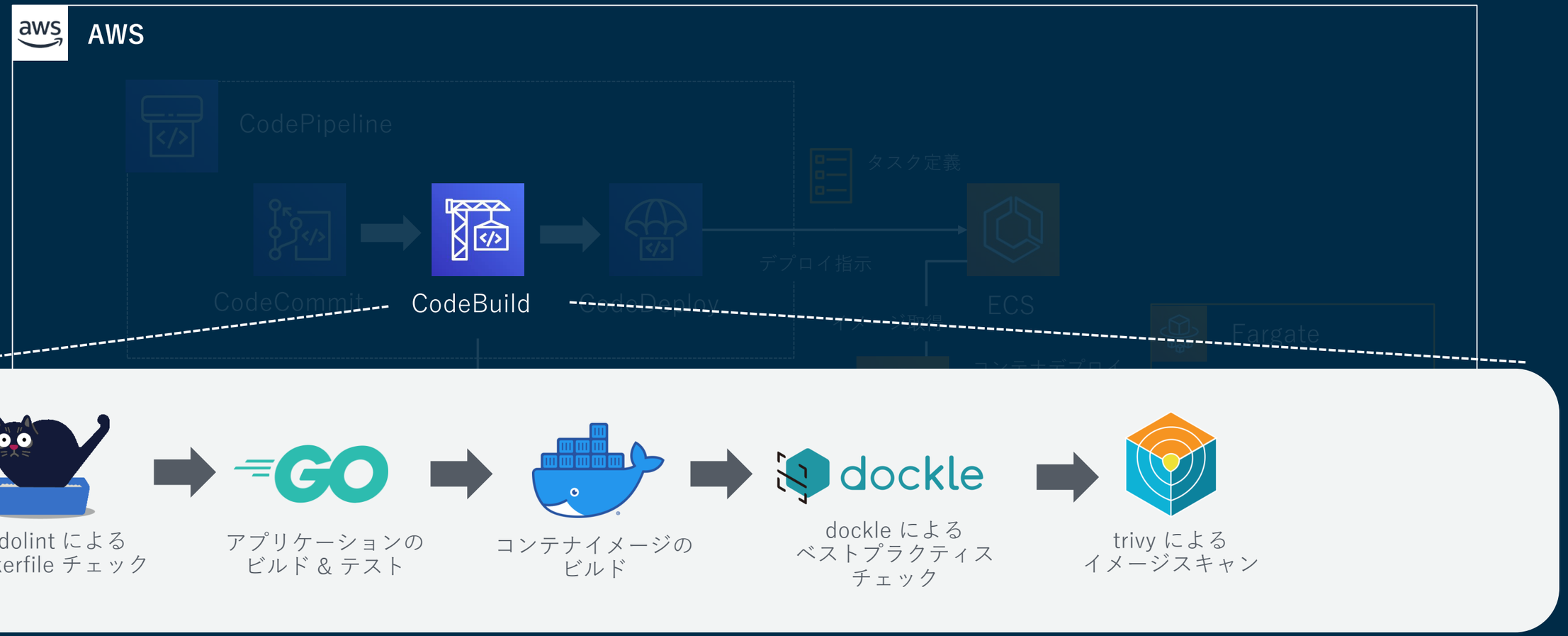
コンテナイメージの
ビルド

In Partnership with intel

ビルド・テストフェーズを担う CodeBuild に OSS をシフトレフトで組み込む

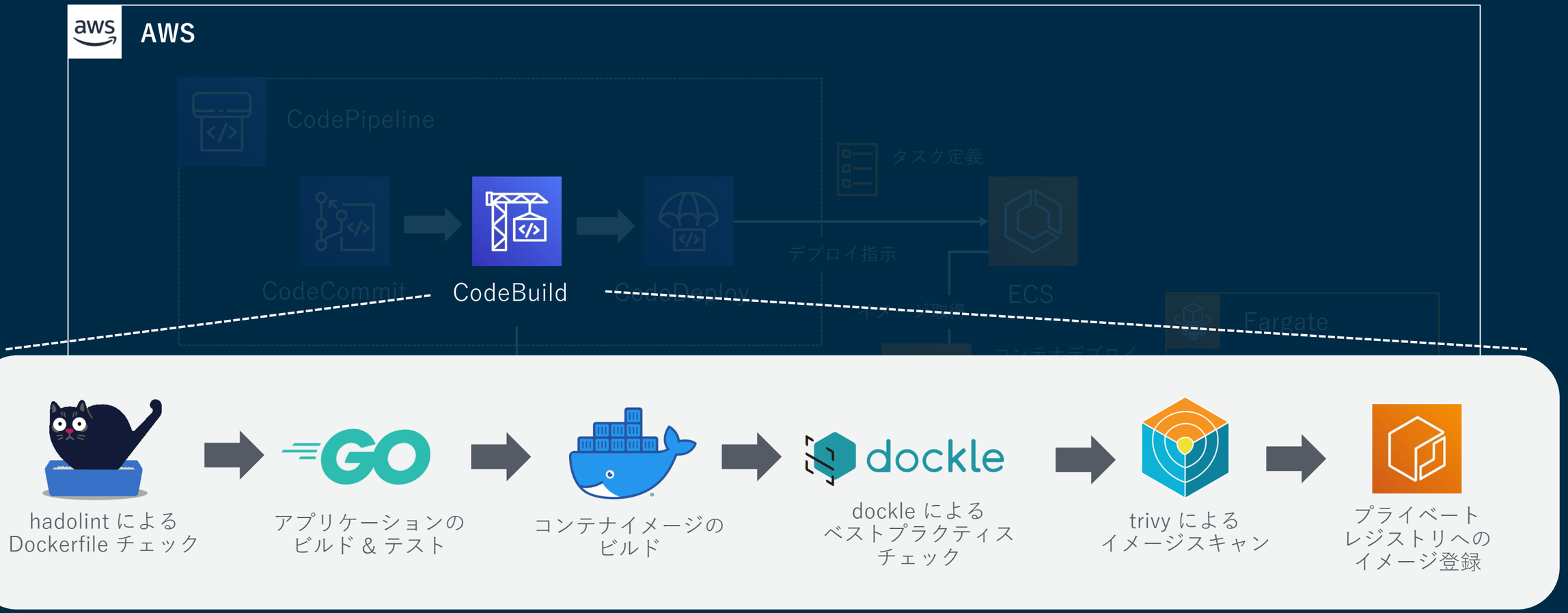


ビルド・テストフェーズを担う CodeBuild に OSS をシフトレフトで組み込む

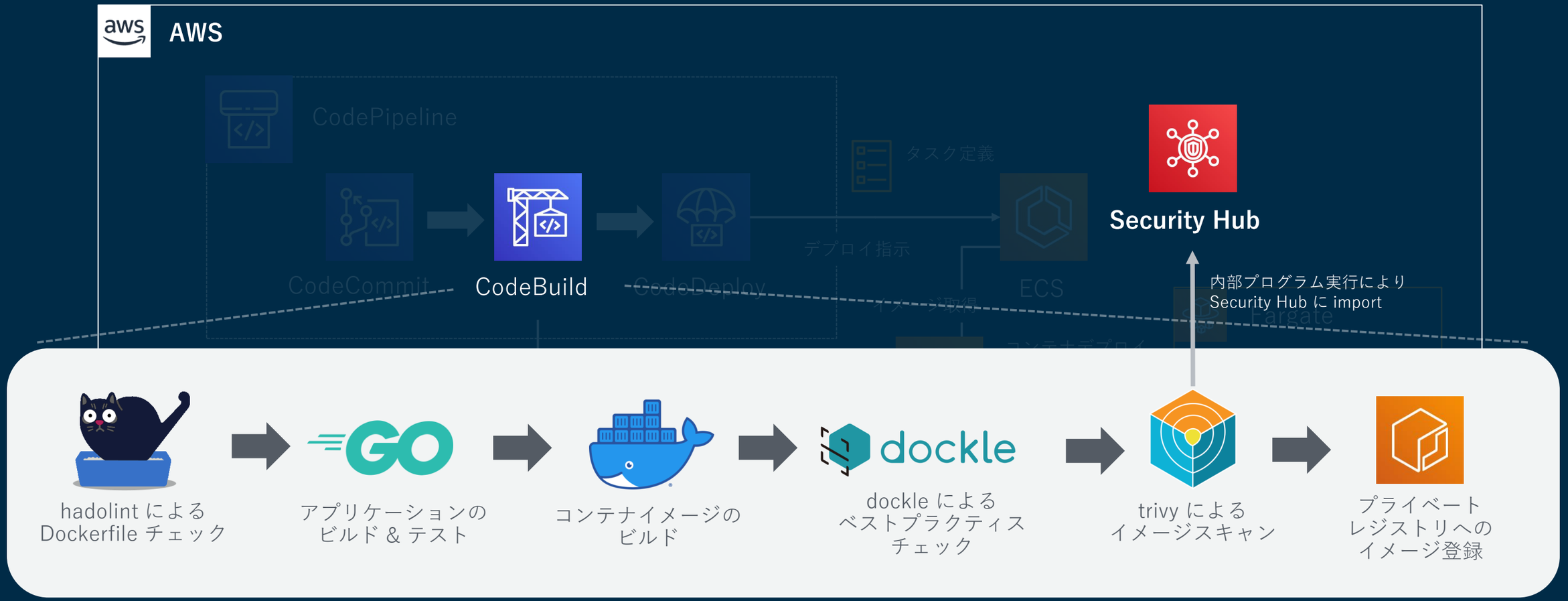


In Partnership with intel

ビルド・テストフェーズを担う CodeBuild に OSS をシフトレフトで組み込む

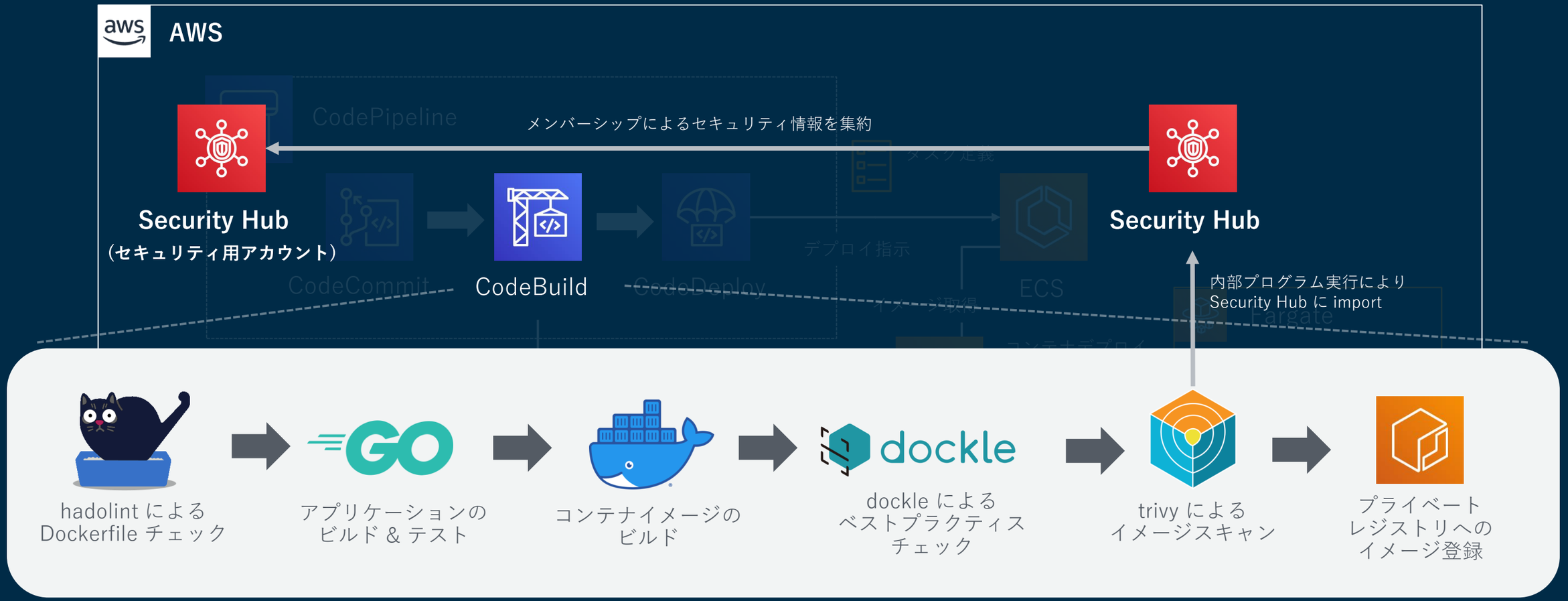


Security Hub と統合して継続的なセキュリティモニタリングに備える



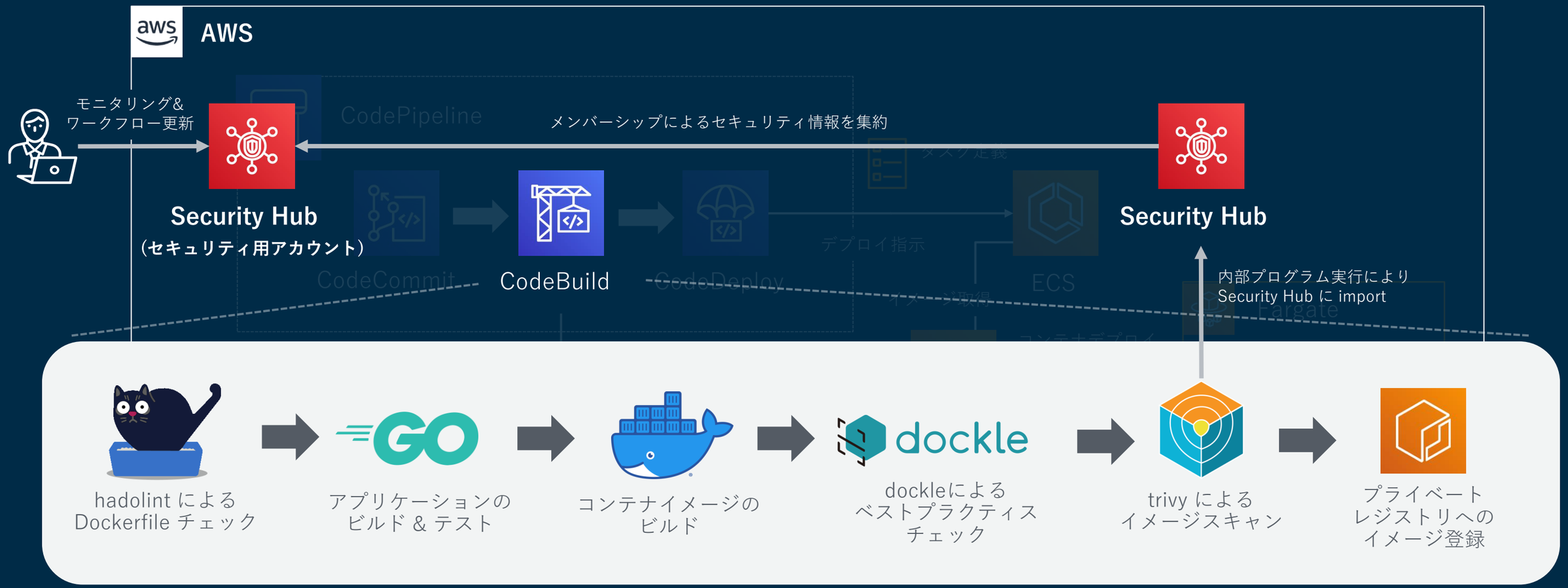
In Partnership with intel

Security Hub と統合して継続的なセキュリティモニタリングに備える



In Partnership with intel

Security Hub と統合して継続的なセキュリティモニタリングに備える



In Partnership with intel

aws DEV DAY
ONLINE JAPAN

よりセキュアなコンテナ運用を目指して

In Partnership with intel

NIST SP800-190 のその他イメージリスクにどう立ち向かうか？

再掲

その他の 1.3〜1.5 は他の AWS サービスとの連携や個別対応で満たしていかなければならない。

項番	項目	リスク内容	対策
1.1	イメージの脆弱性	イメージ内コンポーネントのセキュリティアップデート漏れ等により、脆弱性を含んでいる	脆弱性を排除する、かつ脆弱性を含んだイメージをデプロイしないようにする
1.2	イメージの設定不備	イメージの設定が不適切である 例) root ユーザによる実行	ベストプラクティスに従ったイメージ作成を行う
1.3	埋め込まれたマルウェア	悪意のあるファイルがイメージに含まれている	信頼されたイメージやレジストリを利用する
1.4	埋め込まれた平文の秘密情報	イメージファイル内に予め秘密情報を平文で含んでしまうことで、セキュリティリスクが生じる	秘密情報をイメージの外部で保存する
1.5	信頼できないイメージの使用	外部の信頼できないイメージを利用してしまう	信頼されたイメージやレジストリを利用する

CI/CD と OSS を活用することで対応

※コンテナイメージに関する内容を一部引用

https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-190.pdf

NIST SP800-190 のその他イメージリスクにどう立ち向かうか？

再掲

その他の 1.3〜1.5 は他の AWS サービスとの連携や個別対応で満たしていかなければならない。

項番	項目	リスク内容	対策
1.1	イメージの脆弱性	イメージ内コンポーネントのセキュリティアップデート漏れ等により、脆弱性を含んでいる	脆弱性を排除する、かつ脆弱性を含んだイメージをデプロイしないようにする
1.2	イメージの設定不備	イメージの設定が不適切である 例) root ユーザによる実行	ベストプラクティスに従ったイメージ作成を行う
1.3	埋め込まれたマルウェア	悪意のあるファイルがイメージに含まれている	信頼されたイメージやレジストリを利用する
1.4	埋め込まれた平文の秘密情報	イメージファイル内に予め秘密情報を平文で含んでしまうことで、セキュリティリスクが生じる	秘密情報をイメージの外部で保存する
1.5	信頼できないイメージの使用	外部の信頼できないイメージを利用してしまう	信頼されたイメージやレジストリを利用する

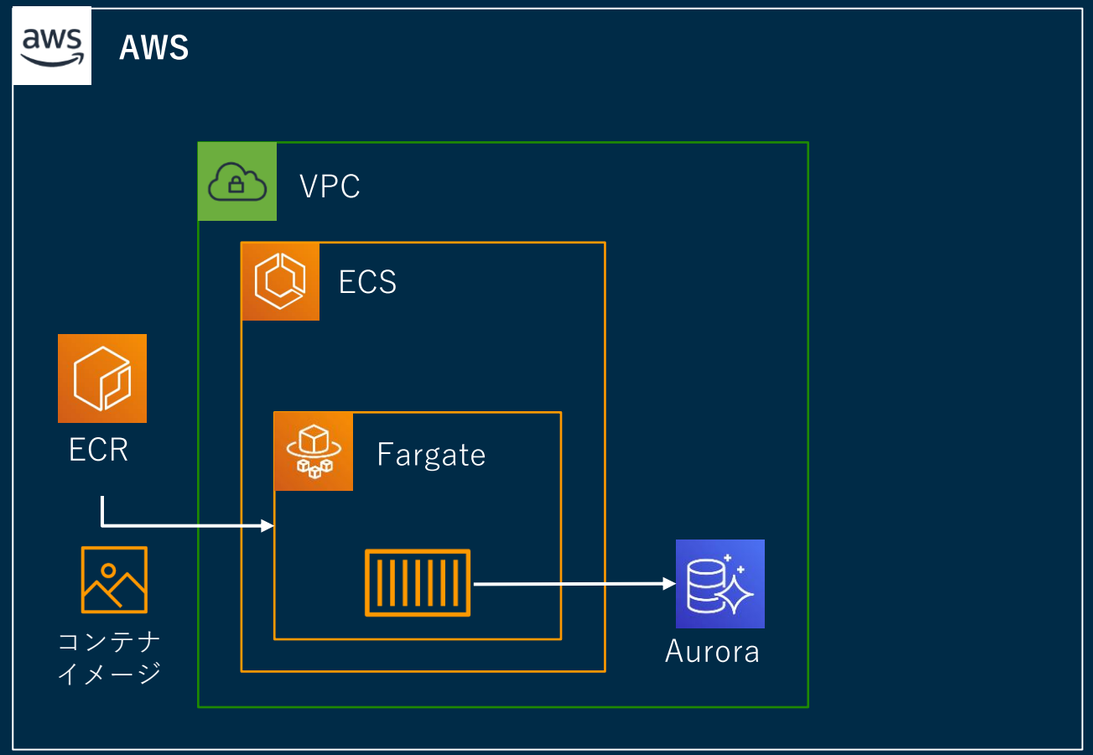
CI/CD と OSS を活用することで対応

※コンテナイメージに関する内容を一部引用

https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-190.pdf

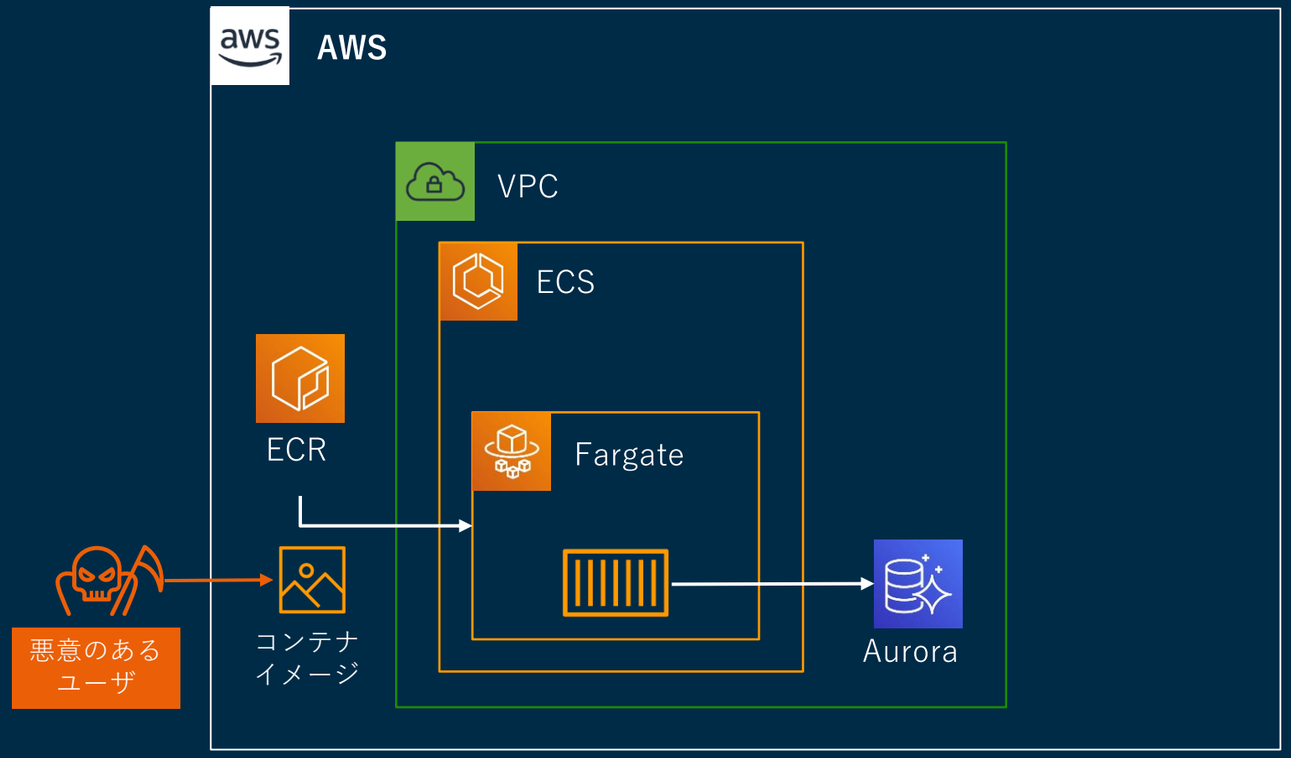
秘密情報は Secrets Manager を活用することで外部から注入する。

コンテナ内に秘密情報を平文で埋め込むと不正に入手されるリスクが介在。



秘密情報は Secrets Manager を活用することで外部から注入する。

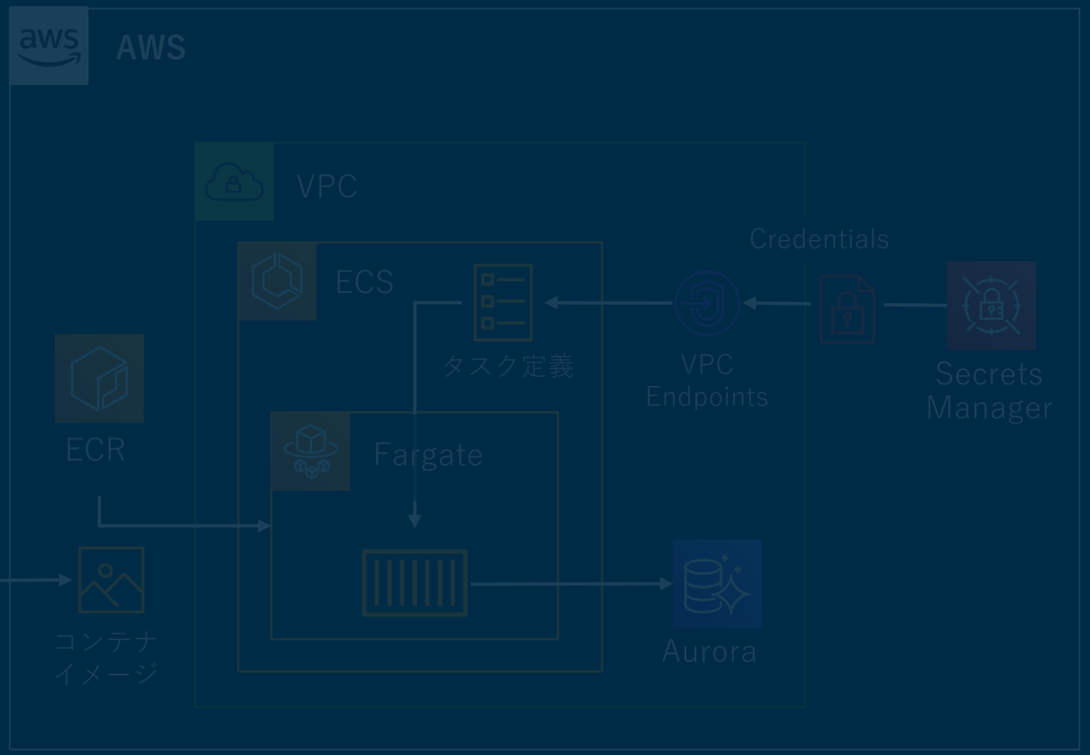
コンテナ内に秘密情報を平文で埋め込むと不正に入手されるリスクが内在。

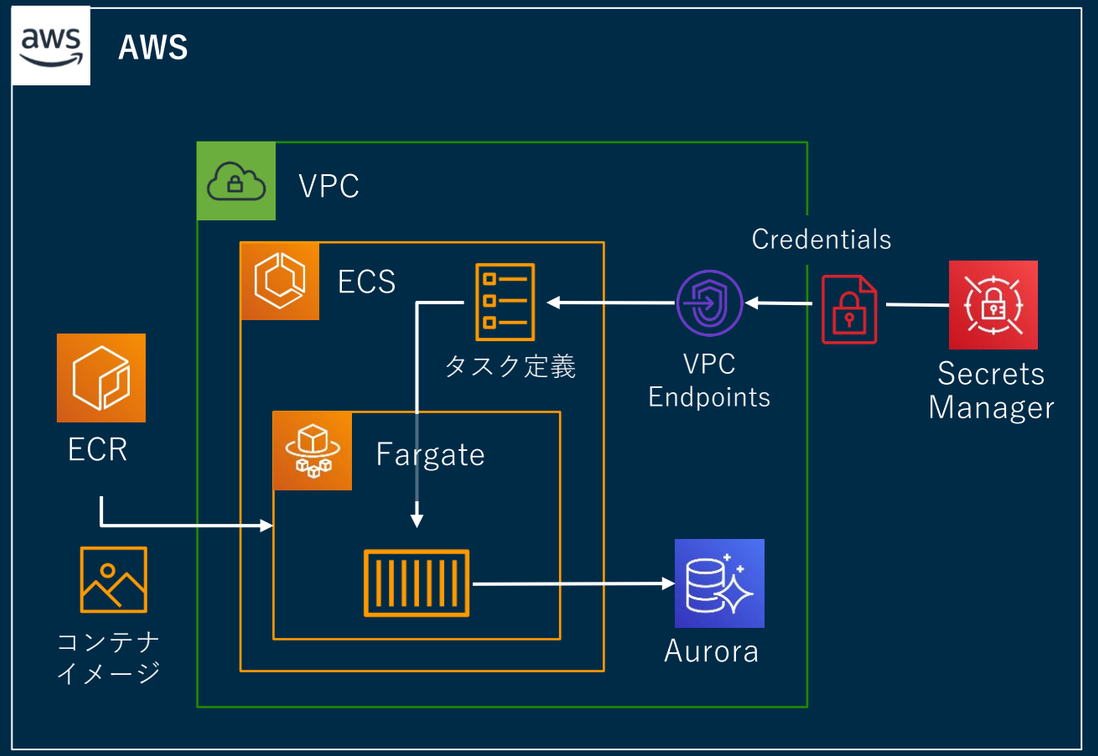


よりセキュアなコンテナ運用を目指して

秘密情報は Secrets Manager を活用することで外部から注入する。

ECS タスク定義から外部クレデンシャル情報を指定することで安全に情報を参照できる。





In Partnership with intel

NIST SP800-190 のその他イメージリスクにどう立ち向かうか？

再掲

その他の 1.3〜1.5 は他の AWS サービスとの連携や個別対応で満たしていかなければならない。

項番	項目	リスク内容	対策
1.1	イメージの脆弱性	イメージ内コンポーネントのセキュリティアップデート漏れ等により、脆弱性を含んでいる	脆弱性を排除する、かつ脆弱性を含んだイメージをデプロイしないようにする
1.2	イメージの設定不備	イメージの設定が不適切である 例) root ユーザによる実行	ベストプラクティスに従ったイメージ作成を行う
1.3	埋め込まれたマルウェア	悪意のあるファイルがイメージに含まれている	信頼されたイメージやレジストリを利用する
1.4	埋め込まれた平文の秘密情報	イメージファイル内に予め秘密情報（パスワードなど）が含まれており、セキュリティリスクが生じる	秘密情報はイメージの外部で保存する
1.5	信頼できないイメージの使用	外部の信頼できないイメージを利用してしまう	信頼されたイメージやレジストリを利用する

CI/CD と OSS を活用することで対応可能

Secrets Manager 等のサービスと連携することで対応可能

※コンテナイメージに関する内容を一部引用

https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-190.pdf

NIST SP800-190 のその他イメージリスクにどう立ち向かうか？

再掲

その他の 1.3〜1.5 は他の AWS サービスとの連携や個別対応で満たしていかなければならない。

項番	項目	リスク内容	対策
1.1	イメージの脆弱性	イメージ内コンポーネントのセキュリティアップデート漏れ等により、脆弱性を含んでいる	脆弱性を排除する、かつ脆弱性を含んだイメージをデプロイしないようにする
1.2	イメージの設定不備	イメージの設定が不適切である 例) root ユーザによる実行	ベストプラクティスに従ったイメージ作成を行う
1.3	埋め込まれたマルウェア	悪意のあるファイルがイメージに含まれている	信頼されたイメージやレジストリを利用する
1.4	埋め込まれた平文の秘密情報	イメージファイル内に予め秘密情報が含まれてしまうことで、セキュリティリスクが生じる	秘密情報はイメージの外部で保存する
1.5	信頼できないイメージの使用	外部の信頼できないイメージを利用してしまう	信頼されたイメージやレジストリを利用する

CI/CD と OSS を活用することで対応可能

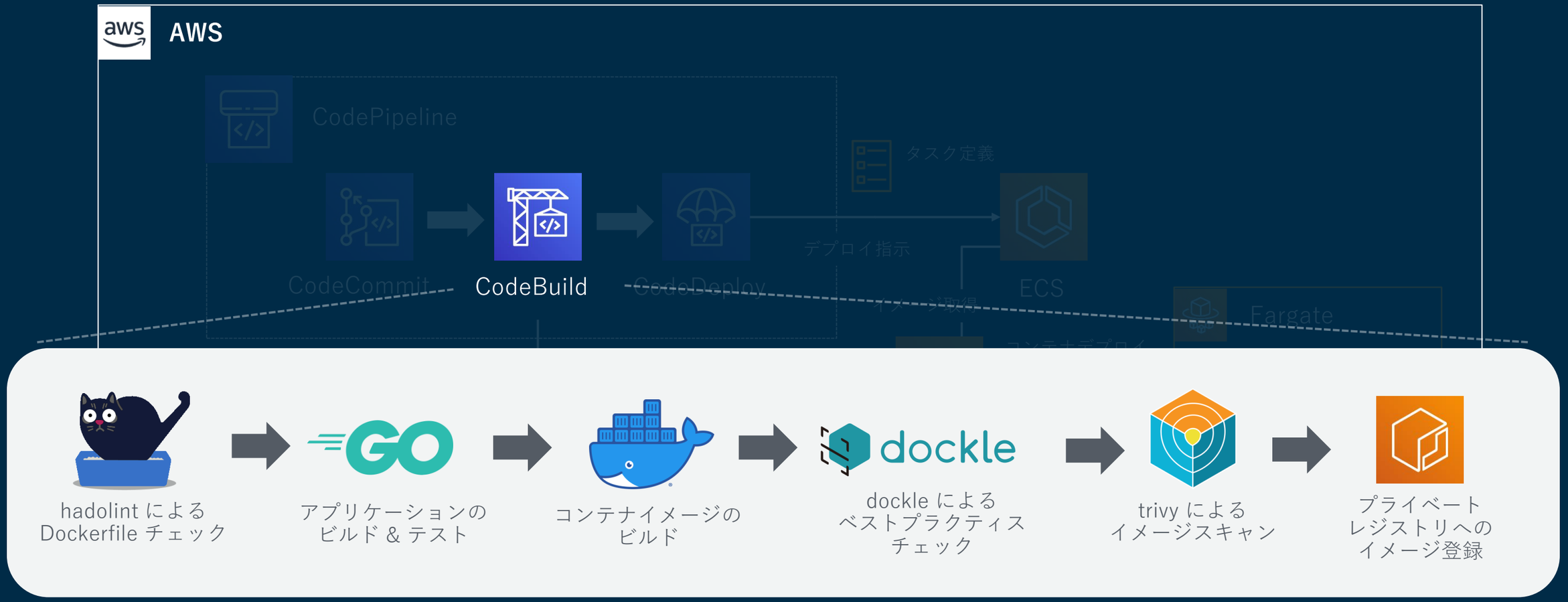
Secrets Manager 等のサービスと連携することで対応可能

※コンテナイメージに関する内容を一部引用

https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-190.pdf

よりセキュアなコンテナ運用を目指して

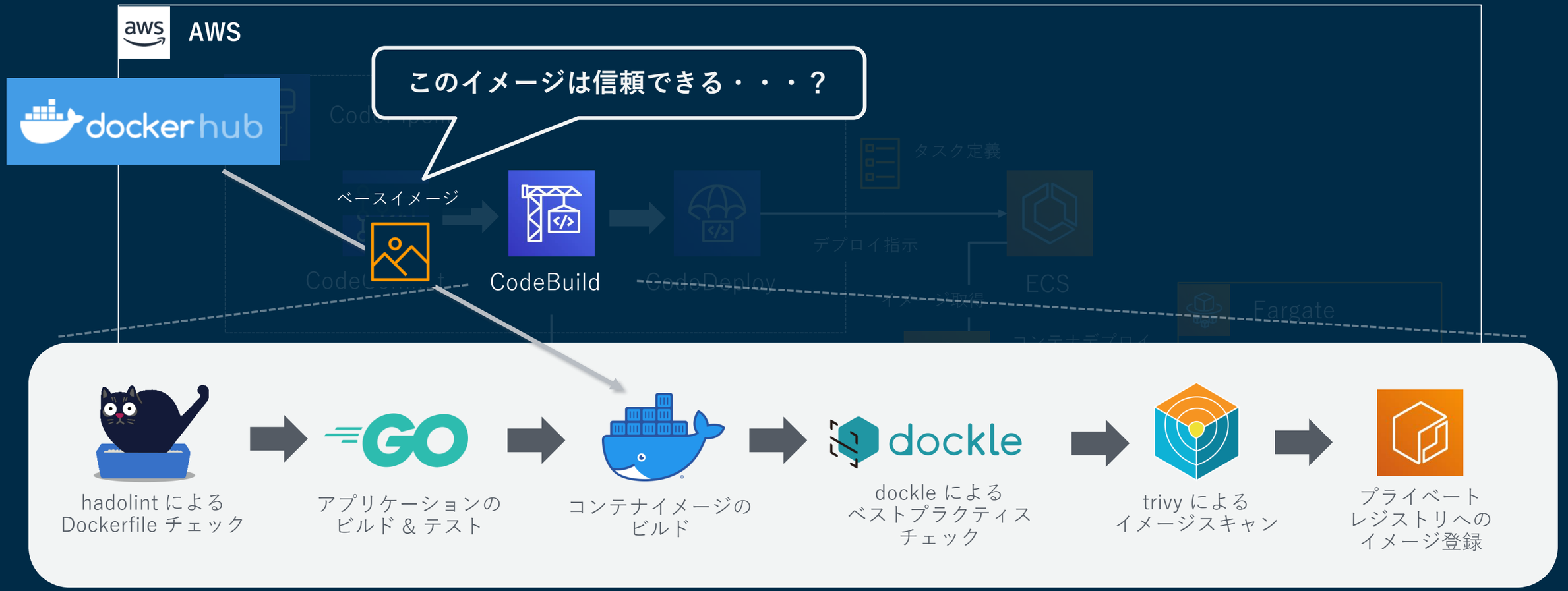
悪意のあるコンテナイメージの利用を防ぐために署名を活用する



In Partnership with intel

よりセキュアなコンテナ運用を目指して

悪意のあるコンテナイメージの利用を防ぐために署名を活用する

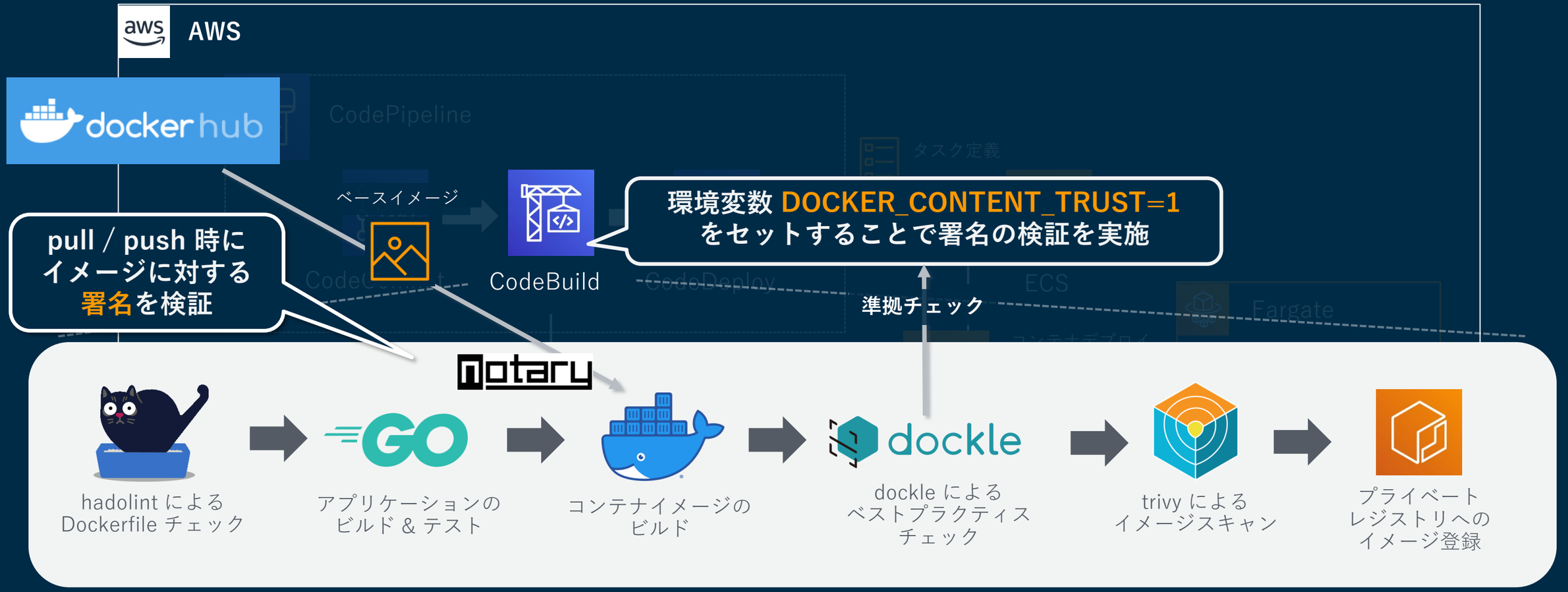


In Partnership with intel

よりセキュアなコンテナ運用を目指して

悪意のあるコンテナイメージの利用を防ぐために署名を活用する

Docker Content Trust (DCT) を利用することで、イメージのなりすましと改ざんから保護可能



In Partnership with intel

Dockle

再掲

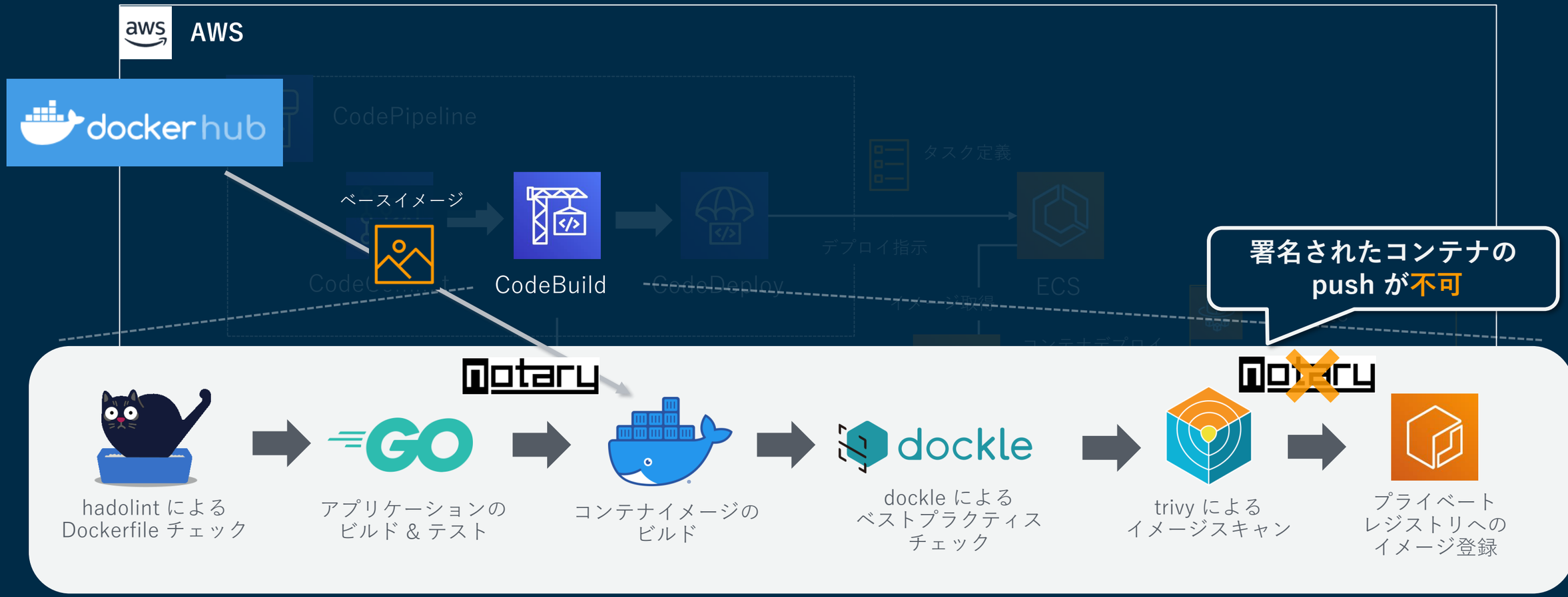
```
❯ dockle trivy_test:v1


WARN    - CIS-DI-0001: Create a user for the container
        * Last user should not be root
INFO    - CIS-DI-0005: Enable Content trust for Docker
        * export DOCKER_CONTENT_TRUST=1 before docker pull/build
INFO    - CIS-DI-0006: Add HEALTHCHECK instruction to the container image
        * not found HEALTHCHECK statement
INFO    - CIS-DI-0008: Confirm safety of setuid/setgid files
        * setuid file: bin/mount urwxr-xr-x
        * setuid file: bin/umount urwxr-xr-x
        * setuid file: usr/bin/newgrp urwxr-xr-x
        * setuid file: usr/bin/chsh urwxr-xr-x
        * setgid file: usr/bin/wall grwxr-xr-x
        * setuid file: usr/bin/chfn urwxr-xr-x
        * setuid file: usr/bin/gpasswd urwxr-xr-x
        * setgid file: usr/bin/chage grwxr-xr-x
        * setuid file: bin/su urwxr-xr-x
        * setgid file: usr/bin/expiry grwxr-xr-x
        * setuid file: usr/bin/passwd urwxr-xr-x
        * setgid file: sbin/pam_extrausers_chkpwd grwxr-xr-x
        * setgid file: sbin/unix_chkpwd grwxr-xr-x
```

CIS ベンチマークでも
DCT の有効化はチェック対象

pull 対象の dockerhub は DCT 利用可能だが、push 対象の ECR は未対応

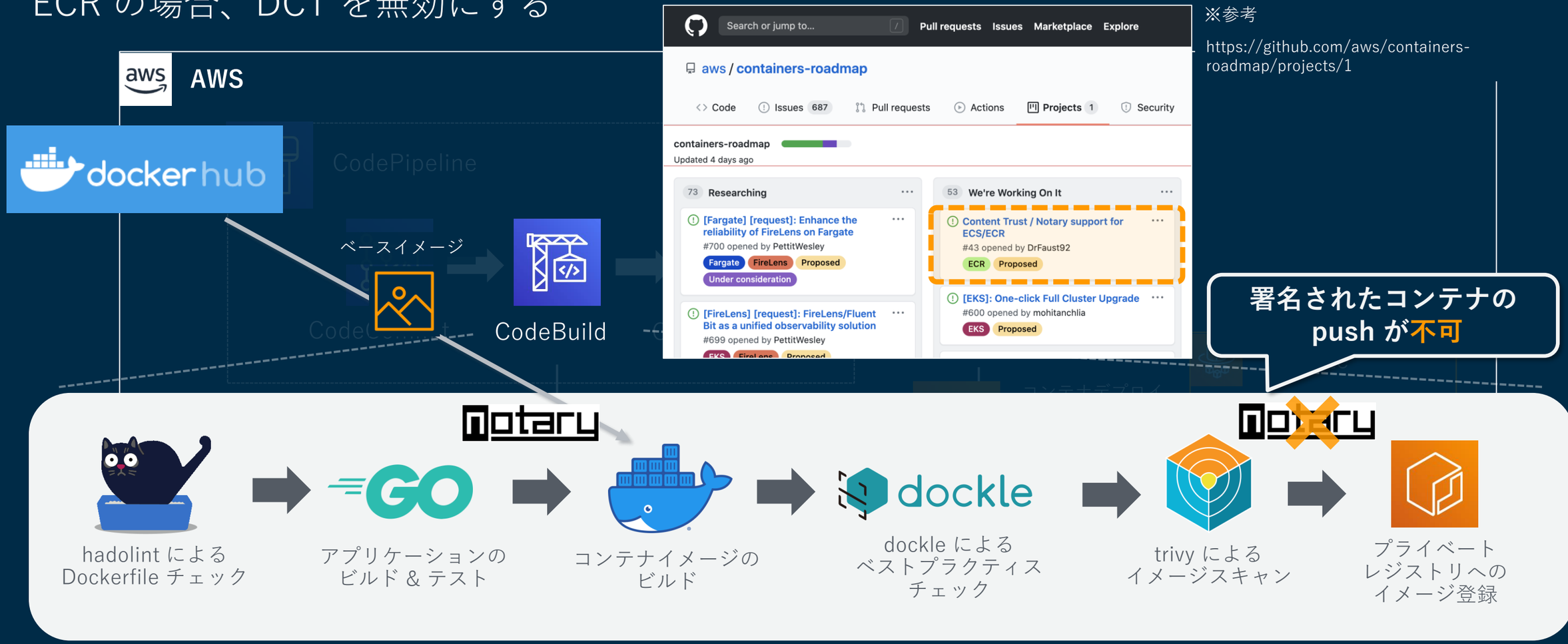
ECR の場合、DCT を無効にする



In Partnership with intel

pull 対象の dockerhub は DCT 利用可能だが、push 対象の ECR は未対応

ECR の場合、DCT を無効にする



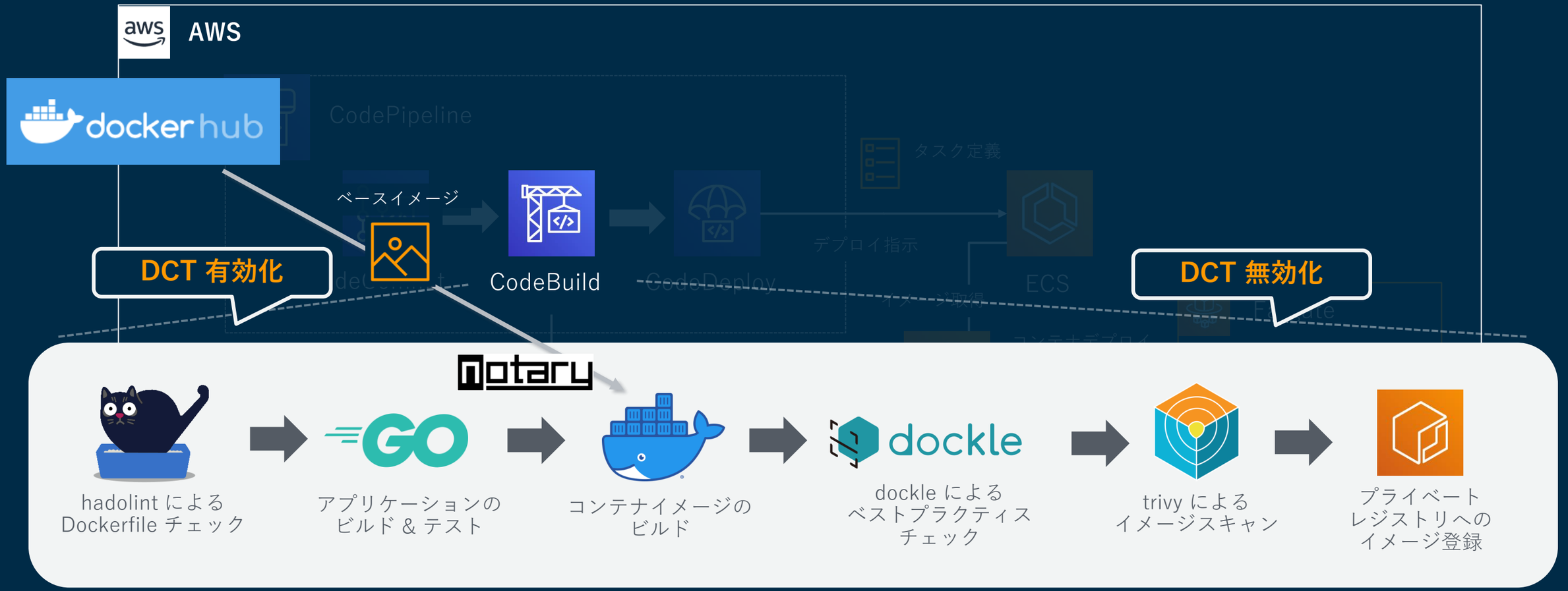
※参考

https://github.com/aws/containers-roadmap/projects/1

In Partnership with intel

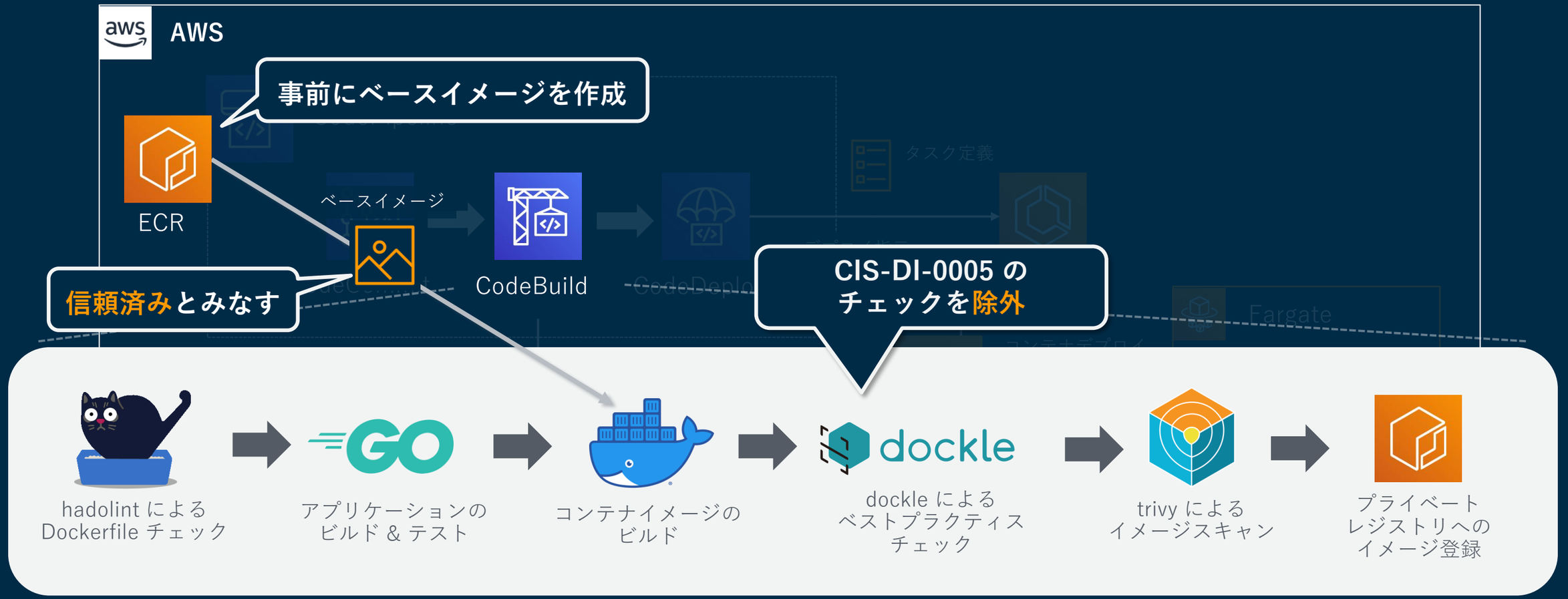
代替その１：pull の時だけ DCT を有効にする

ECR に push する前にDCTを無効化することでエラーを回避できる



代替その２：DCT は利用せずに事前に信頼済みのベースイメージを作成する

個別のベースイメージを事前検証しておく



NIST SP800-190 のその他イメージリスクにどう立ち向かうか？

再掲

その他の 1.3～1.5 は他の AWS サービスとの連携や個別対応で満たしていかなければならない。

項番	項目	リスク内容	対策
1.1	イメージの脆弱性	イメージ内コンポーネントのセキュリティアップデート漏れ等により、脆弱性を含んでいる	脆弱性を排除する、かつ脆弱性を含んだイメージをデプロイしないようにする
1.2	イメージの設定不備	イメージの設定が不適切である 例) root ユーザによる実行	ベストプラクティスに従ったイメージ作成を行う
1.3	埋め込まれたマルウェア	悪意のあるファイルがイメージに含まれている	信頼されたイメージやレジストリを利用する
1.4	埋め込まれた平文の秘密情報	イメージファイル内に予め秘密情報（パスワードや暗号鍵）が含まれてしまうセキュリティリスクが生じる	秘密情報はイメージの外部で保存する
1.5	信頼できないイメージの使用	外部の信頼できないイメージを利用してしまう	信頼されたイメージやレジストリを利用する

CI/CD と OSS を活用することで対応可能

ベースイメージ検証による代替

Secrets Manager 等のサービスと連携することで対応可能

ベースイメージ検証による代替

※コンテナイメージに関する内容を一部引用

https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-190.pdf

aws DEV DAY
ONLINE JAPAN

まとめ

In Partnership with intel

まとめ

✓ DevSecOps は組織の文化、プラクティス、ツールにセキュリティを協調させたもの

✓ OSS を CI/CD に組み合わせることでセキュアなコンテナを運用できる

NIST SP800-190 と照らし合わせた対応が可能

✓ 仕組み化を文化醸成への架け橋に

最終的には、ツールやプラクティスから組織の文化を醸成していくことで、サイクルとして成立する

aws DEV DAY
ONLINE JAPAN

Thank you!

Masaya ARAI
@msy78

In Partnership with intel