Google Cloud

モダンな開発を維持して
ソフトウェア サプライチェーンを守る
Software Delivery Shield のご紹介

Google Cloud
カスタマー エンジニア
中丸 良

Table of Contents

01

モダンな開発を振り返る

ローカル開発からデプロイまで

1. ローカル開発

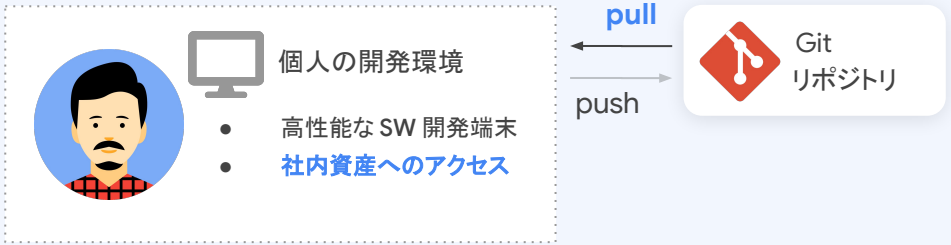
社内ネットワーク

個人の開発環境

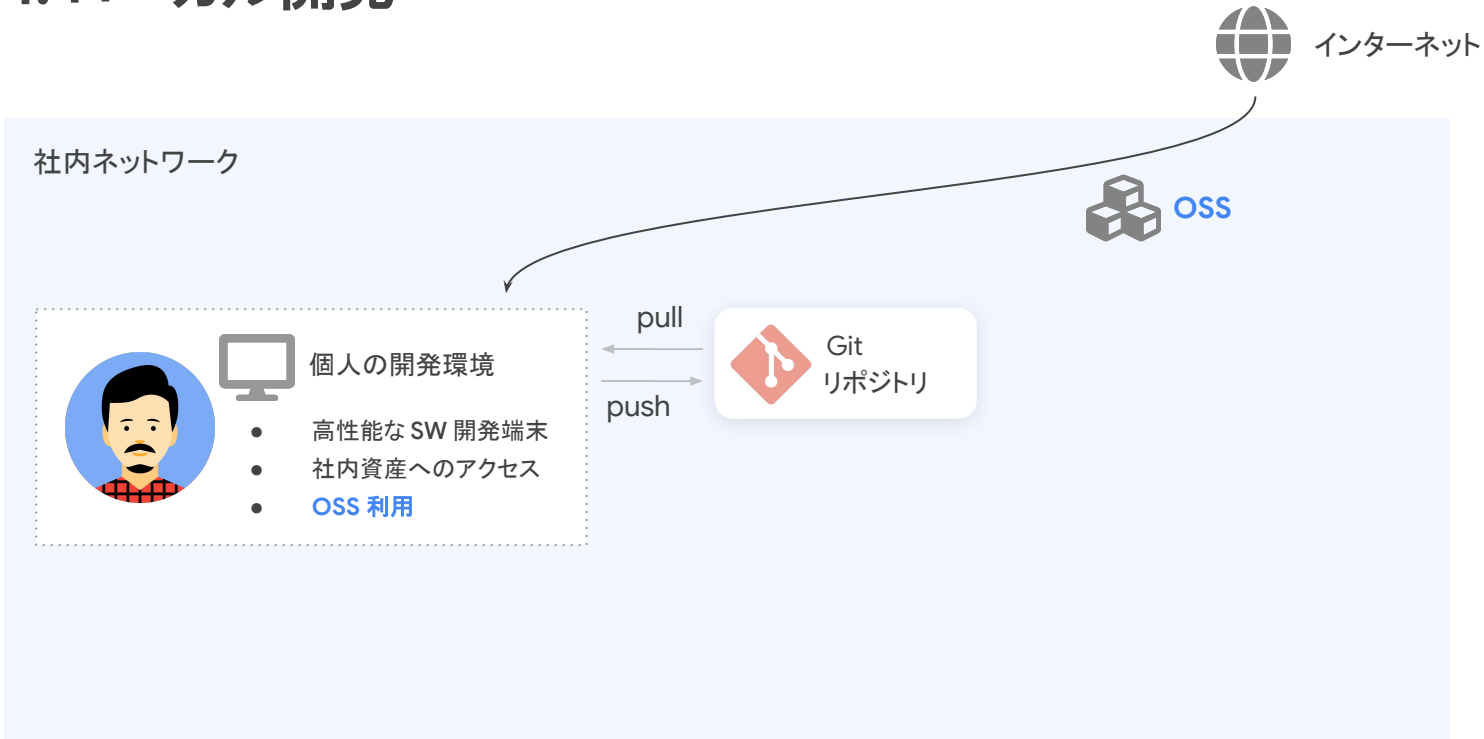
- 高性能な SW 開発端末

1. ローカル開発

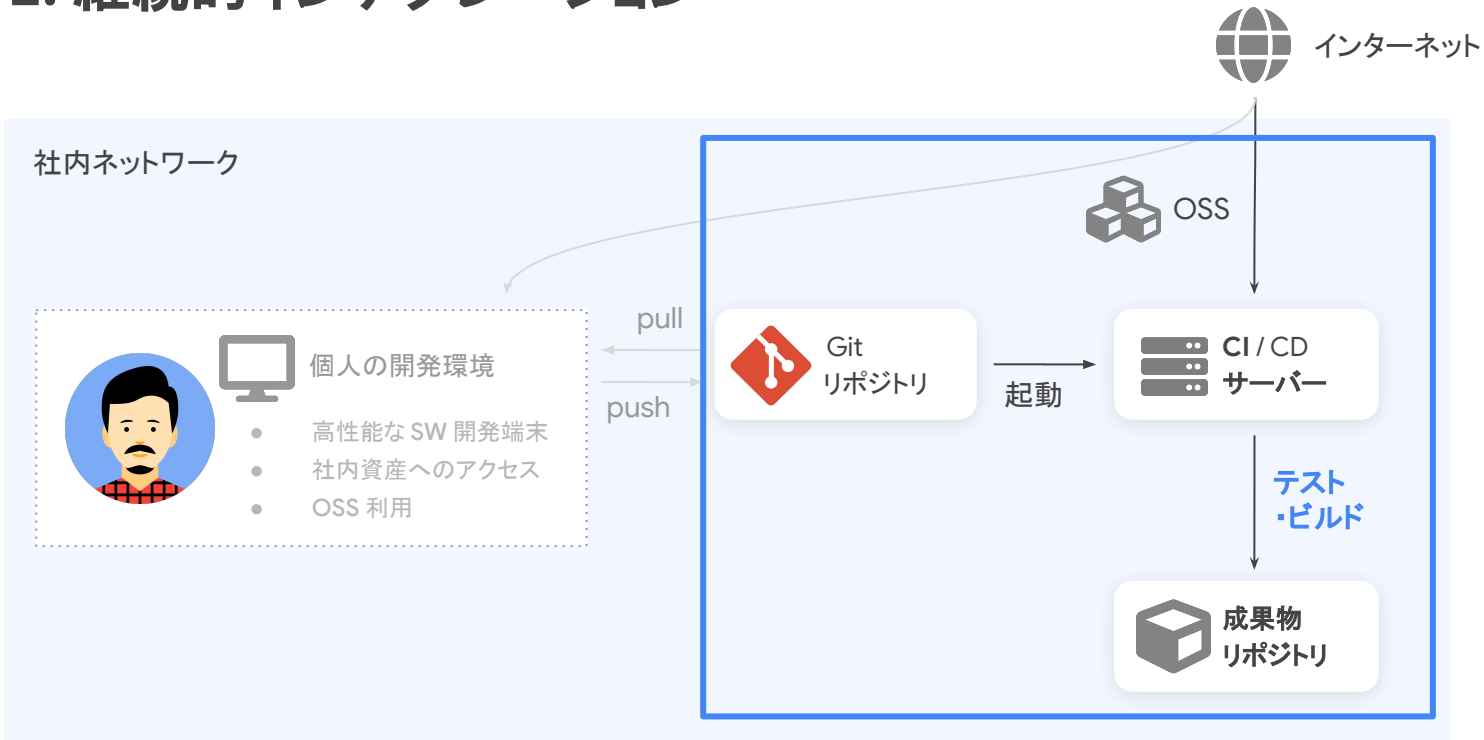
社内ネットワーク



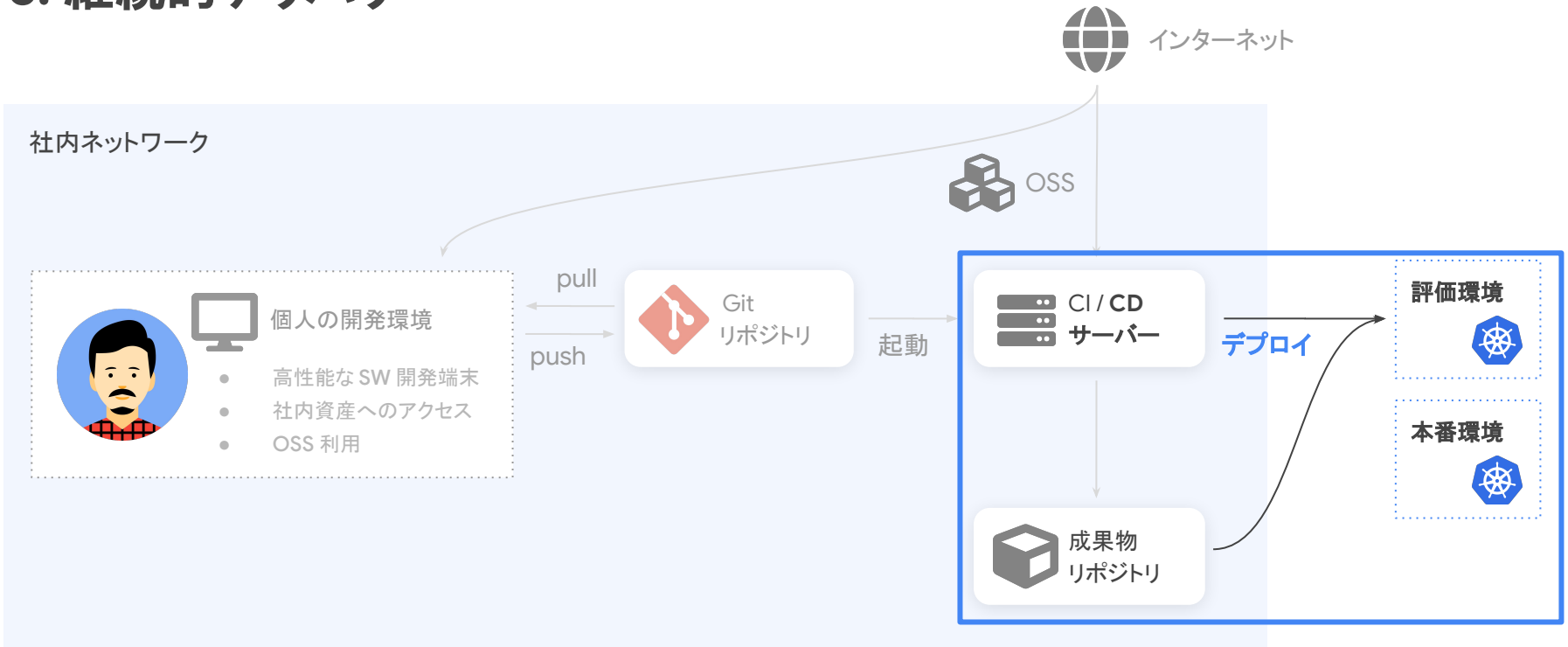
1. ローカル開発


インターネット
社内ネットワーク
OSS
pull
push
Git
リポジトリ
個人の開発環境
高性能な SW 開発端末
社内資産へのアクセス
OSS 利用

2. 継続的インテグレーション



3. 継続的デリバリー


インターネット
社内ネットワーク
OSS
個人の開発環境
高性能な SW 開発端末
社内資産へのアクセス
OSS 利用
pull
push
Git
リポジトリ
起動
CI / CD
サーバー
デプロイ
評価環境
本番環境
成果物
リポジトリ

02

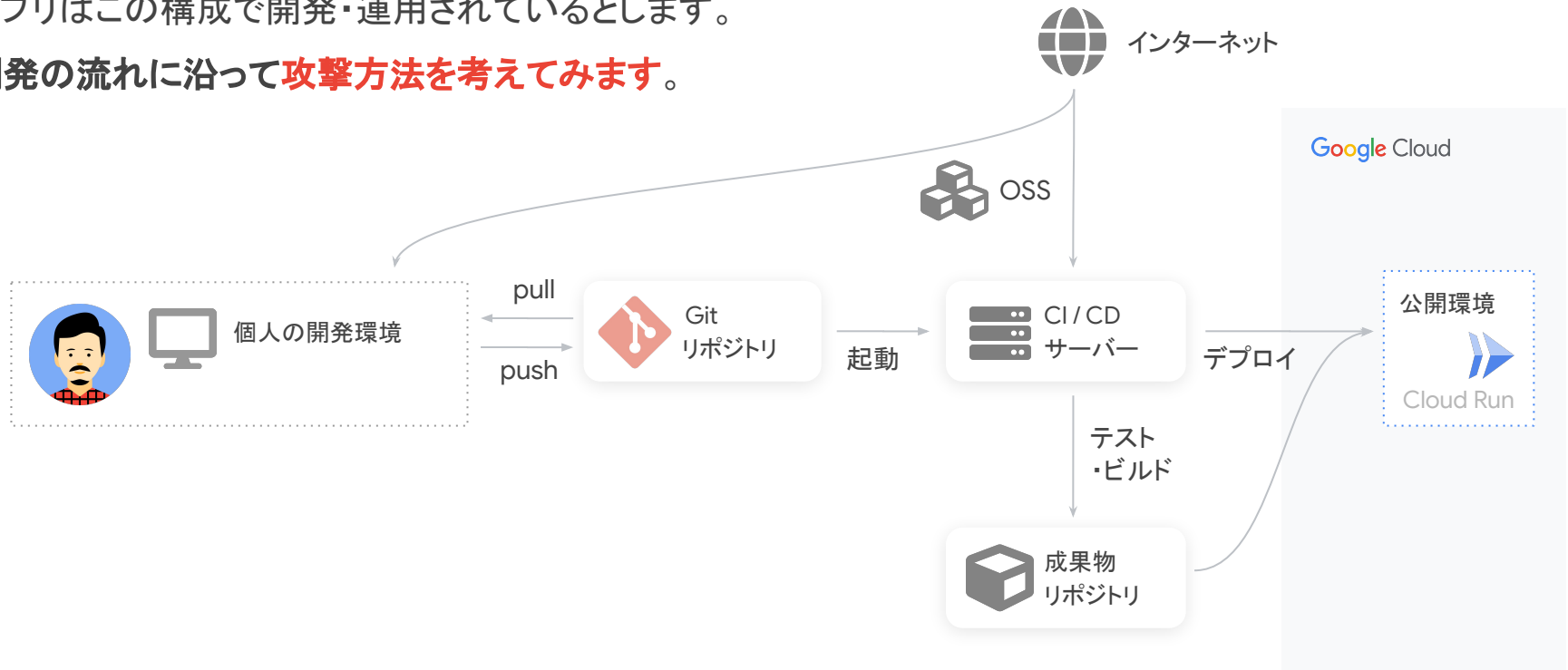
ソフトウェアに対する攻守の例

攻撃者になって 5 つのシーンを考察しよう

さあ、みなさんは攻撃者です。どう攻撃できるか、一緒に考えましょう！

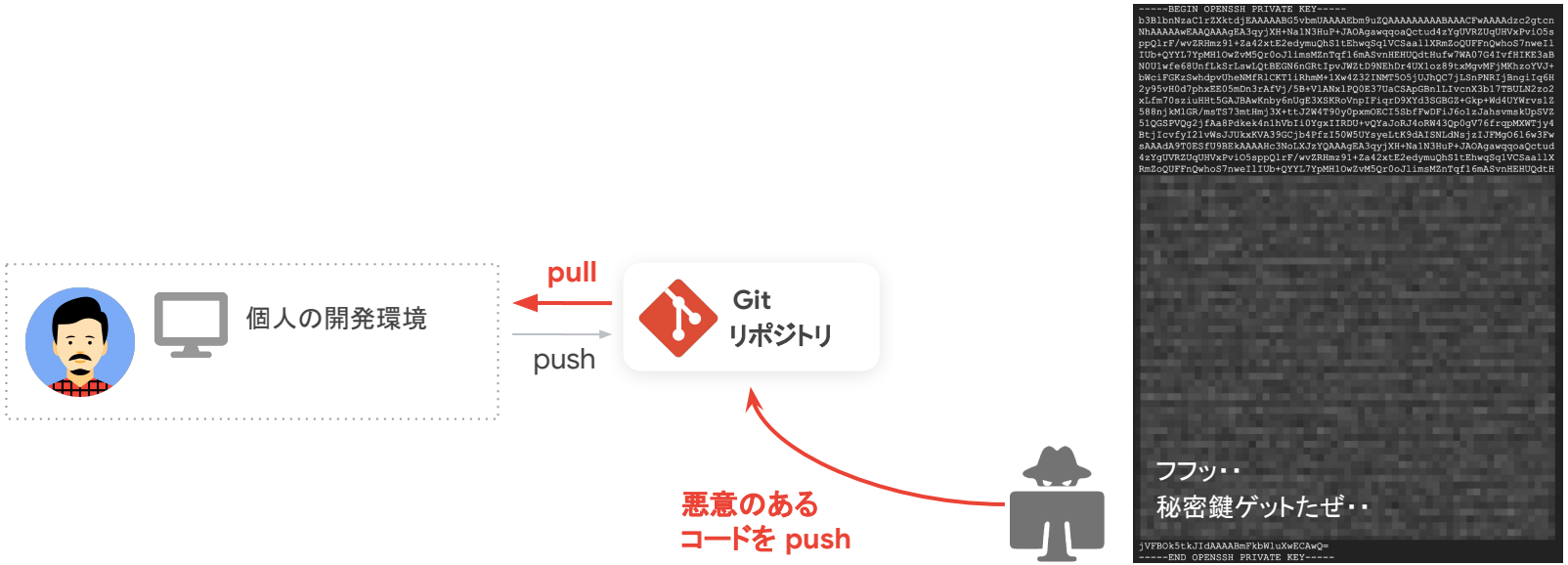
アプリはこの構成で開発・運用されているとします。

開発の流れに沿って攻撃方法を考えてみます。

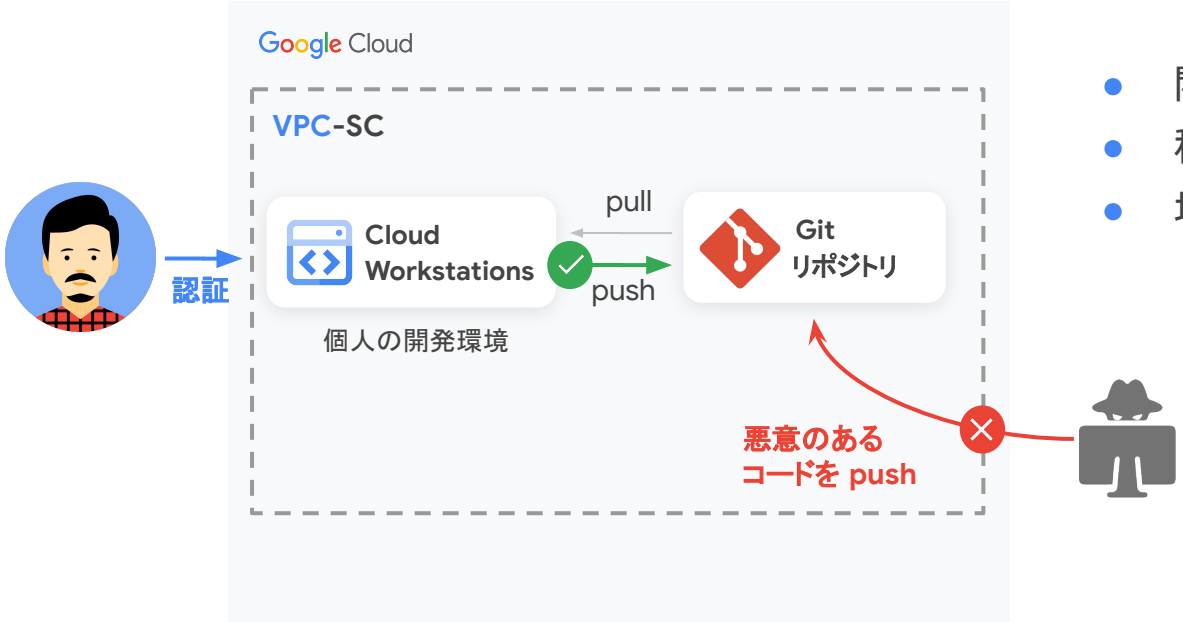


1. ソースコードへの攻撃

1. ソースコードへの攻撃



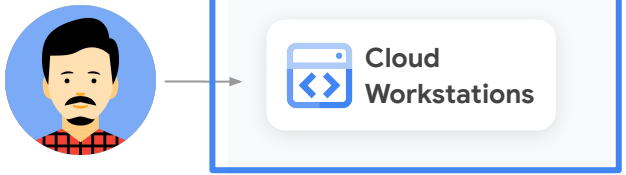
1. ソースコードへの攻撃に対する防御例



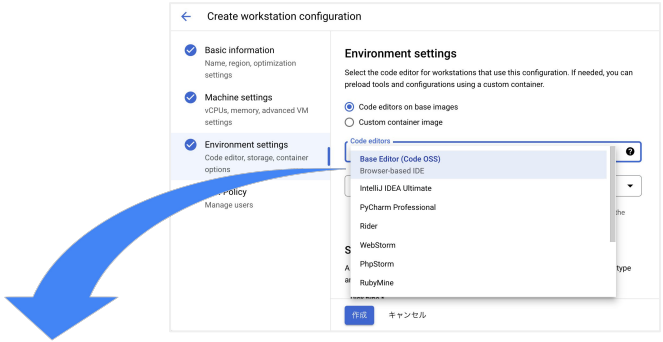
- 開発環境へ接続に認証を要求
- 秘密鍵などをより安全に管理
- 境界防御でのリスク軽減

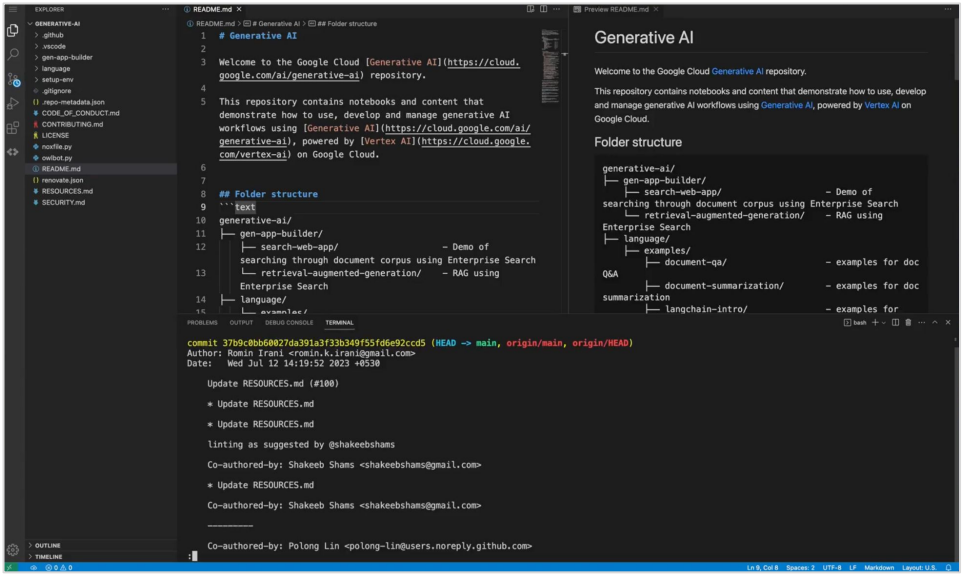
Cloud Workstations

Google Cloud

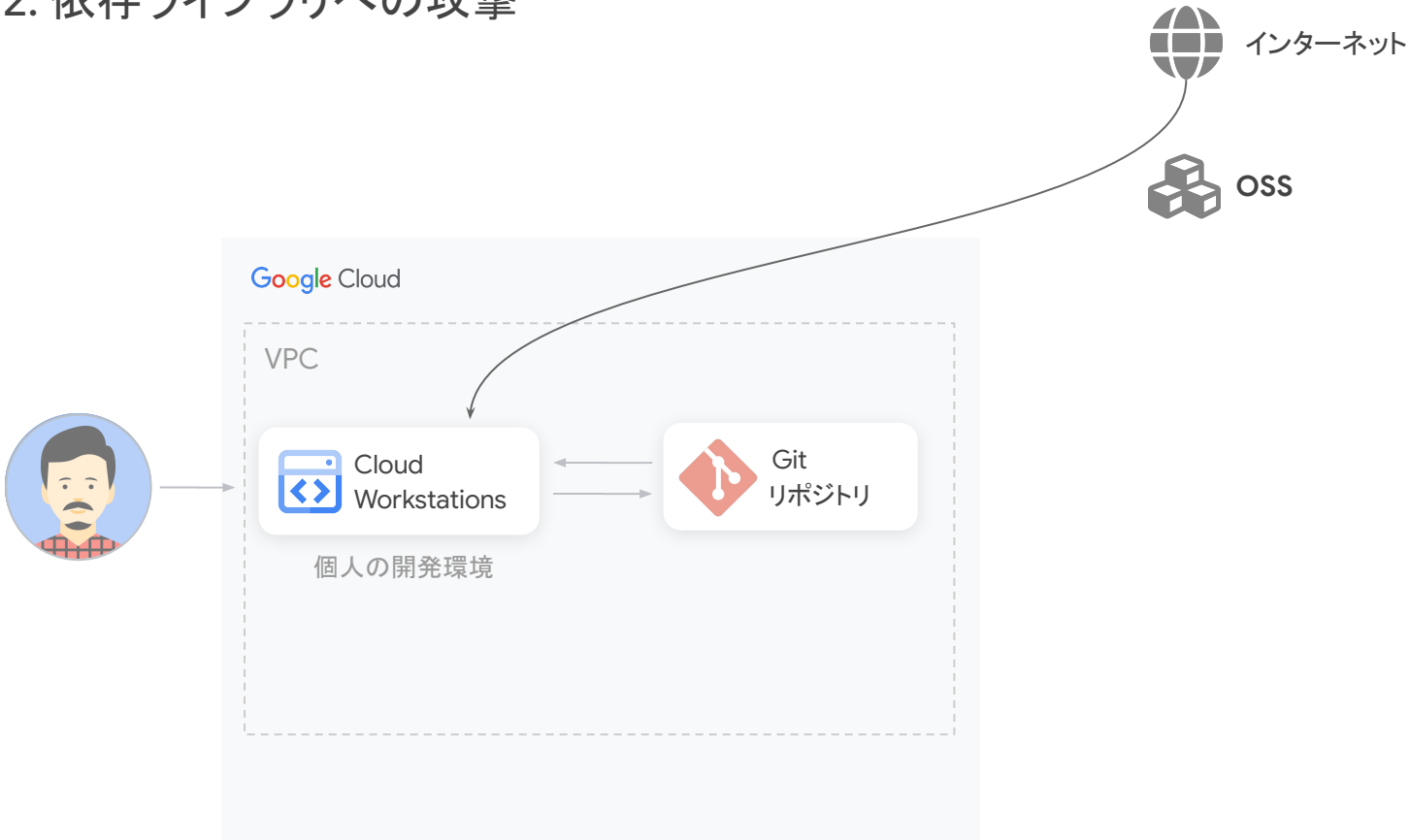


- **クラウド上、VPC 内における開発環境**
- Code-OSS や IntelliJ IDEA などに対応
- コンテナによる環境の管理・配布



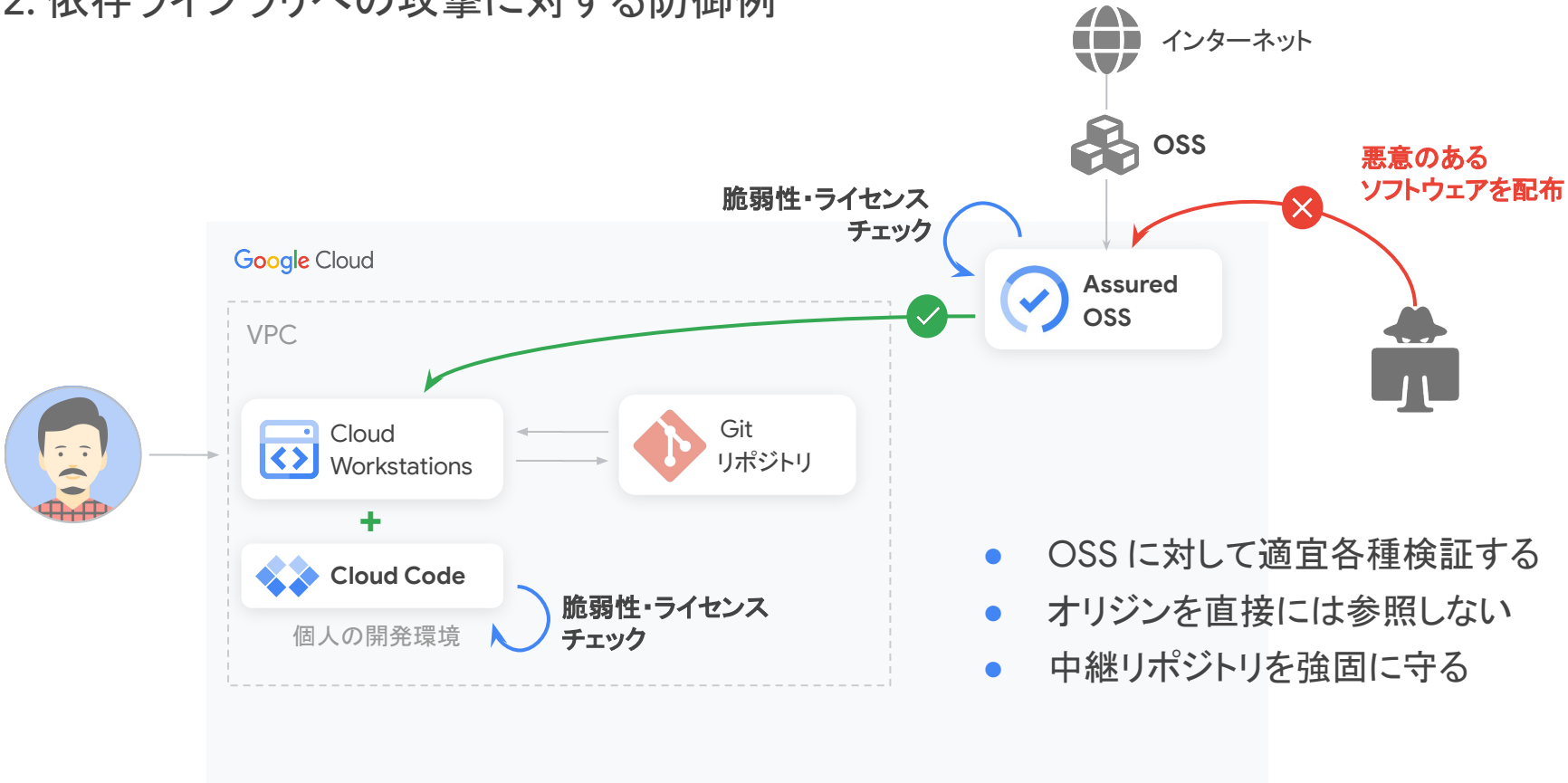


2. 依存ライブラリへの攻撃



2. 依存ライブラリへの攻撃

2. 依存ライブラリへの攻撃に対する防御例



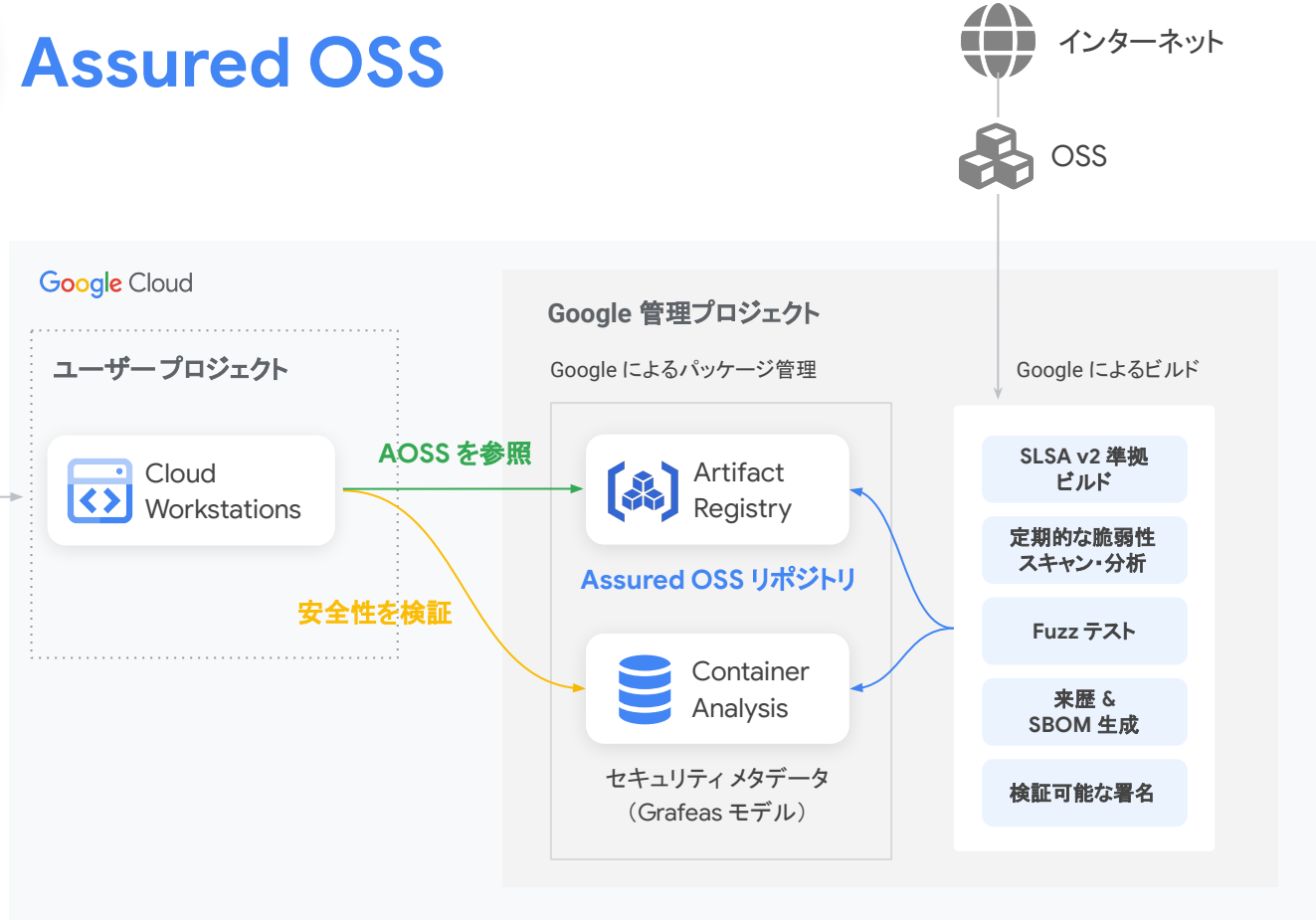
- OSS に対して適宜各種検証する
- オリジンを直接には参照しない
- 中継リポジトリを強固に守る

Assured OSS

- **Google が社内で検証し、実際に製品で利用している OSS**
- 250 を超える Java と Python のパッケージ
- バージョン管理されたパッケージ + 署名された来歴つき

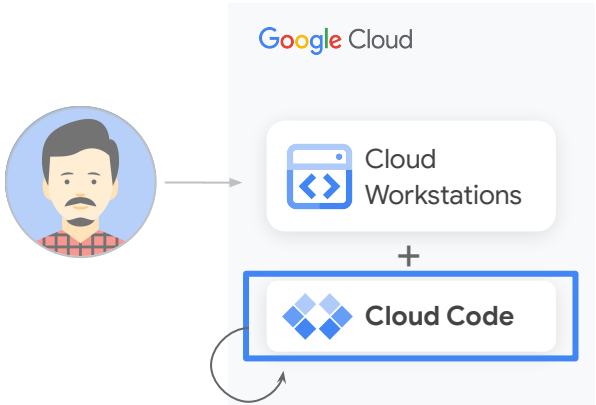
https://cloud.google.com/assured-open-source-software?hl=ja

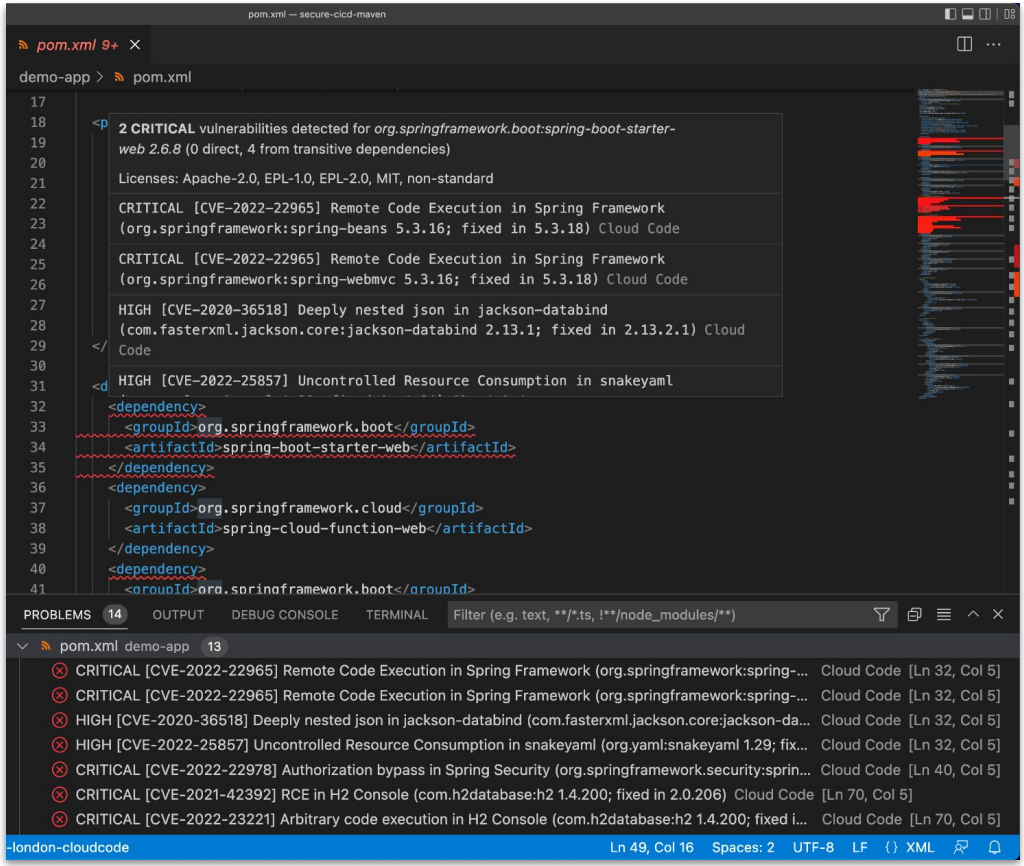
Assured OSS


インターネット
OSS
Google Cloud
Google 管理プロジェクト
ユーザープロジェクト
Google によるパッケージ管理
Google によるビルド
Cloud Workstations
AOSS を参照
Artifact Registry
Assured OSS リポジトリ
安全性を検証
Container Analysis
セキュリティ メタデータ
（Grafeas モデル）
SLSA v2 準拠 ビルド
定期的な脆弱性 スキャン・分析
Fuzz テスト
来歴 & SBOM 生成
検証可能な署名

Cloud Code

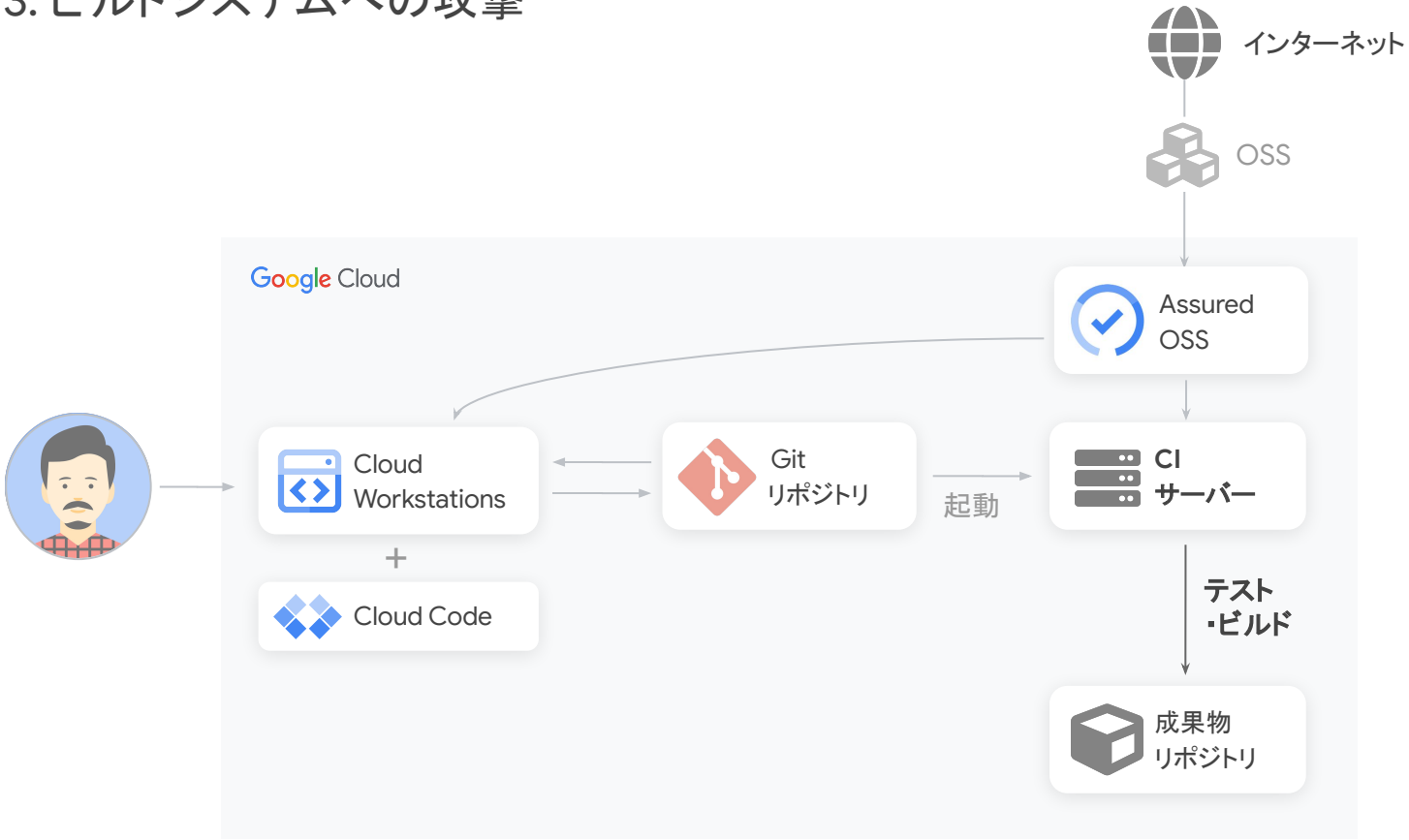
IDE プラグイン！ローカルでも使えます



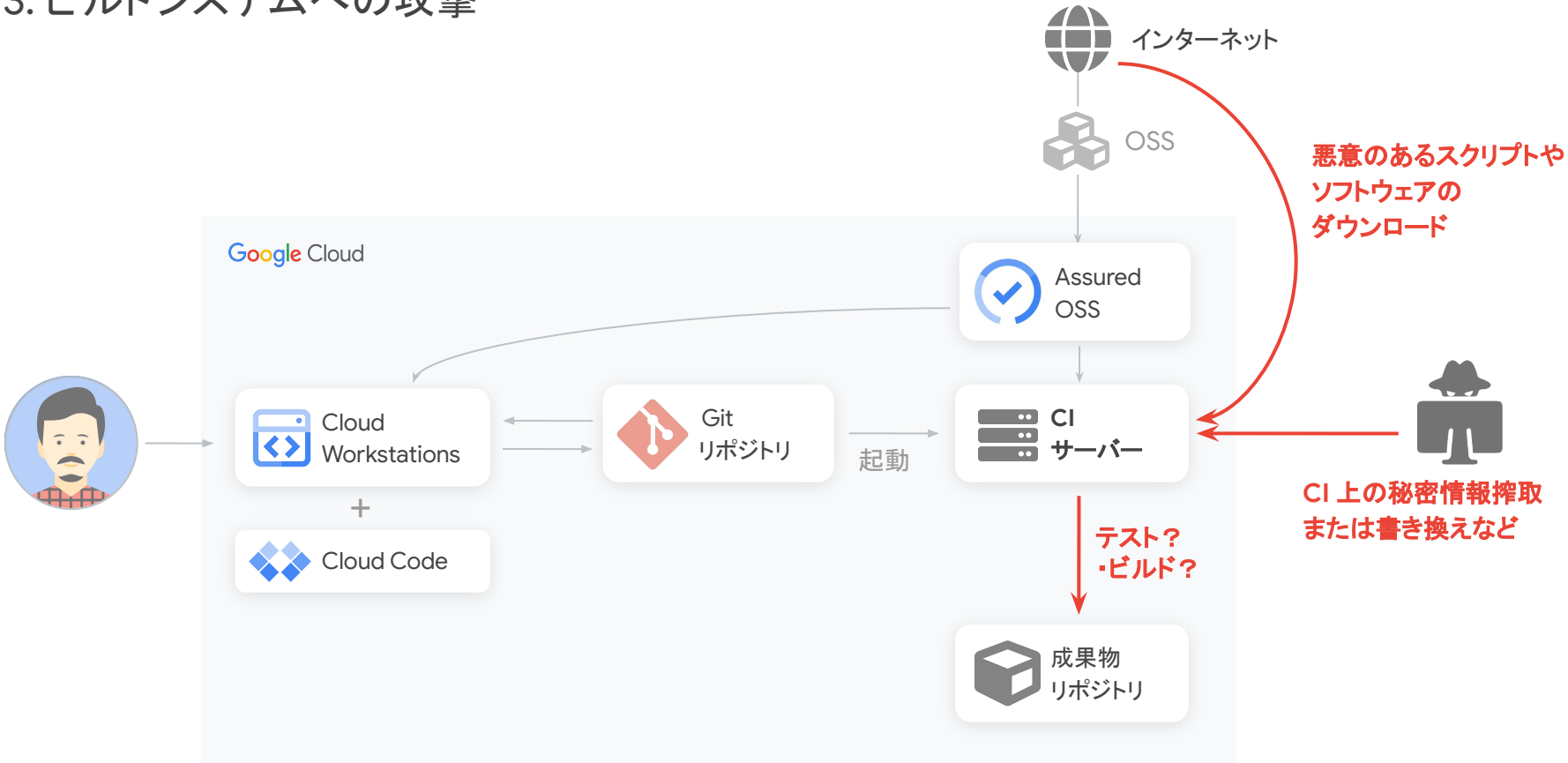


https://cloud.google.com/code?hl=ja

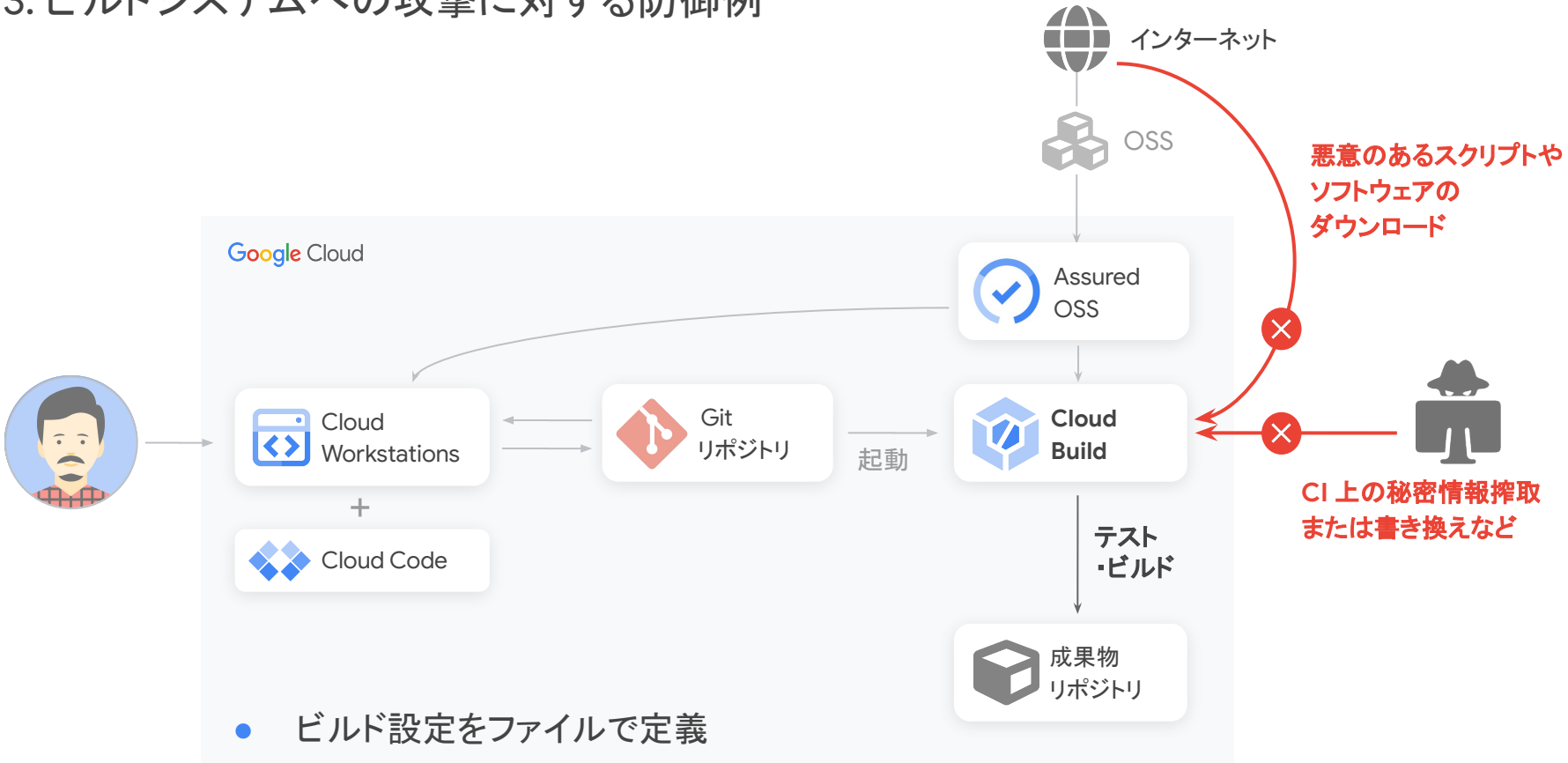
3. ビルドシステムへの攻撃



3. ビルドシステムへの攻撃



3. ビルドシステムへの攻撃に対する防御例



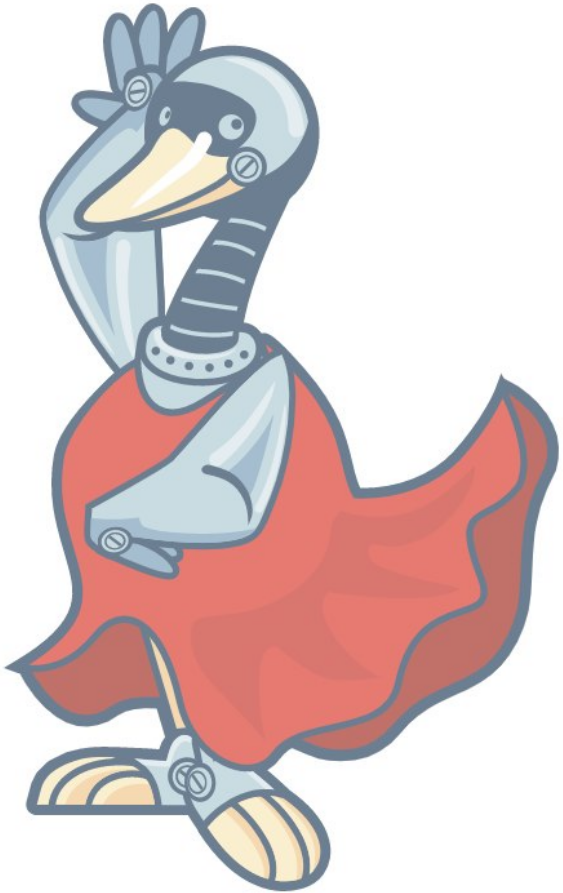
- ビルド設定をファイルで定義
- 分離され、毎回**クリーンなビルド環境**

SLSA

ソフトウェアを開発し、ユーザーに届けるまでの
整合性に関する基準を形式化したもの

SLSA は 4 つのレベルで構成され、
Level 4 であれば 利用者はソフトウェアが改ざんされておらず
ソースまで安全に追跡できるという確信 が持てます。

- **Level 1**
  ビルド プロセスの完全な自動化 & 来歴の生成
- **Level 2**
  バージョン管理 & 署名された来歴
- **Level 3**
  ソースとビルド基盤が基準を満たし、監査可能性 & 来歴の保証
- **Level 4**
  すべての変更で 2 名体制レビュー & 閉域での再現可能なビルド

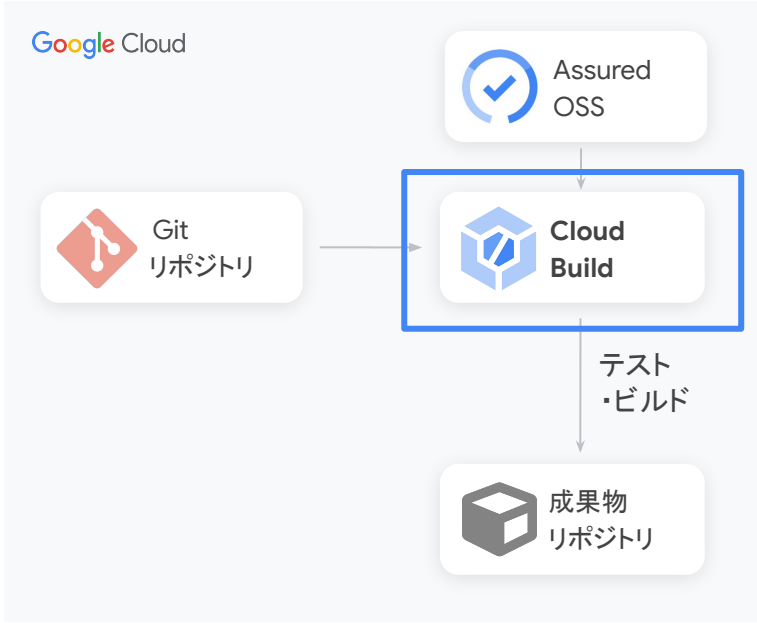


https://slsa.dev/

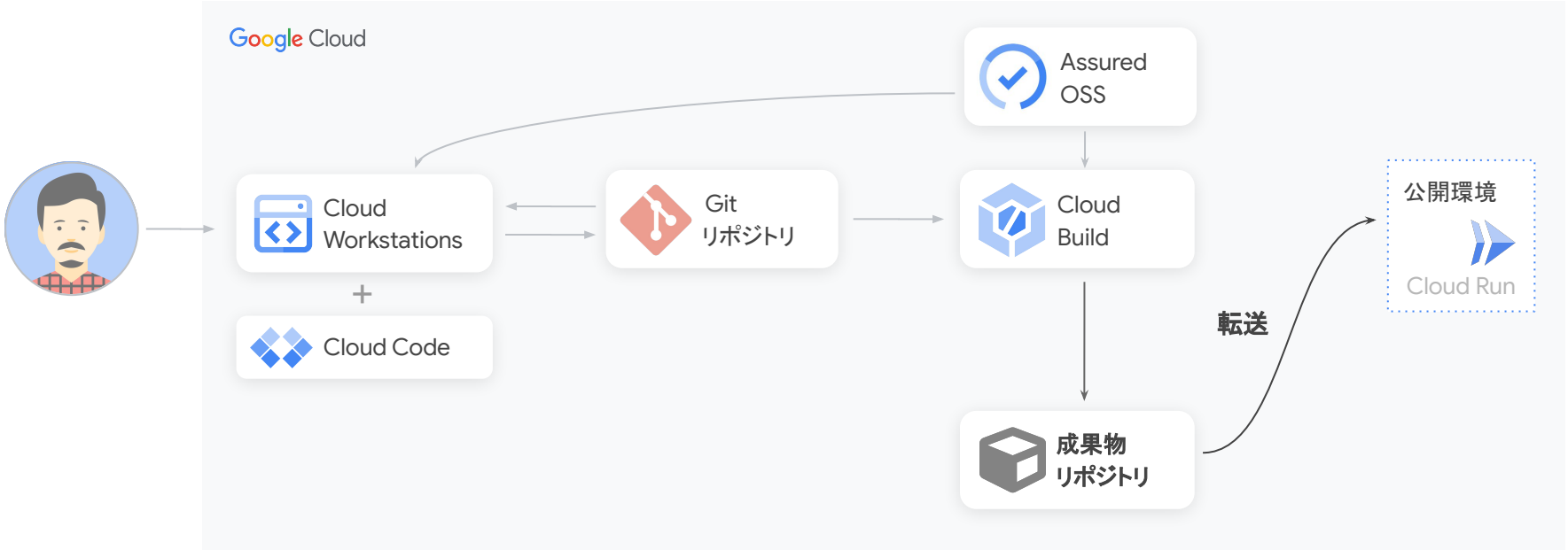
Cloud Build

フルマネージドな
CI / CD プラットフォーム

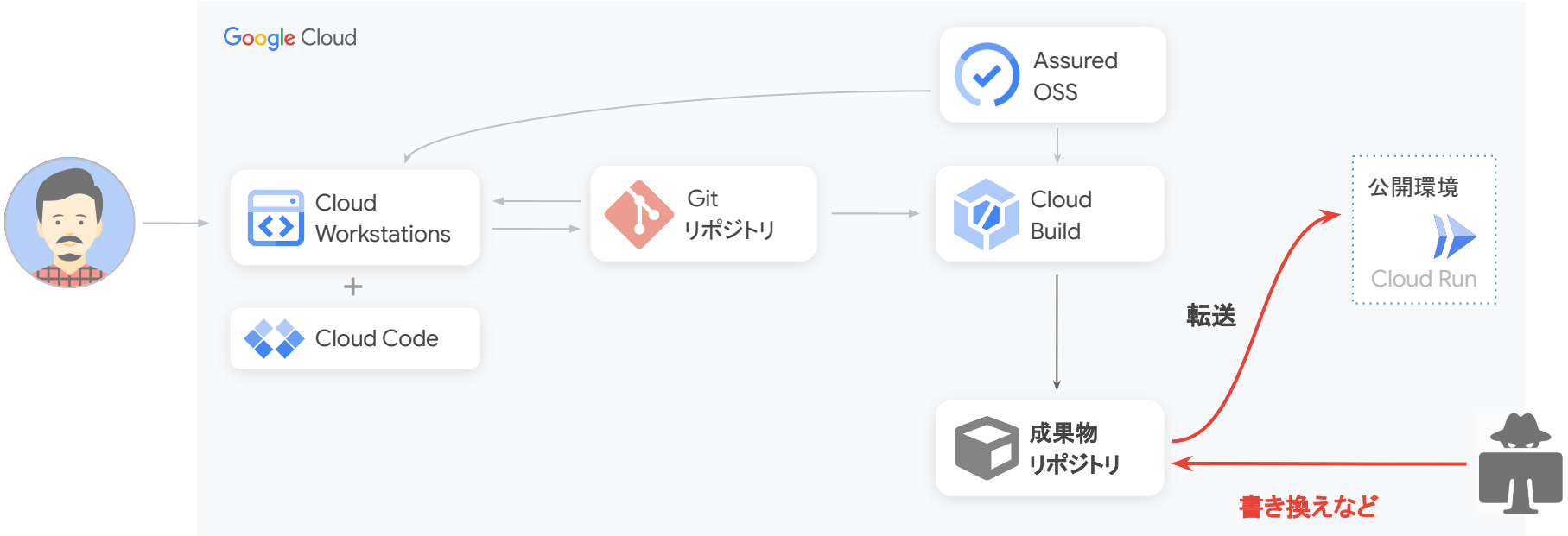
- **コンテナ化アプリケーション、Maven および Python パッケージの改ざんできないビルド来歴の生成**
- セキュリティ分析情報の表示
  - **SLSA レベル**
  - 脆弱性
  - 来歴など



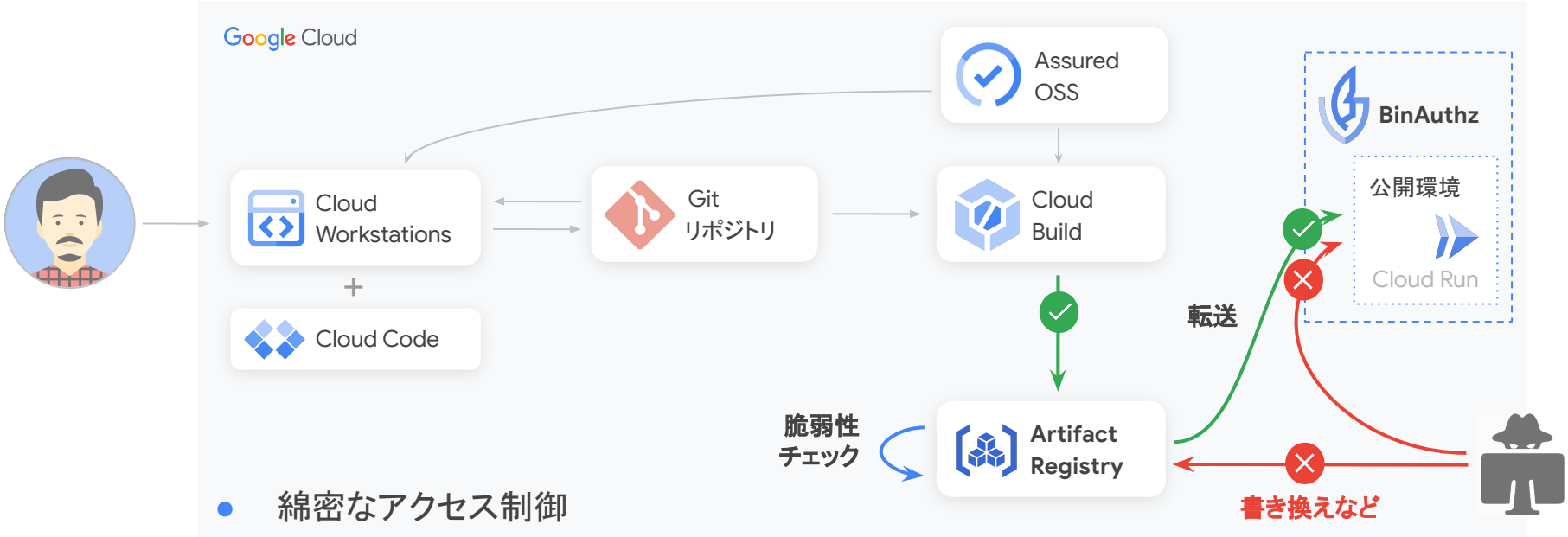
4. 成果物への攻撃


Google Cloud
Assured OSS
Cloud Workstations
+
Cloud Code
Git リポジトリ
Cloud Build
公開環境
Cloud Run
転送
成果物 リポジトリ

4. 成果物への攻撃


Google Cloud
Assured OSS
Cloud Workstations
+
Cloud Code
Git リポジトリ
Cloud Build
公開環境
Cloud Run
転送
成果物 リポジトリ
書き換えなど

4. 成果物への攻撃に対する防御例

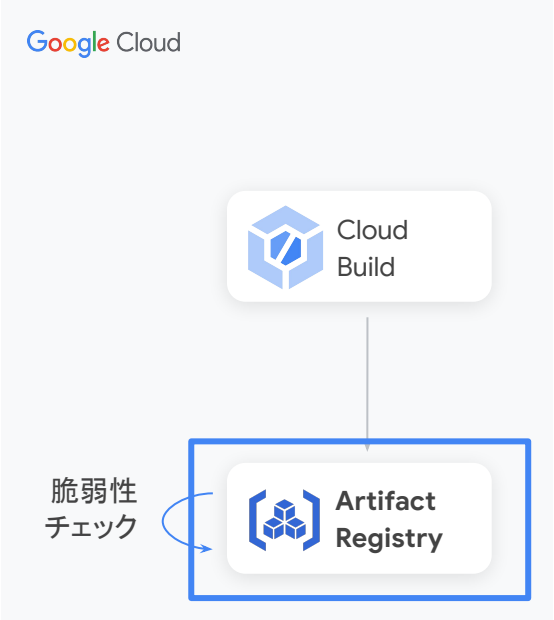


- 綿密なアクセス制御
- **電子署名**や**許可リスト**によるワークロード保護

Artifact Registry

フルマネージドな
成果物の保存、管理、保護サービス

- Cloud IAM によるアクセス制御
- コンテナ、Maven、Go の**脆弱性スキャン**
- ソフトウェア部品表 (SBOM) の生成

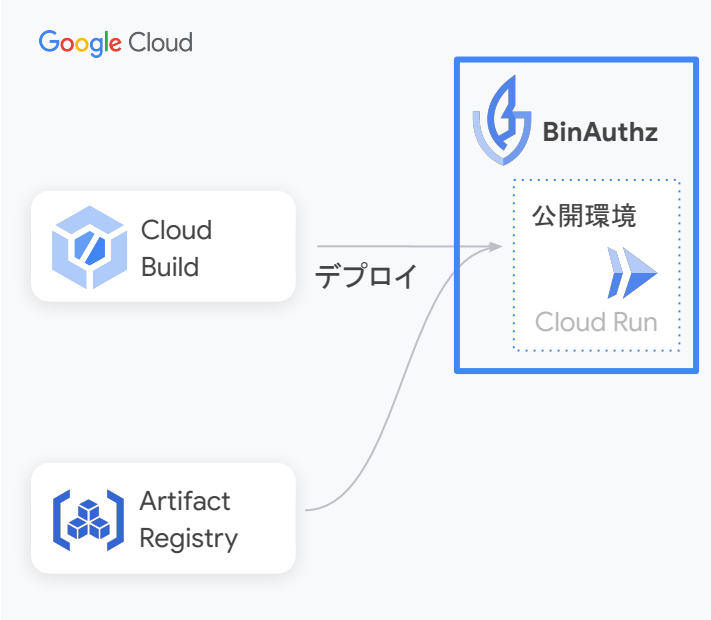


https://cloud.google.com/artifact-registry?hl=ja

Binary Authorization

信頼できるワークロードのみを
デプロイするための仕組み

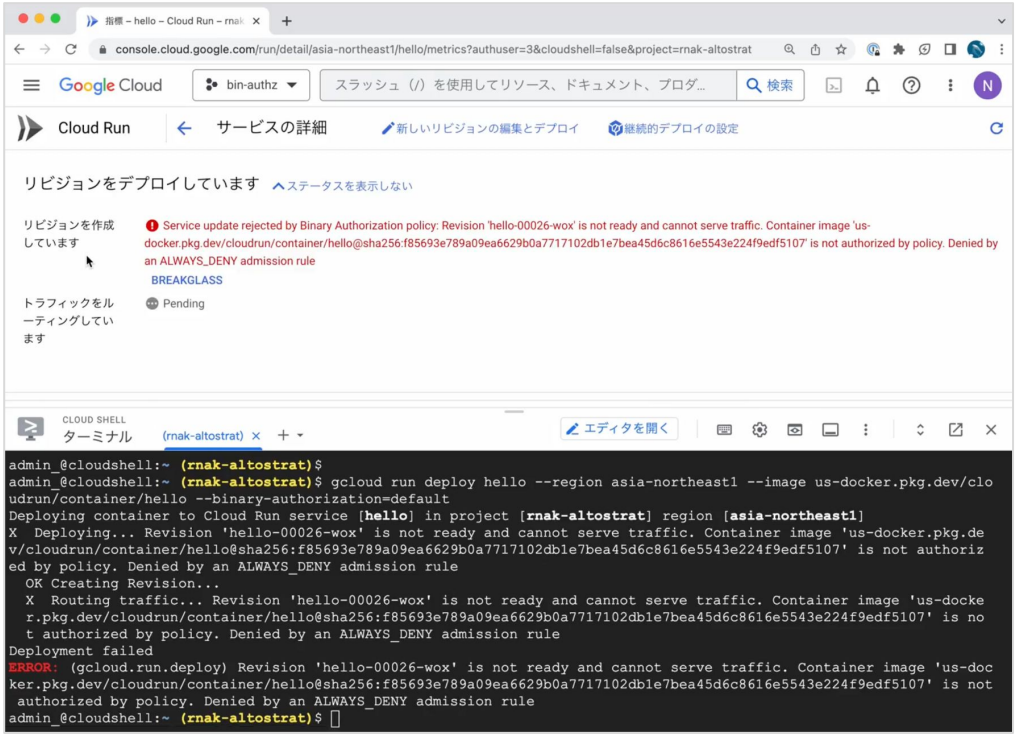
- 意図しないコードや悪意のあるコードが実行されるリスクを低減
- リポジトリは**許可リスト**にある？
- イメージに**信頼できる署名**はされている？



https://cloud.google.com/binary-authorization?hl=ja

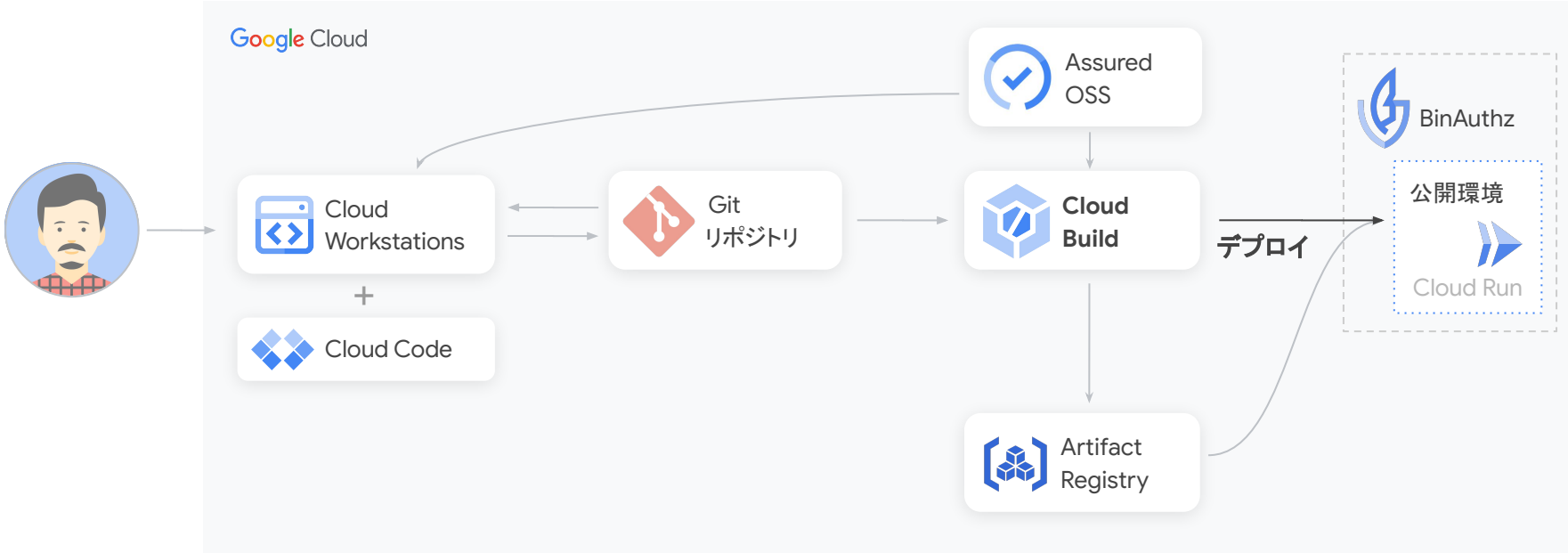
Binary Authorization

```
$ cat << EOF >policy.yaml
  admissionWhitelistPatterns:
    - namePattern: gcr.io/google_containers/*
    - namePattern: gcr.io/google-containers/*
    - namePattern: k8s.gcr.io/*
    - namePattern: gke.gcr.io/*
    - namePattern: gcr.io/stackdriver-agents/*
    - namePattern: asia-northeast1-docker.pkg.dev/company-common/libs/monitor[@:]*
    - namePattern: asia-northeast1-docker.pkg.dev/project-alpha/some-app/*
  defaultAdmissionRule:
    enforcementMode: ENFORCED_BLOCK_AND_AUDIT_LOG
    evaluationMode: ALWAYS_DENY # or REQUIRE_ATTESTATION
  globalPolicyEvaluationMode: ENABLE
  name: projects/project-alpha/policy
  EOF
```

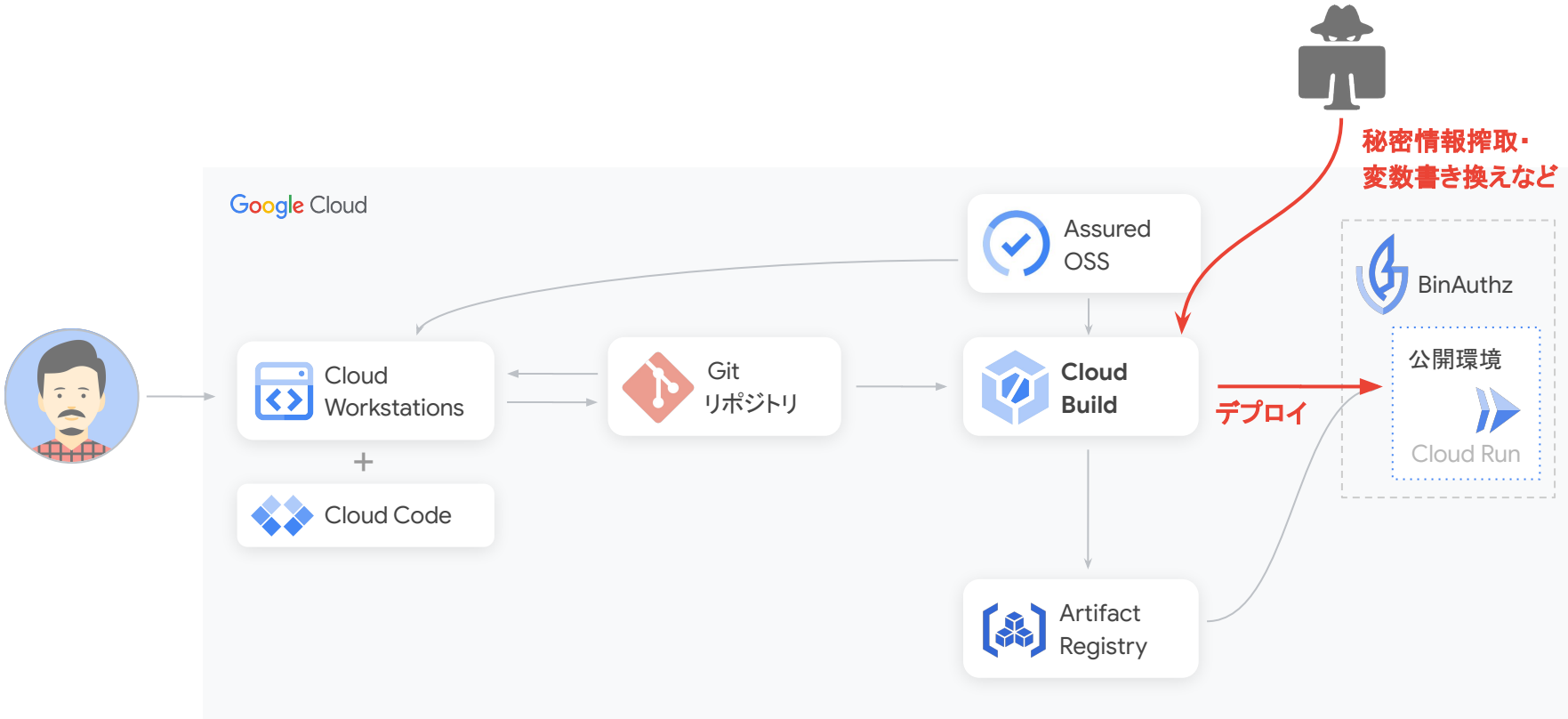


- **事前に定義したポリシーに従って**
- GKE や Cloud Run にデプロイする
  イメージを **許可** / **拒否** します

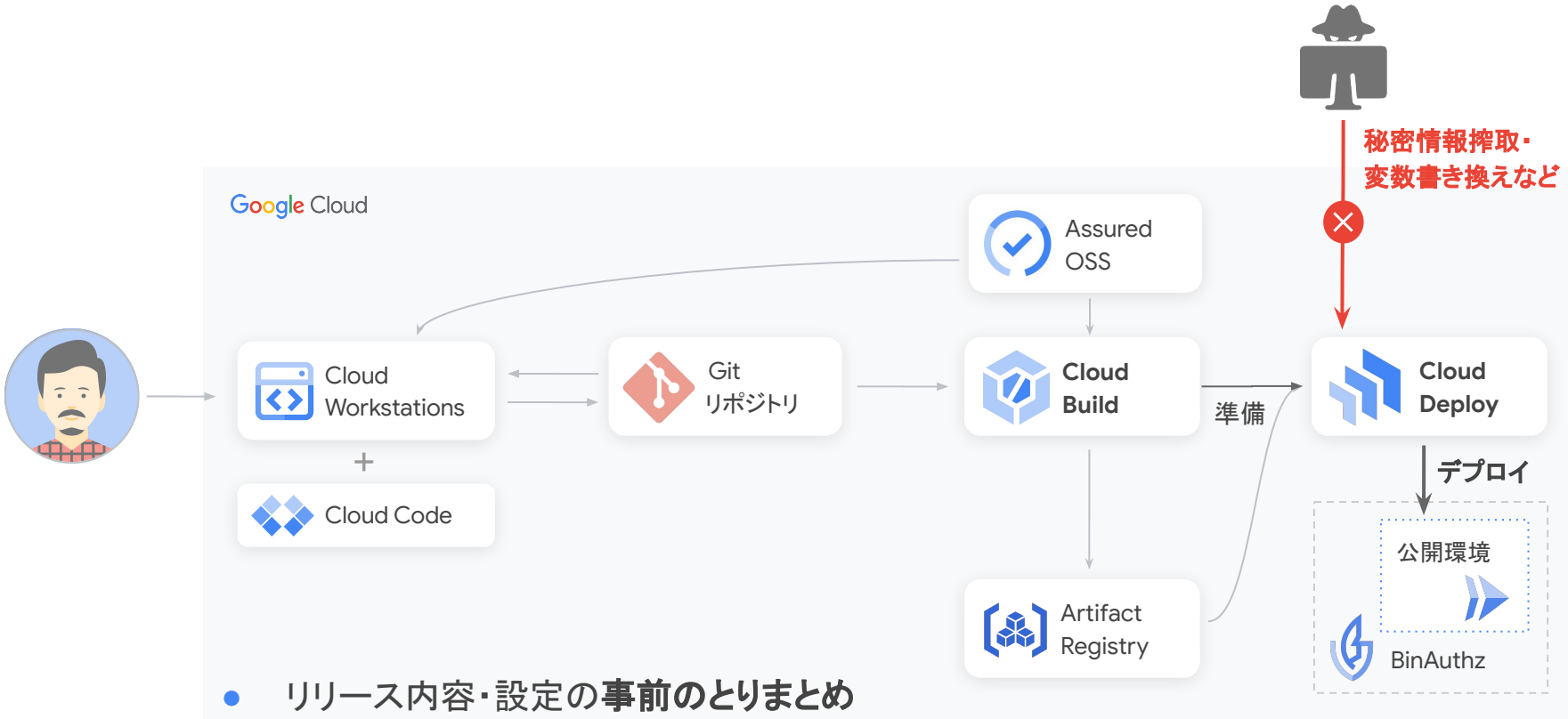
5. デプロイシステムへの攻撃



5. デプロイシステムへの攻撃



5. デプロイシステムへの攻撃に対する防御例

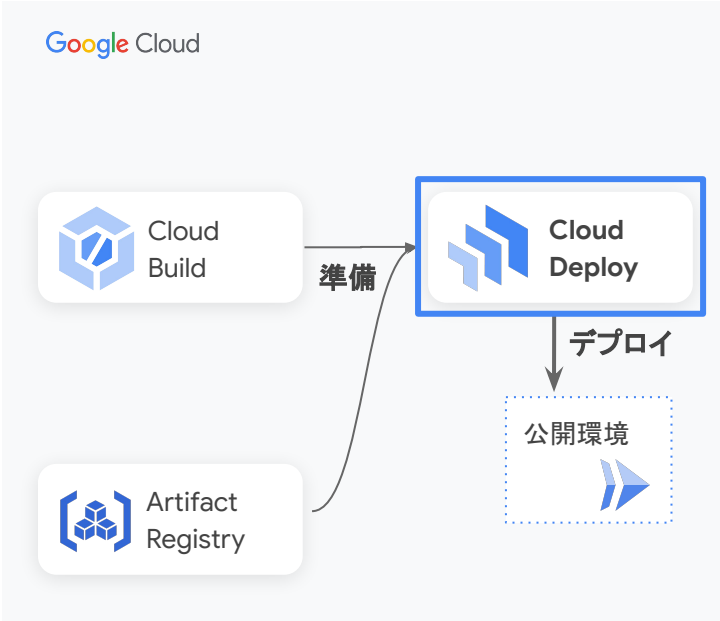


- リリース内容・設定の**事前のとりまとめ**
- CI と CD 分離による最小権限の原則の徹底

Cloud Deploy

フルマネージドな
継続的デリバリーのためのサービス

- Skaffold を活用したオープンな抽象化
- **成果物と設定値を「事前に」とりまとめ**
- 各環境へ順次、モダンな手法でロールアウト

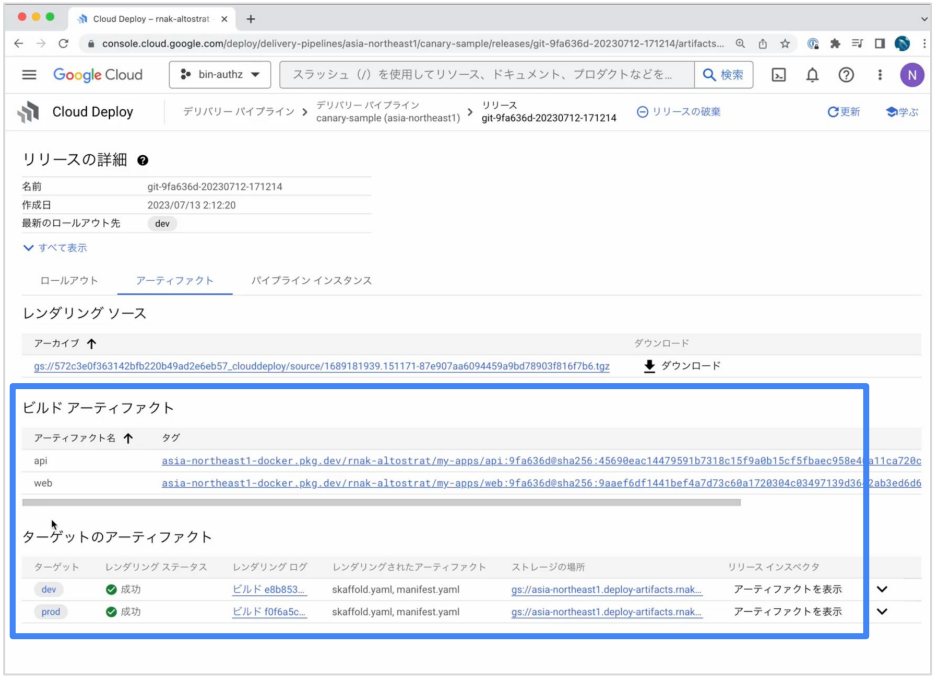


https://cloud.google.com/deploy?hl=ja

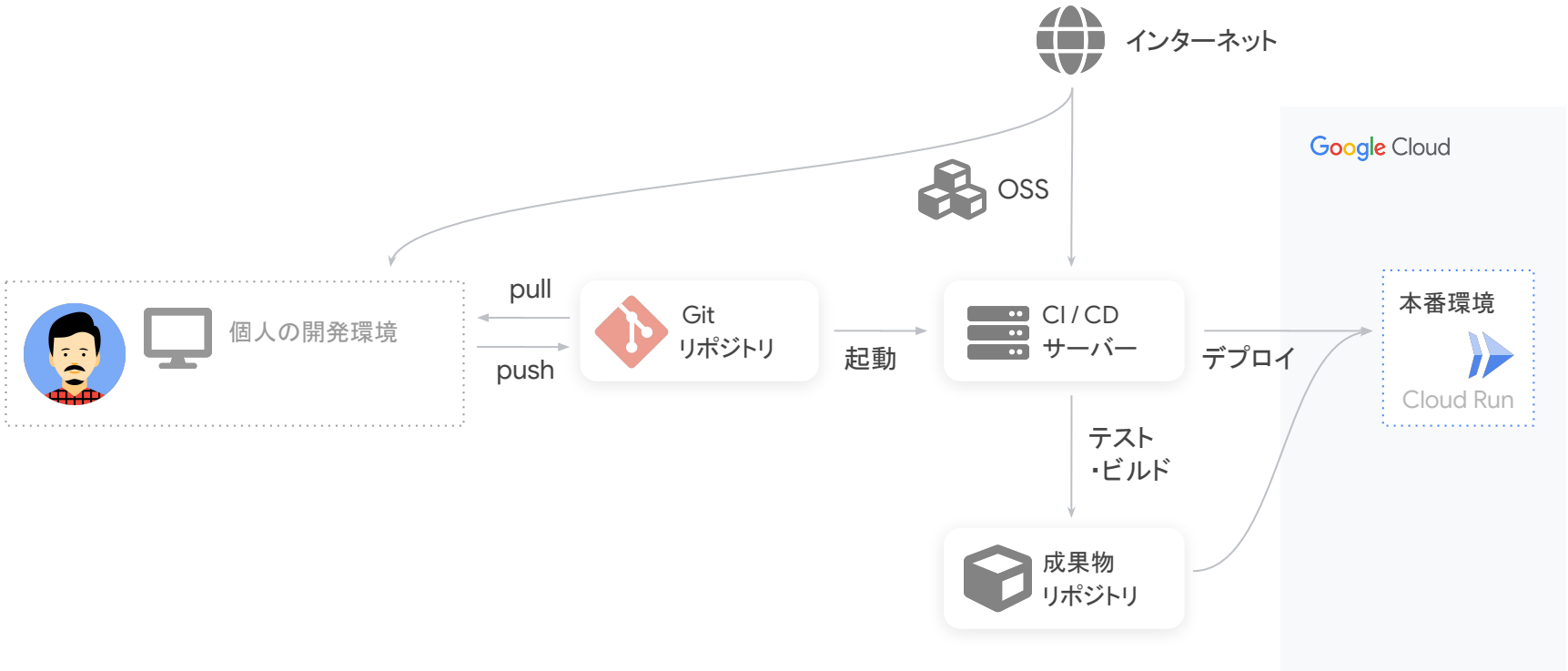
Cloud Deploy

事前にとりまとめる‥とは？

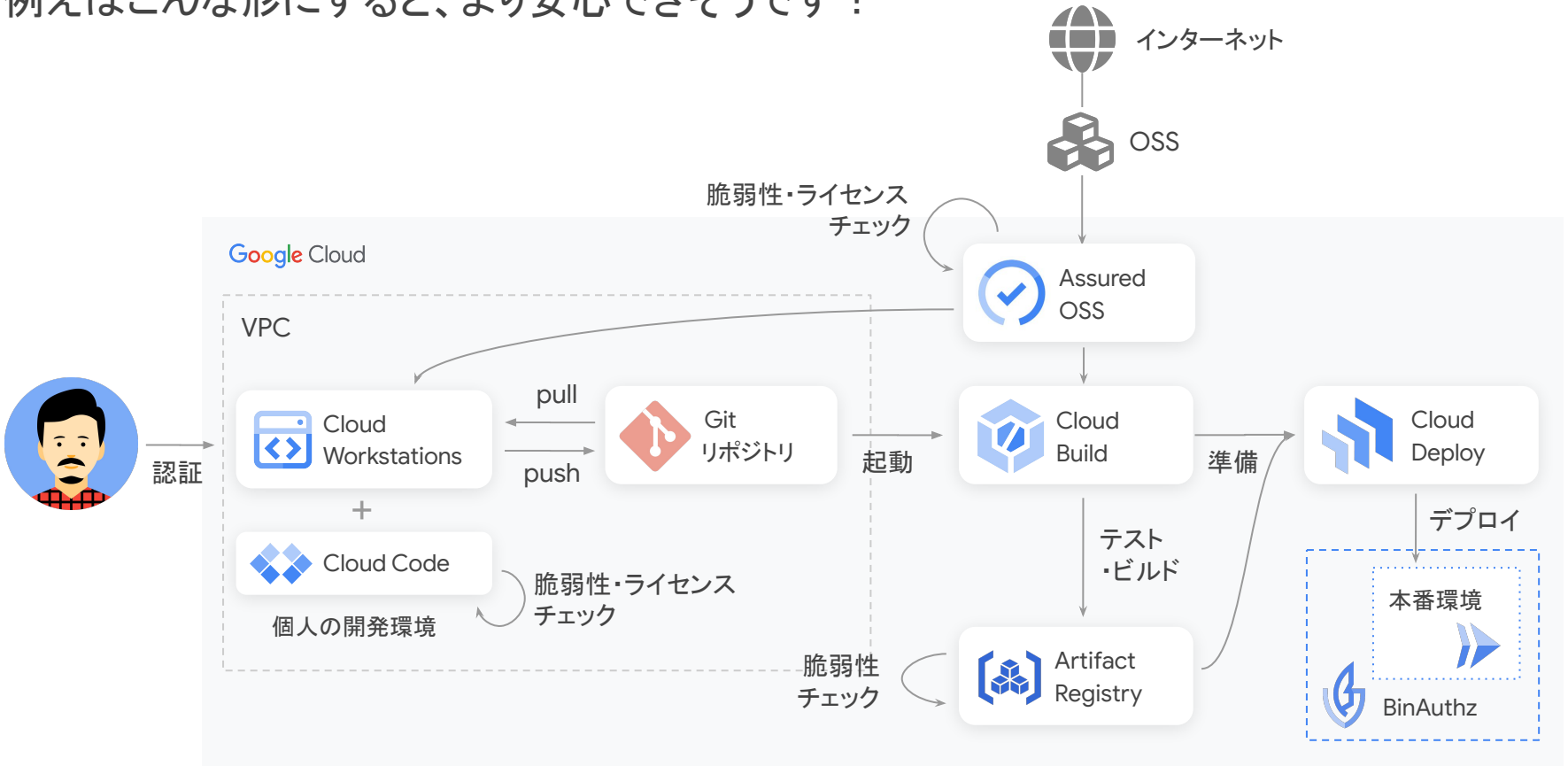
- **一緒にデプロイする成果物**をまとめる
  = ビルド アーティファクト
- **全環境分の設定値**を作成し、保存する
  = ターゲットのアーティファクト



まとめ: これまでこんな形でやってきた開発・運用ですが・・



例えばこんな形にすると、より安心できそうです！



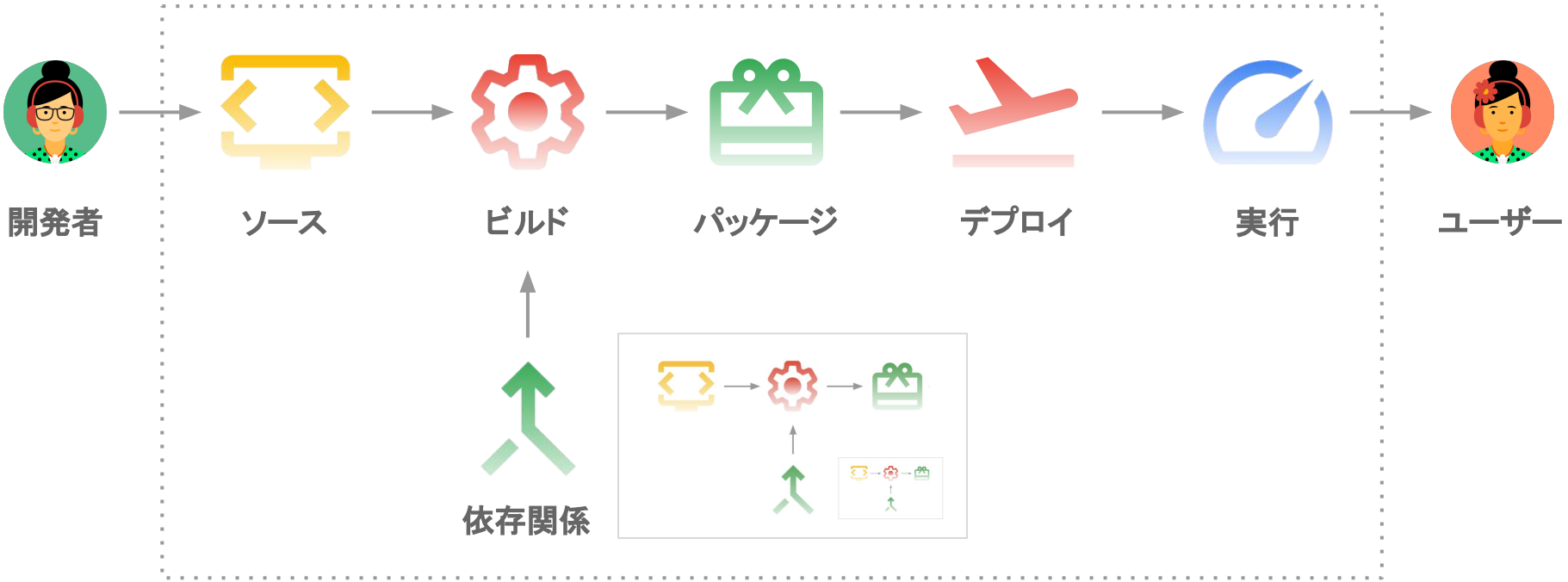
03

ソフトウェア サプライチェーンとは

そしてなぜ、今なのか

ソフトウェア サプライ チェーン

開発者がソフトウェアを作り、それをユーザーが使うまでの一連の流れ。



ソフトウェア サプライ チェーンの保護がいま、喫緊の課題

650% OSS サプライ チェーン攻撃が急増

Sonatype: https://www.sonatype.com/resources/state-of-the-software-supply-chain-2021

84% の商用コードベースは OSS 脆弱性を含んでいる

Synopsys:
https://venturebeat.com/entrepreneur/synopsys-84-of-codebases-contain-an-open-source-vulnerability/

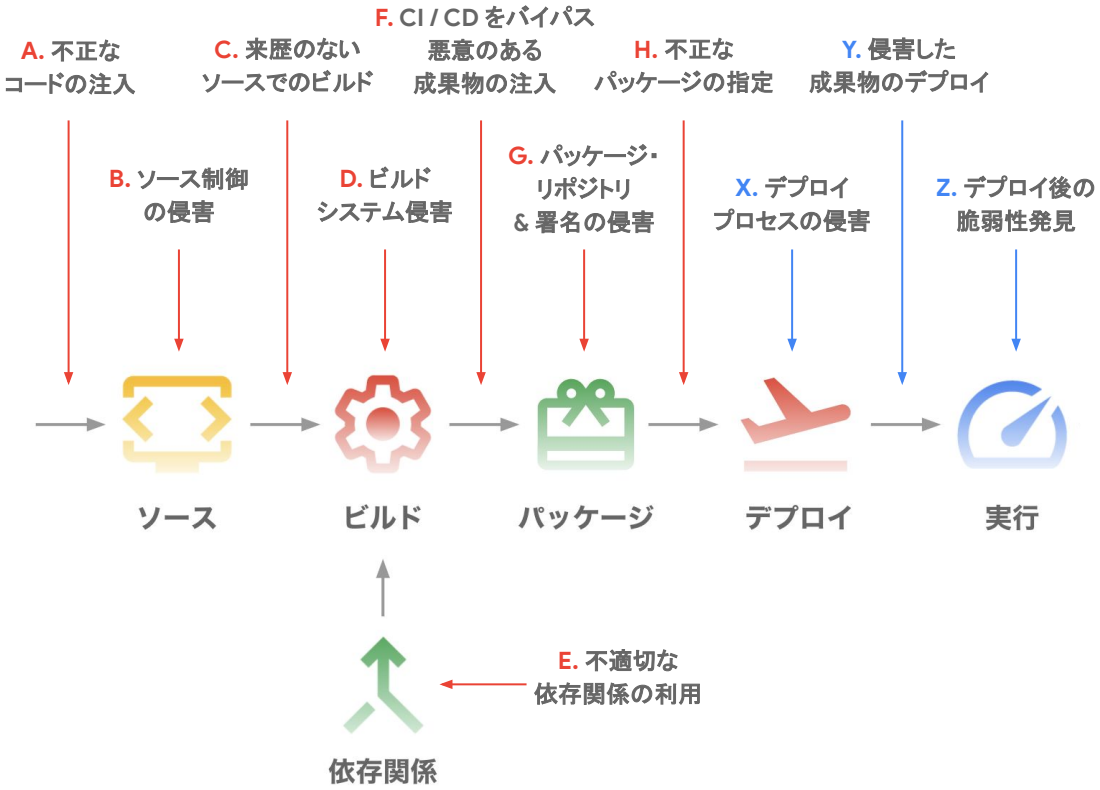
45% の組織は 2025 までサプライ チェーン攻撃を受けると予測

Gartner: https://www.gartner.com/en/documents/4003625

サプライ チェーン攻撃

サプライチェーンの
依存する「ソフトウェア」や
「外部サービス」に対して攻撃、
目的を達成するもの。

OSS や外部サービスへの
依存度が高まる昨今、
対策が難しい攻撃の一つ。



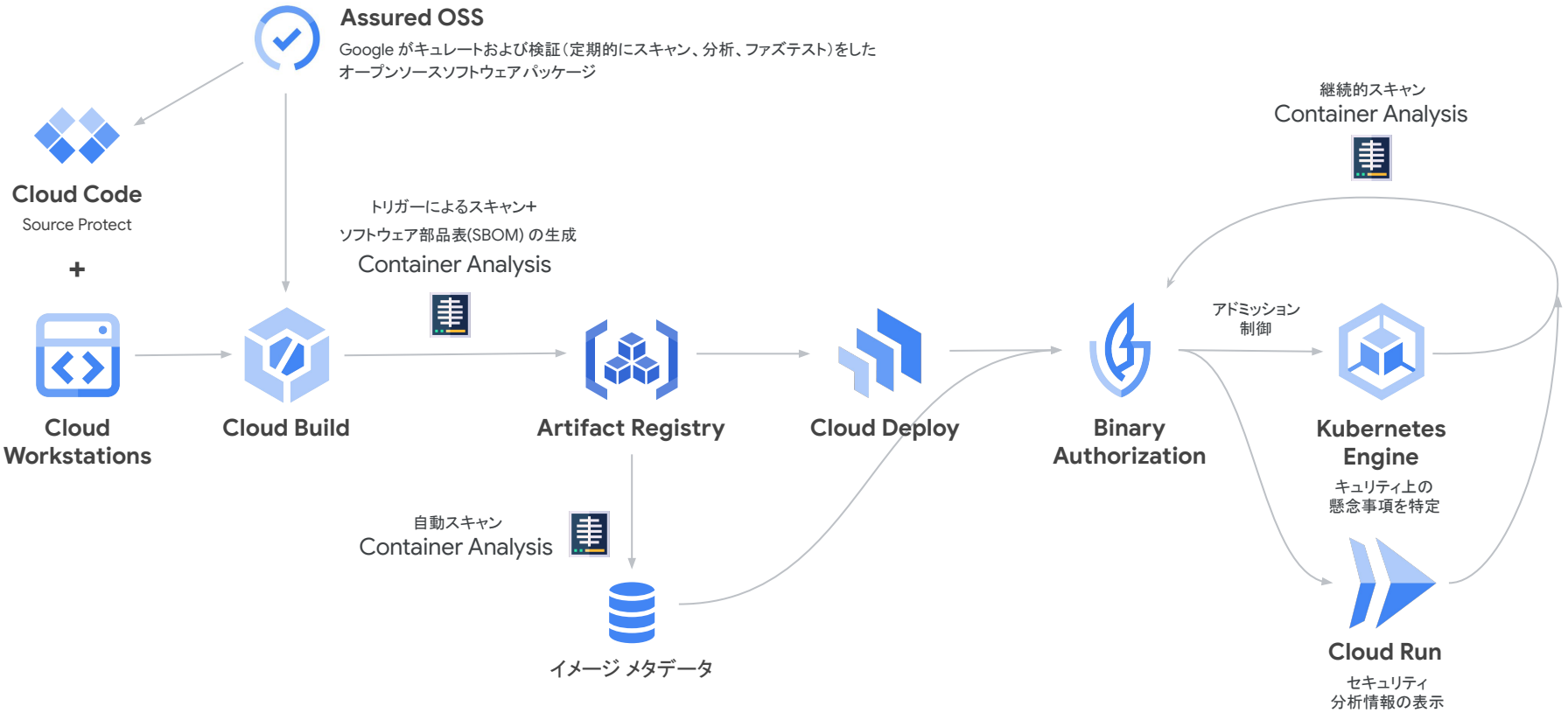
Software supply chain threats | Software supply chain security
https://cloud.google.com/software-supply-chain-security/docs/attack-vectors?hl=ja

Software Delivery Shield

ソフトウェア サプライ チェーンの
セキュリティを強化する
フルマネージドの
エンド ツー エンド ソリューション

https://cloud.google.com/solutions/software-supply-chain-security

Software Delivery Shield を構成する Google Cloud のサービス群



お帰りの際は（途中退席を含む）
アンケートのご回答をお願いします。

https://goo.gle/gcma_sv

Next Tokyo ’23 開催決定

11 月 15 日 (水), 16 日 (木) @ 東京ビッグサイト

参加登録 受付中

Thank you.

Google Cloud