

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

モダンな開発を振り返る	01
ソフトウェアに対する攻守の例	02
ソフトウェア サプライチェーン	03

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

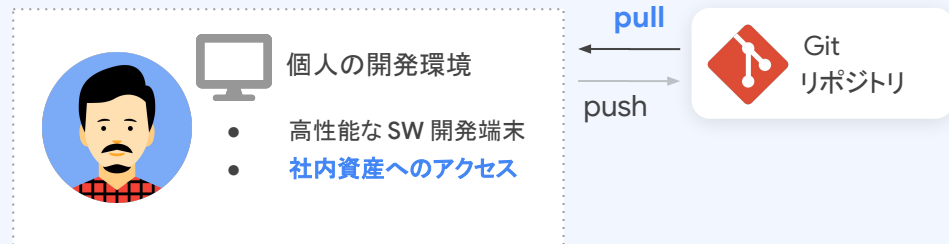


個人の開発環境

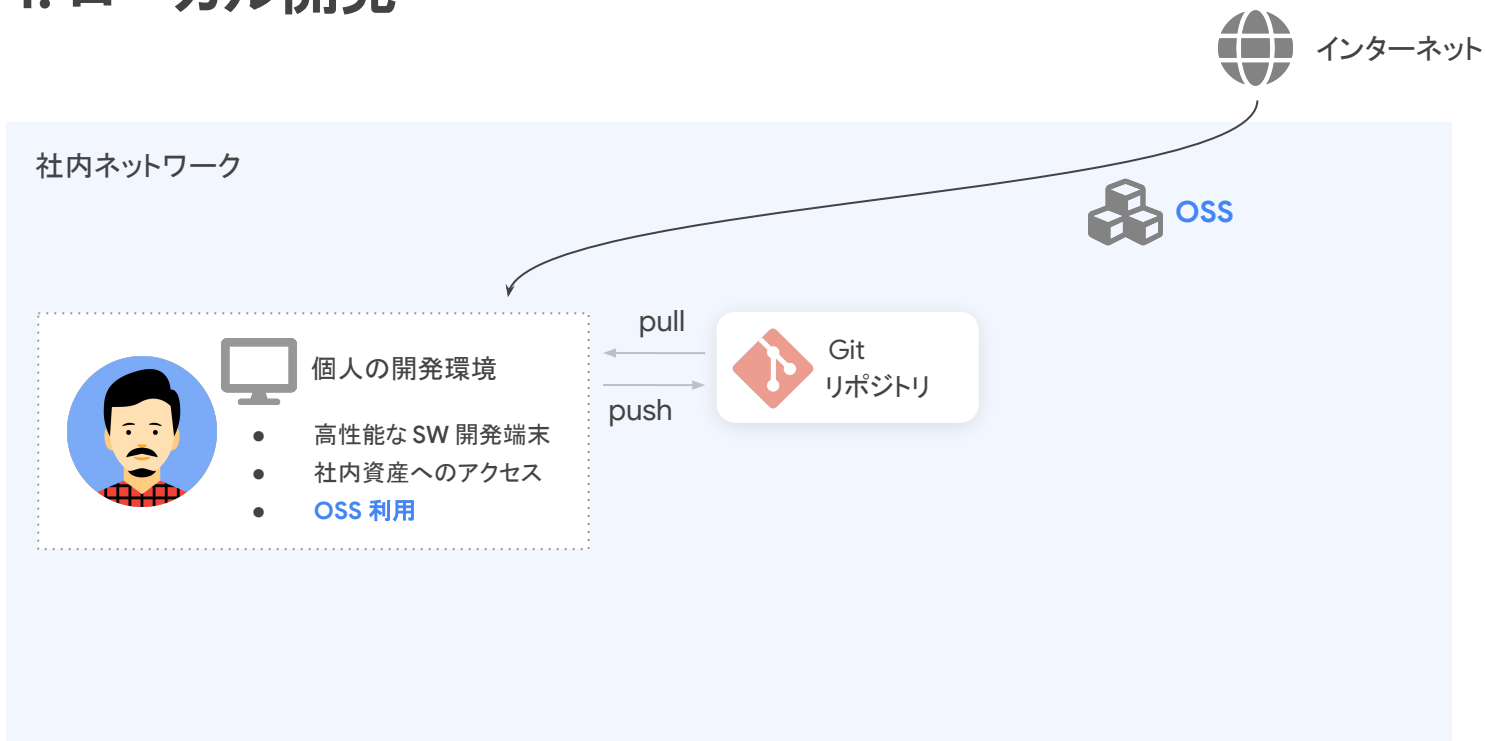
- 高性能な SW 開発端末

1. ローカル開発

社内ネットワーク



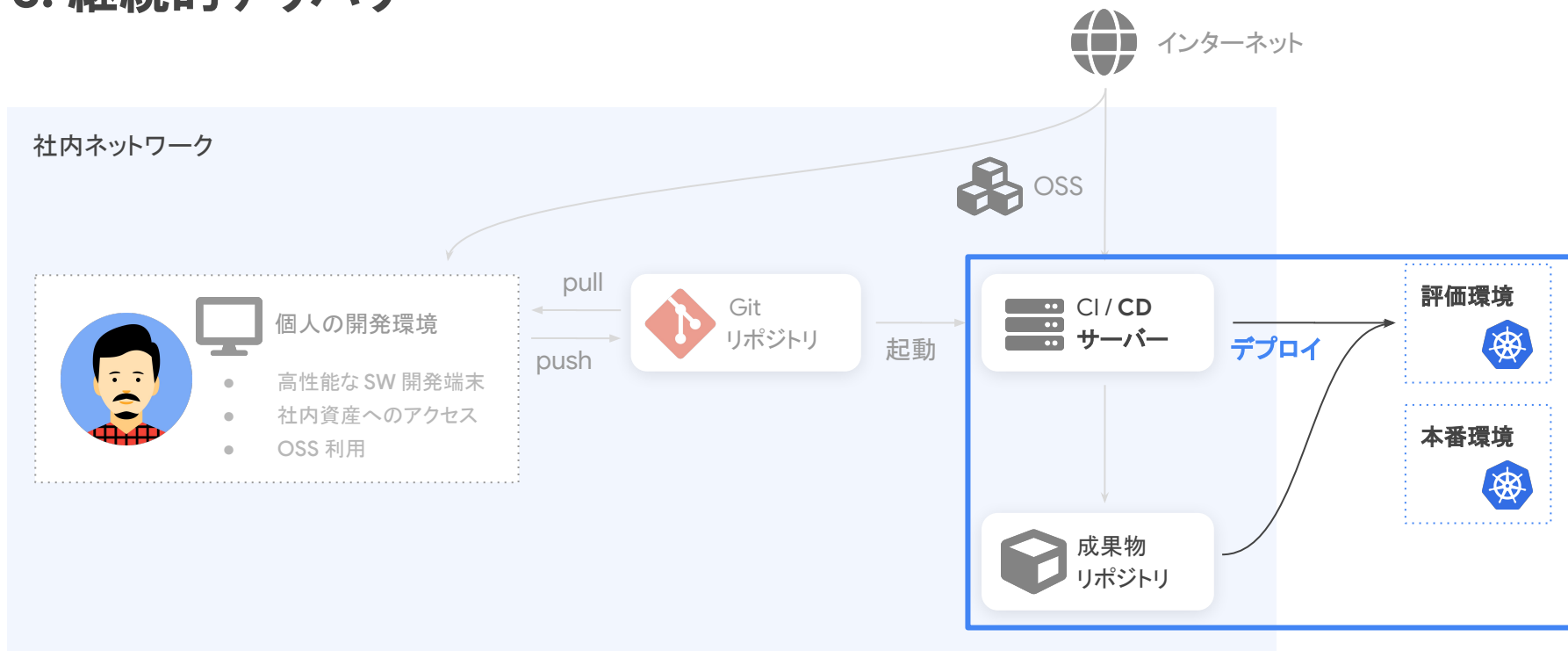
1. ローカル開発



2. 継続的インテグレーション



3. 継続的デリバリー



02

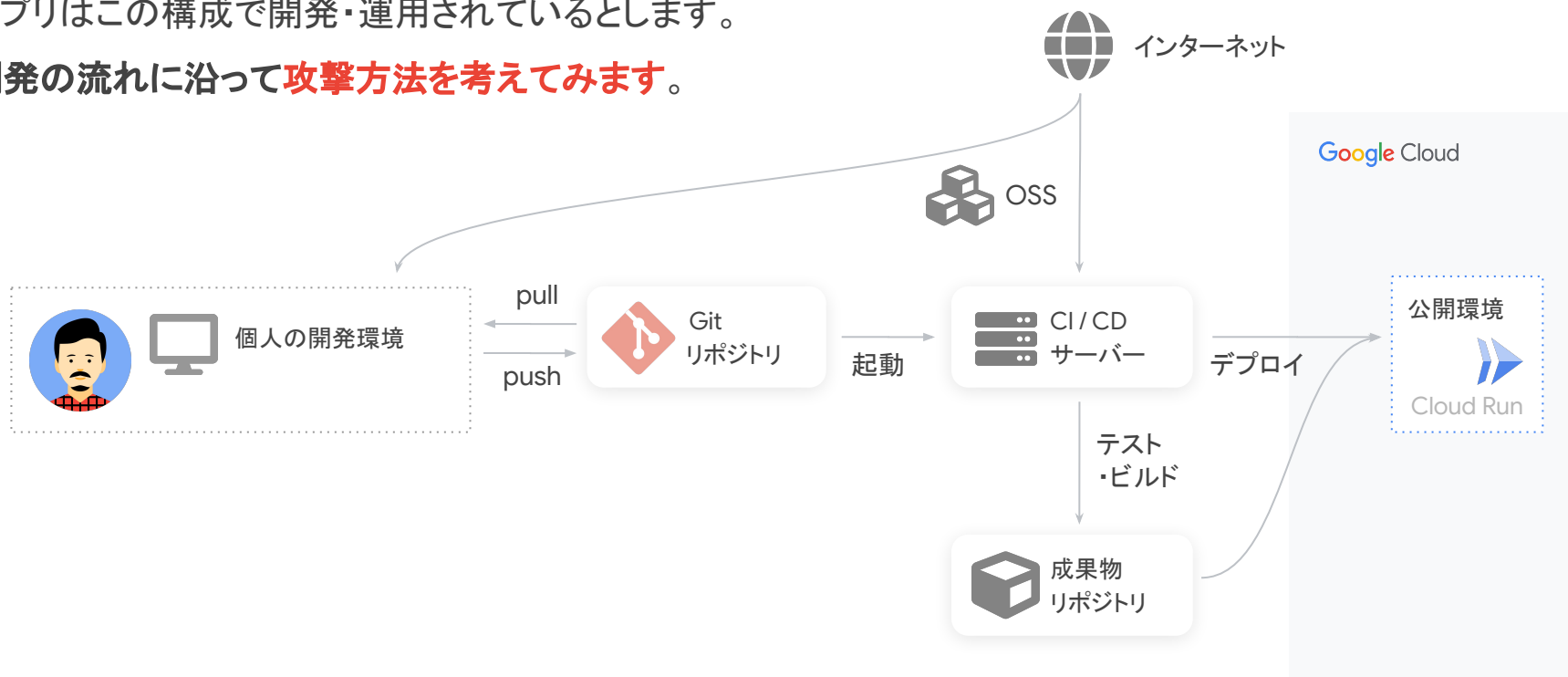
ソフトウェアに対する攻守の例

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。

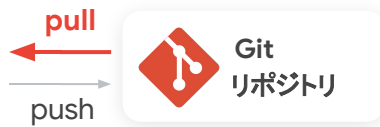
開発の流れに沿って**攻撃方法を考えてみます**。



1. ソースコードへの攻撃



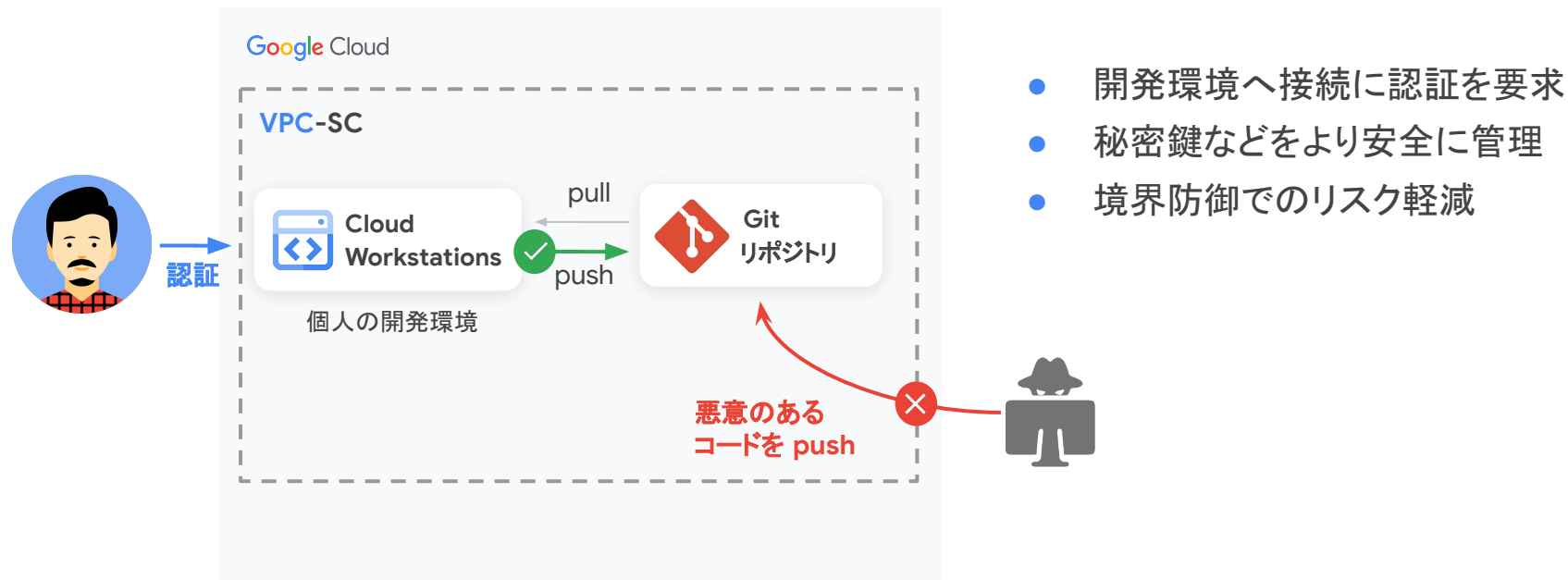
1. ソースコードへの攻撃



悪意のある
コードを push



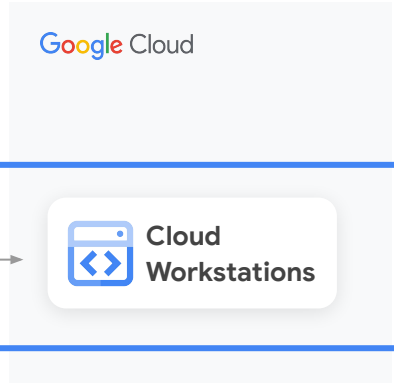
1. ソースコードへの攻撃に対する防御例



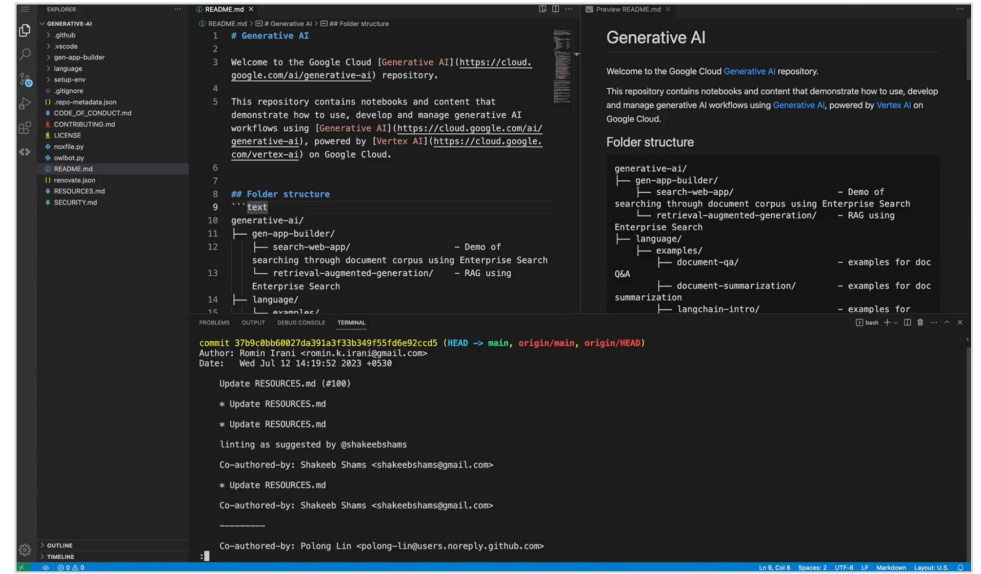
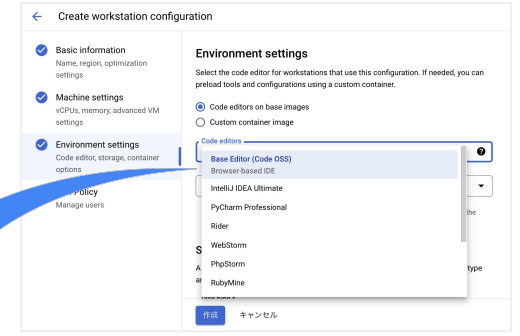
- 開発環境へ接続に認証を要求
- 秘密鍵などをより安全に管理
- 境界防御でのリスク軽減



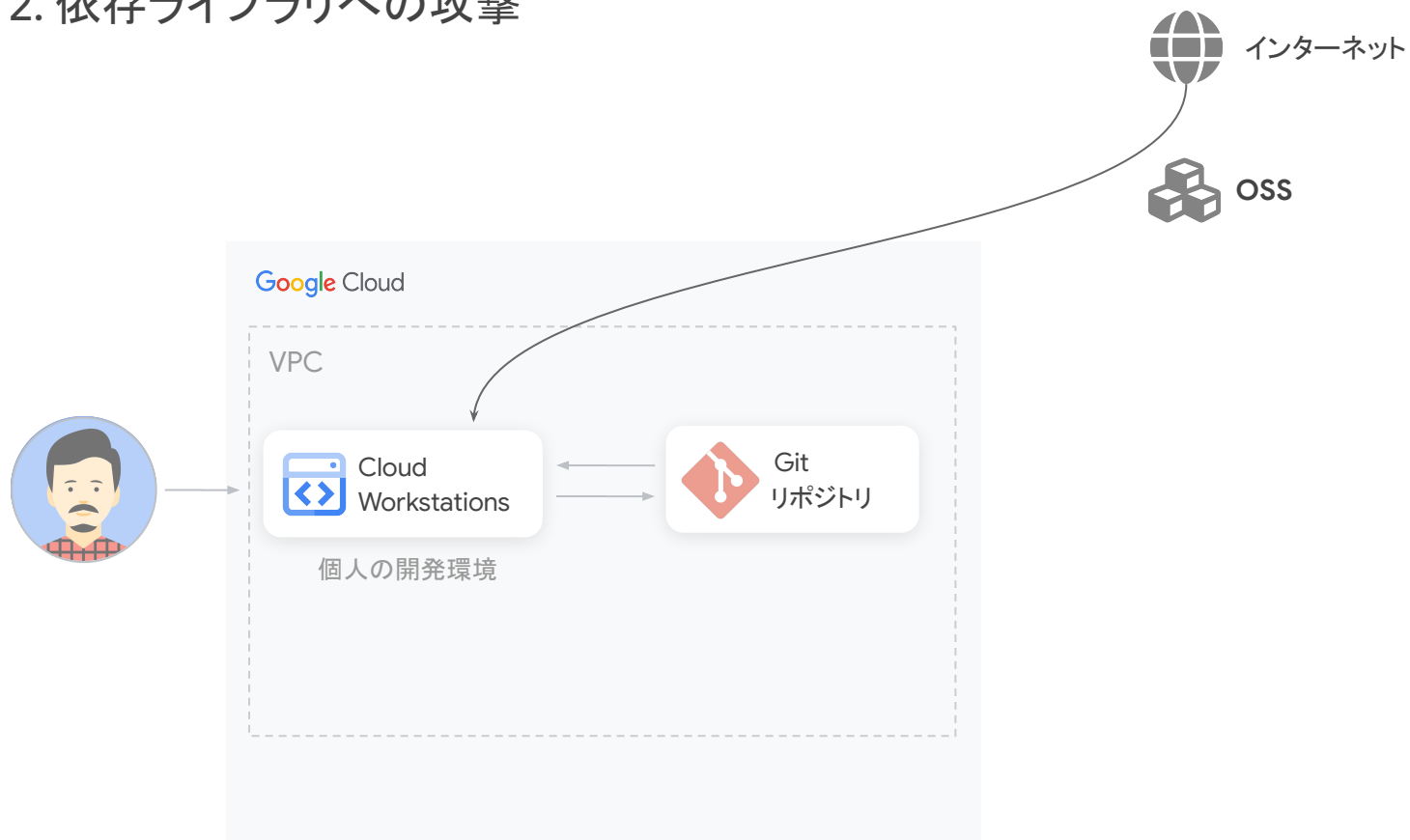
Cloud Workstations



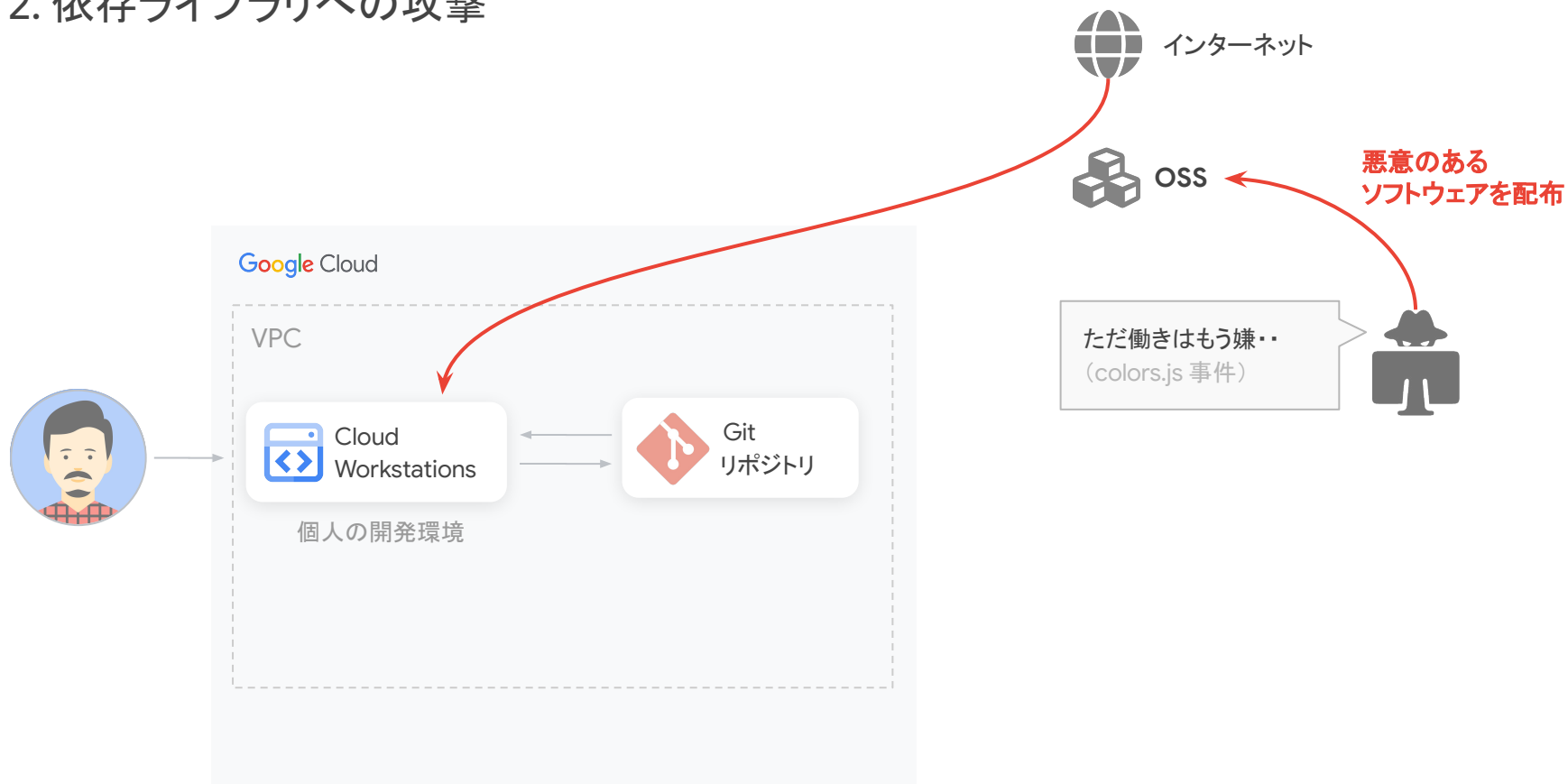
- クラウド上、VPC 内における開発環境
- Code-OSS や IntelliJ IDEA などに対応
- コンテナによる環境の管理・配布



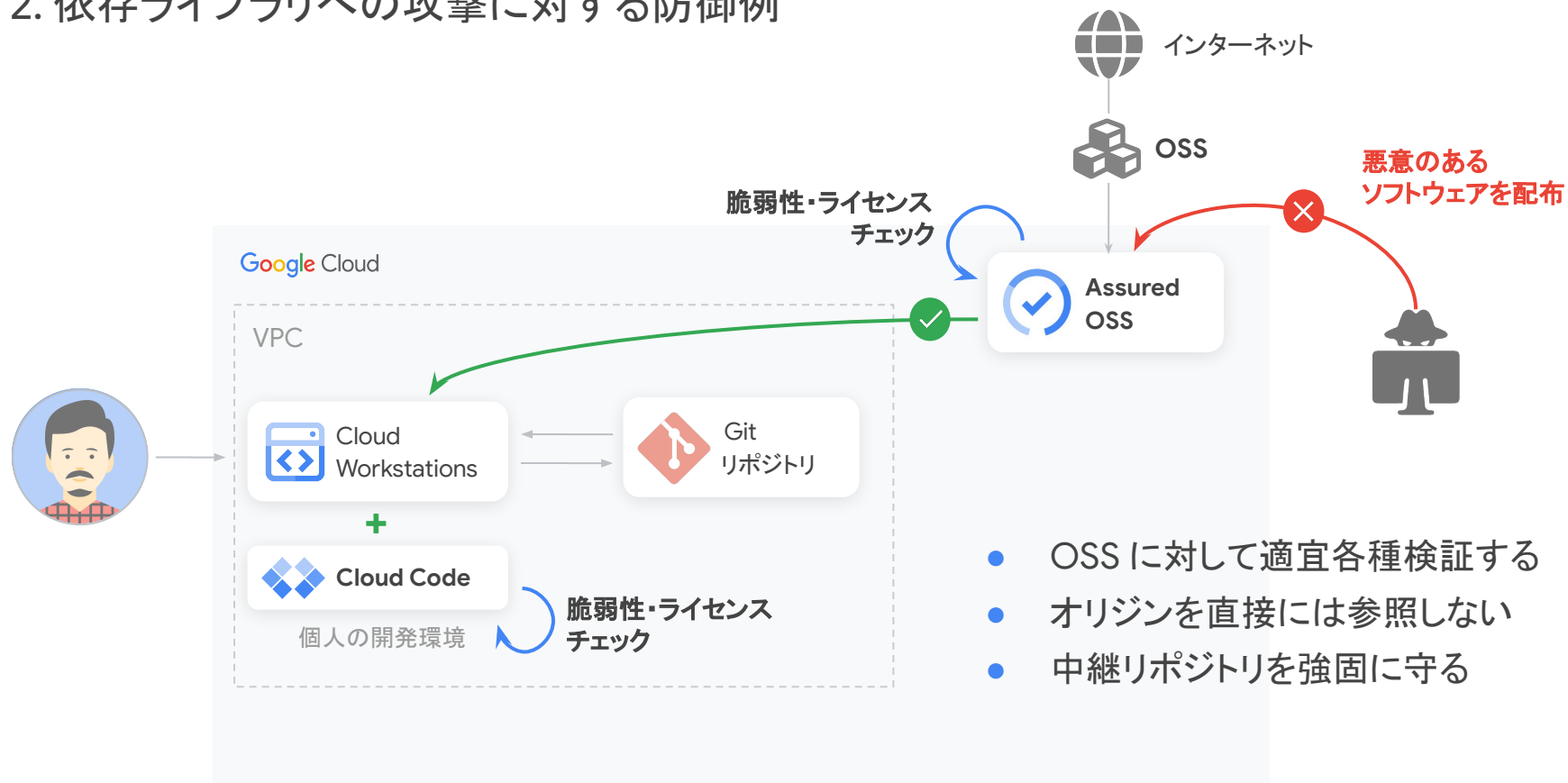
2. 依存ライブラリへの攻撃



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





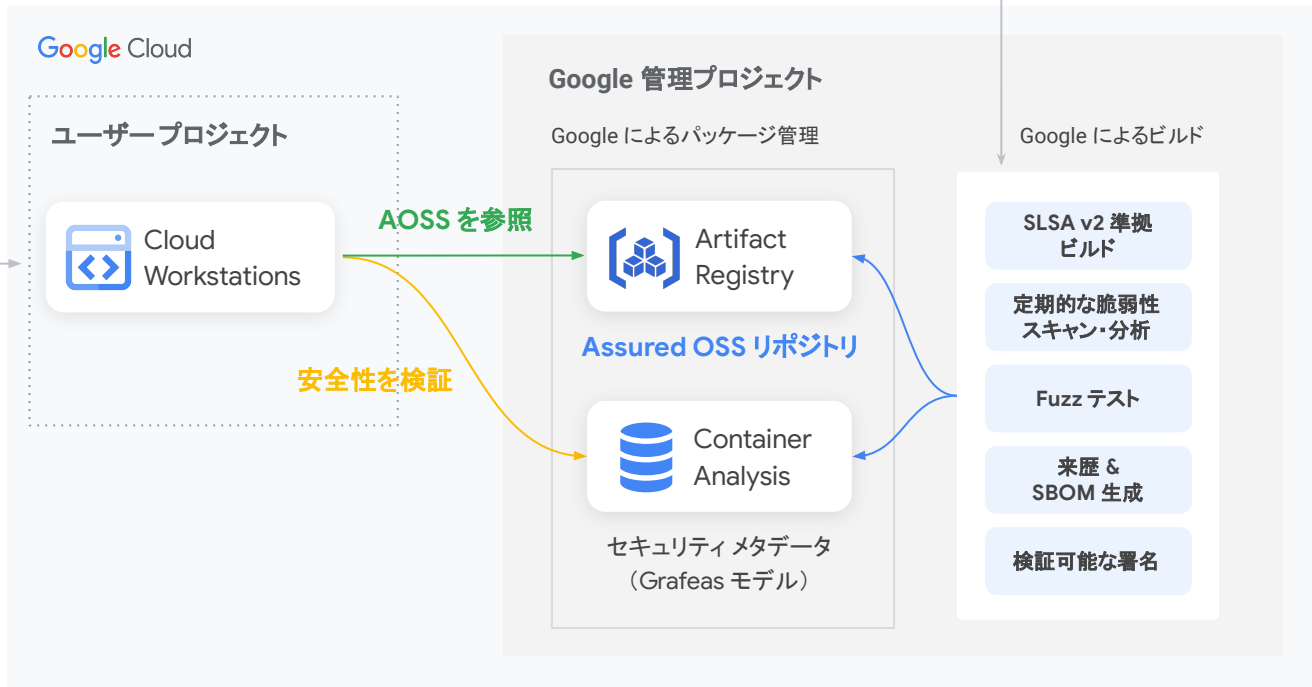
Assured OSS



- **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



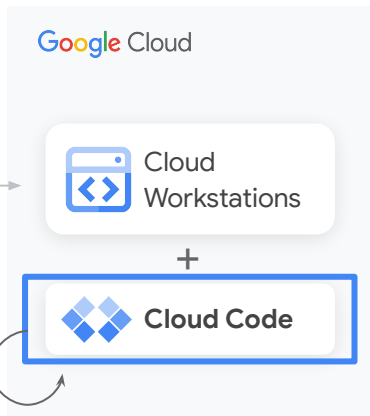
Assured OSS





Cloud Code

IDE プラグイン！ローカルでも使えます



脆弱性・ライセンス
チェック

```

pom.xml 9+ X
demo-app > pom.xml
17
18 <p 2 CRITICAL vulnerabilities detected for org.springframework.boot:spring-boot-starter-
19 web 2.6.8 (0 direct, 4 from transitive dependencies)
20 Licenses: Apache-2.0, EPL-1.0, EPL-2.0, MIT, non-standard
21
22 CRITICAL [CVE-2022-22965] Remote Code Execution in Spring Framework
23 (org.springframework:spring-beans 5.3.16; fixed in 5.3.18) Cloud Code
24
25 CRITICAL [CVE-2022-22965] Remote Code Execution in Spring Framework
26 (org.springframework:spring-webmvc 5.3.16; fixed in 5.3.18) Cloud Code
27
28 HIGH [CVE-2020-36518] Deeply nested json in jackson-databind
29 (com.fasterxml.jackson.core:jackson-databind 2.13.1; fixed in 2.13.2.1) Cloud
30 Code
31
32 <d HIGH [CVE-2022-25857] Uncontrolled Resource Consumption in snakeyaml
33 <dependency>
34 <groupId>org.springframework.boot</groupId>
35 <artifactId>spring-boot-starter-web</artifactId>
36 </dependency>
37 <dependency>
38 <groupId>org.springframework.cloud</groupId>
39 <artifactId>spring-cloud-function-web</artifactId>
40 </dependency>
41 <dependency>
42 <groupId>org.springframework.boot</groupId>

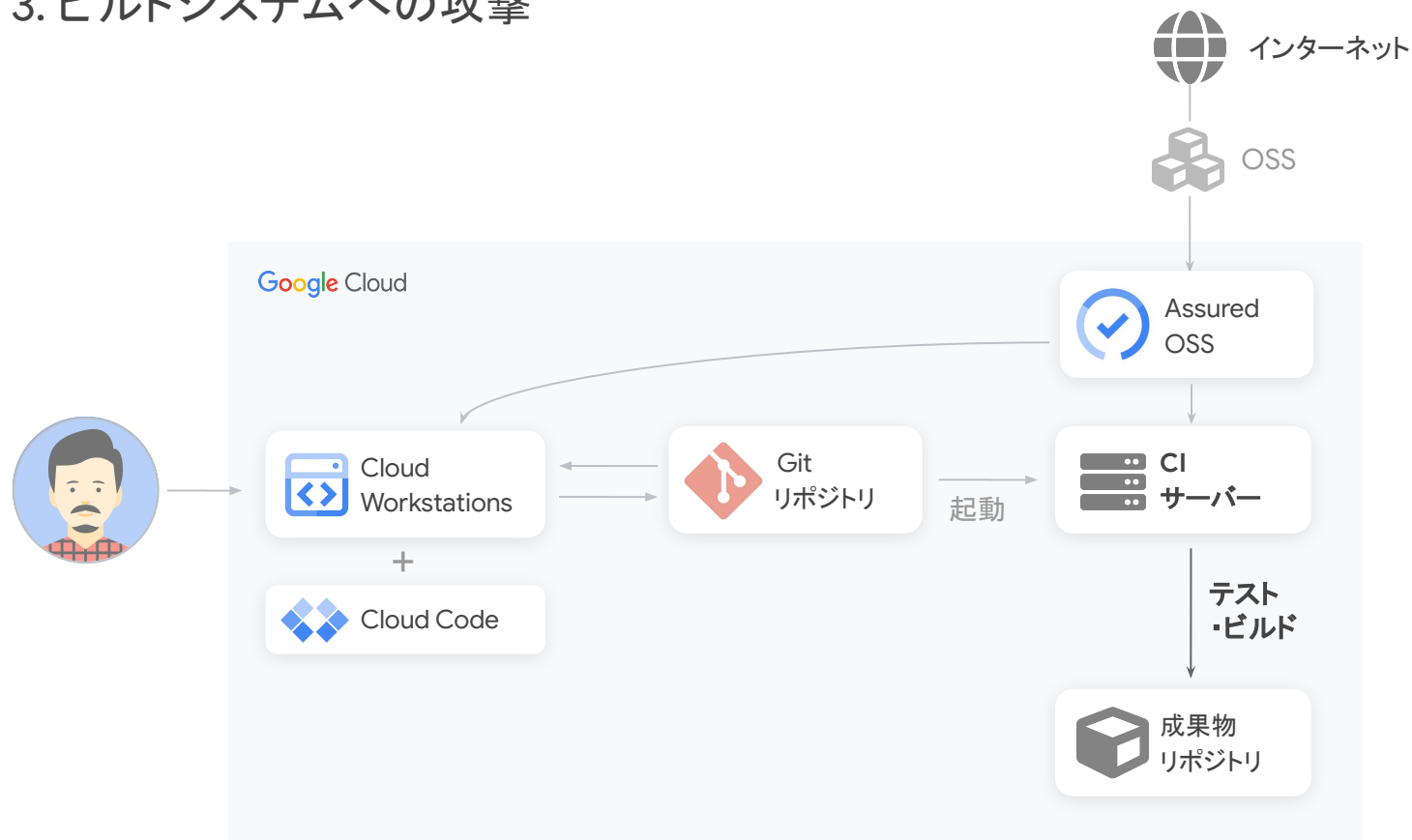
```

PROBLEMS 14 OUTPUT DEBUG CONSOLE TERMINAL Filter (e.g. text, **/*.ts, !**/*.node_modules/**)

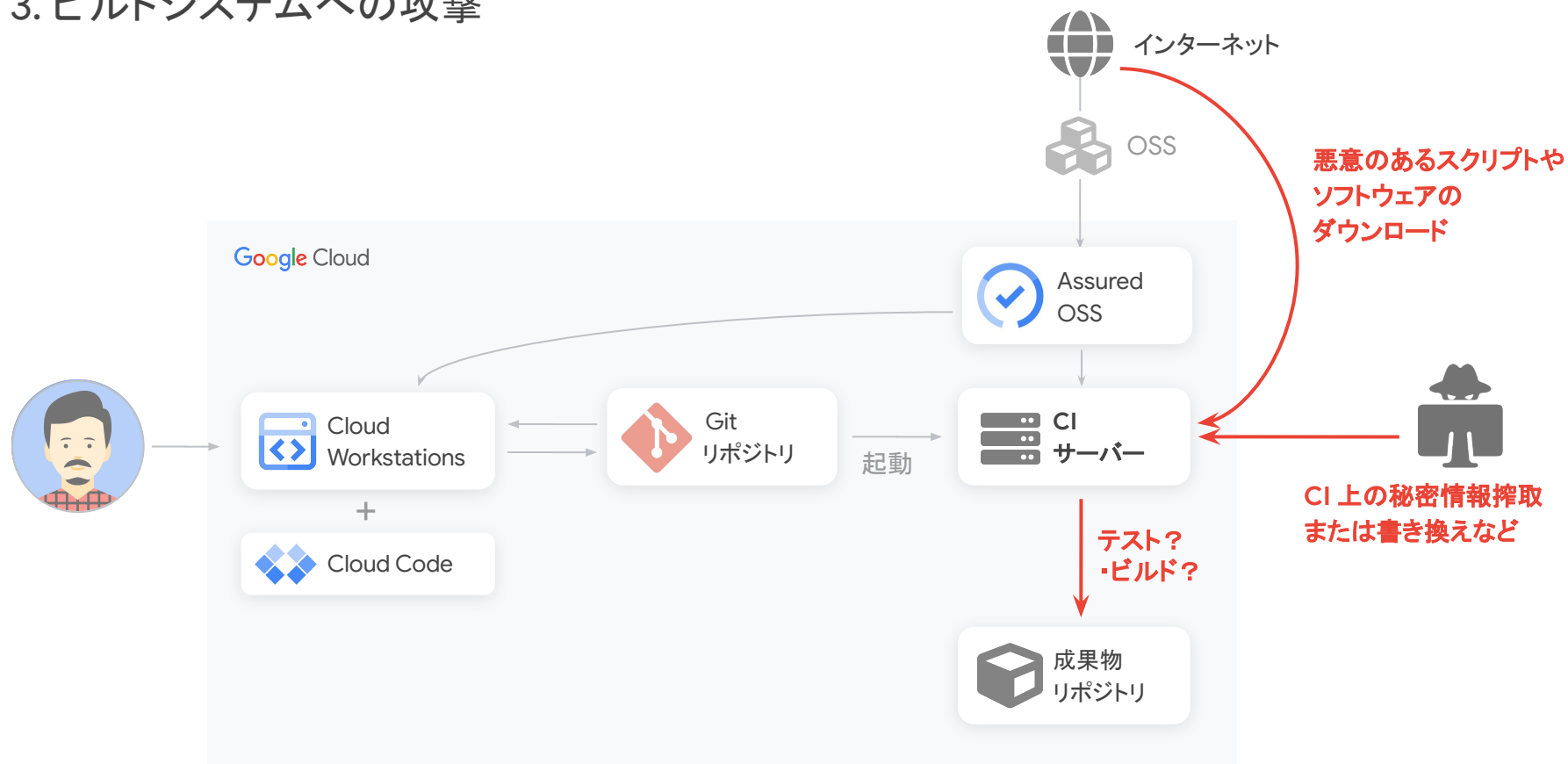
- CRITICAL [CVE-2022-22965] Remote Code Execution in Spring Framework (org.springframework:spring-... Cloud Code [Ln 32, Col 5]
- CRITICAL [CVE-2022-22965] Remote Code Execution in Spring Framework (org.springframework:spring-... Cloud Code [Ln 32, Col 5]
- HIGH [CVE-2020-36518] Deeply nested json in jackson-databind (com.fasterxml.jackson.core:jackson-da... Cloud Code [Ln 32, Col 5]
- HIGH [CVE-2022-25857] Uncontrolled Resource Consumption in snakeyaml (org.yaml:snakeyaml 1.29; fix... Cloud Code [Ln 32, Col 5]
- CRITICAL [CVE-2022-22978] Authorization bypass in Spring Security (org.springframework.security:sprin... Cloud Code [Ln 40, Col 5]
- CRITICAL [CVE-2021-42392] RCE in H2 Console (com.h2database:h2 1.4.200; fixed in 2.0.206) Cloud Code [Ln 70, Col 5]
- CRITICAL [CVE-2022-23221] Arbitrary code execution in H2 Console (com.h2database:h2 1.4.200; fixed i... Cloud Code [Ln 70, Col 5]

london-cloudcode Ln 49, Col 16 Spaces: 2 UTF-8 LF () XML

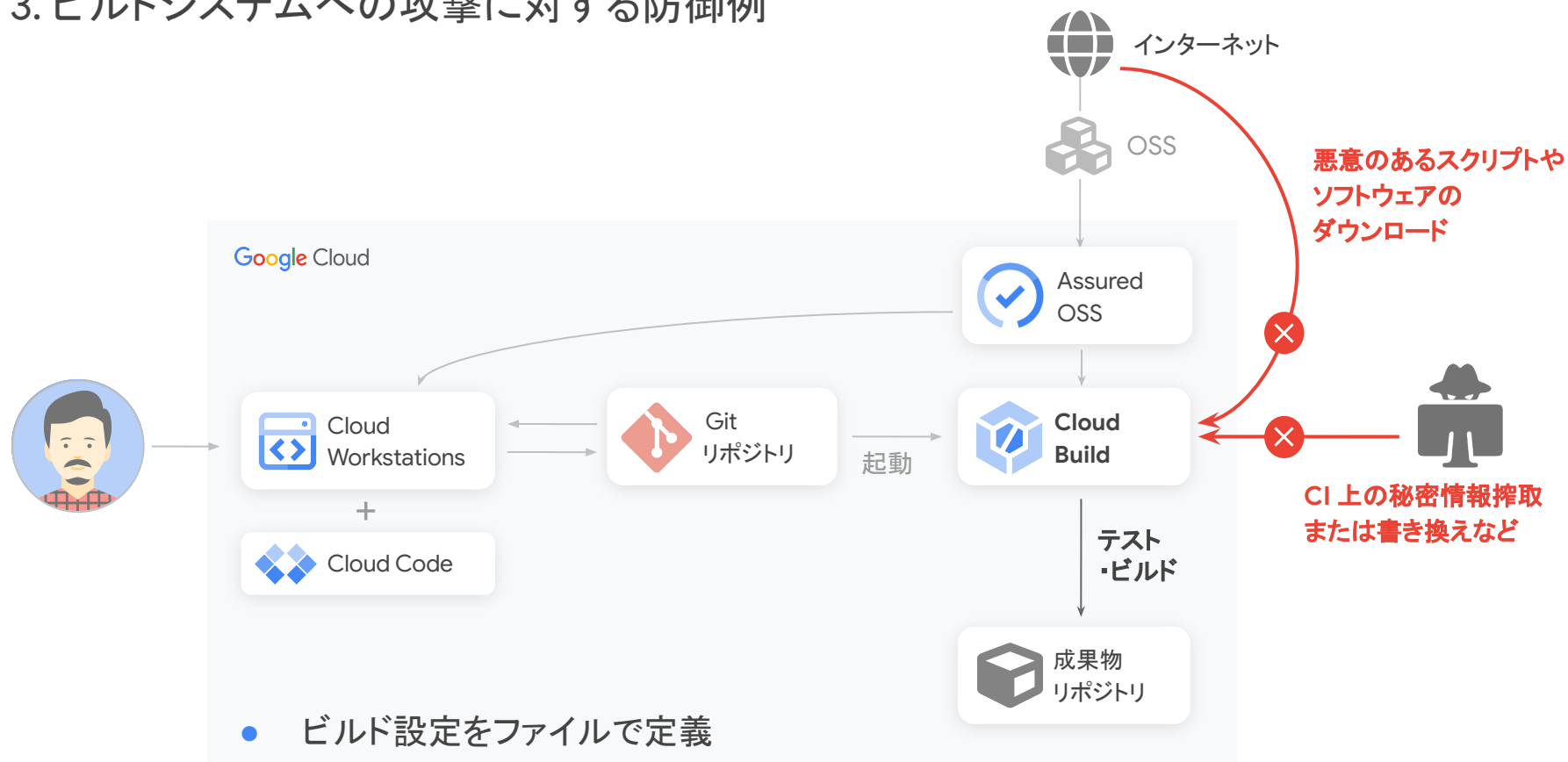
3. ビルドシステムへの攻撃



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/>

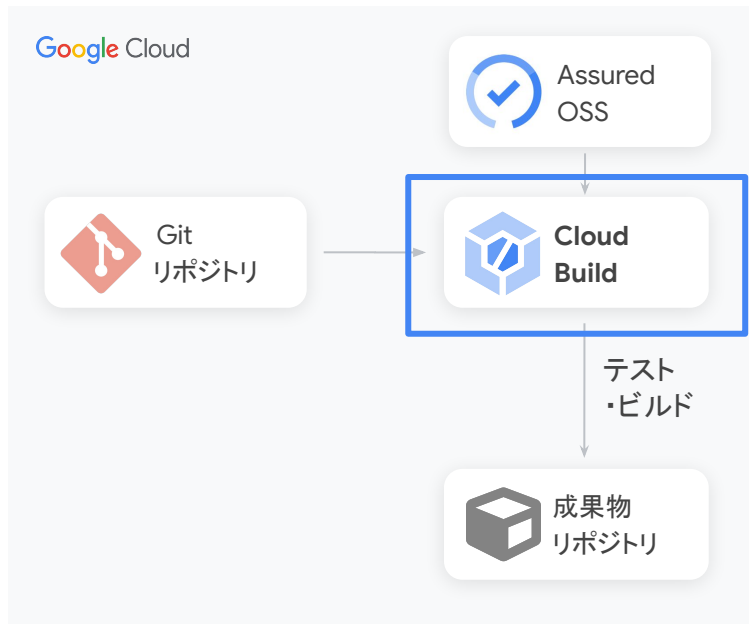




Cloud Build

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



dev に関するセキュリティ分析情報 [プレビュー](#) ✕

ソフトウェア デリバリー シールドは、ソフトウェア デリバリーのライフサイクル全体にわたって、アーティファクトの整合性を保護する新しいソリューションです。不正行為を防止し、整合性を高め、パッケージとインフラストラクチャを保護する仕組みについて、[詳細をご覧ください](#)。

3 アーティファクト (dev)
SLSA ビルドレベル 3 [説明](#)

脆弱性

重大	高	中	低
0	0	0	0

スキャンされたアーティファクト [dev](#)

ビルド

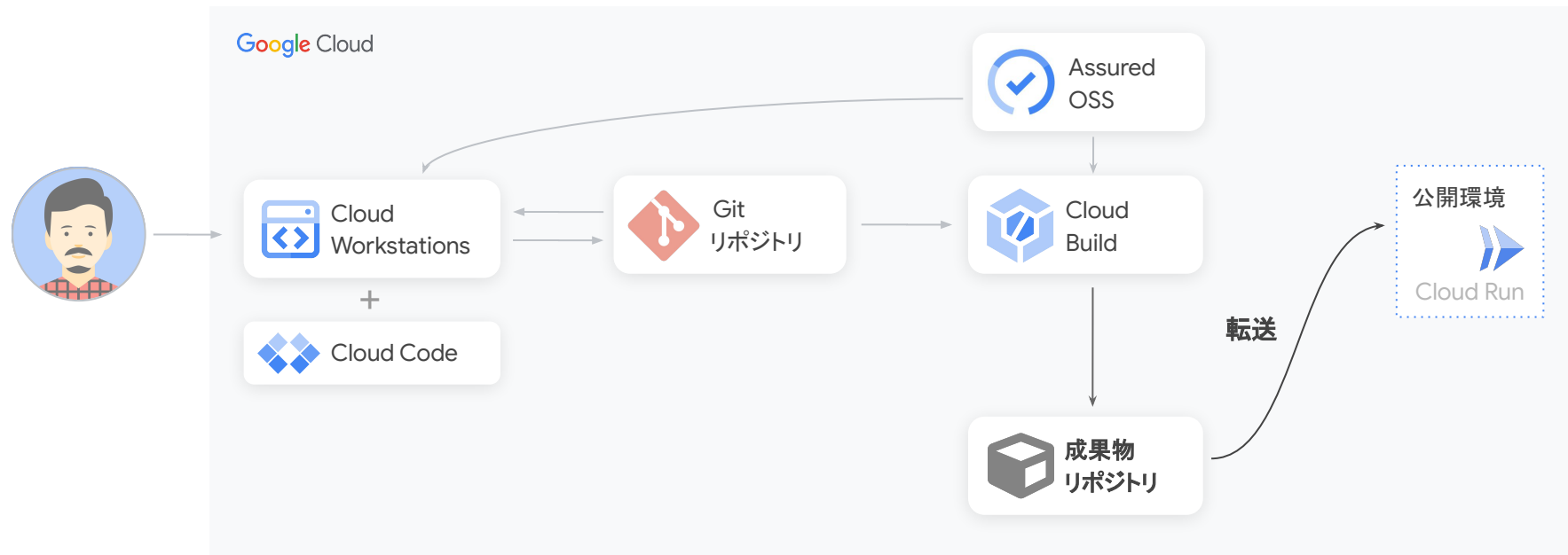
詳細

ログ	ce05709f
Builder	Cloud Build
完了日時	7 時間前

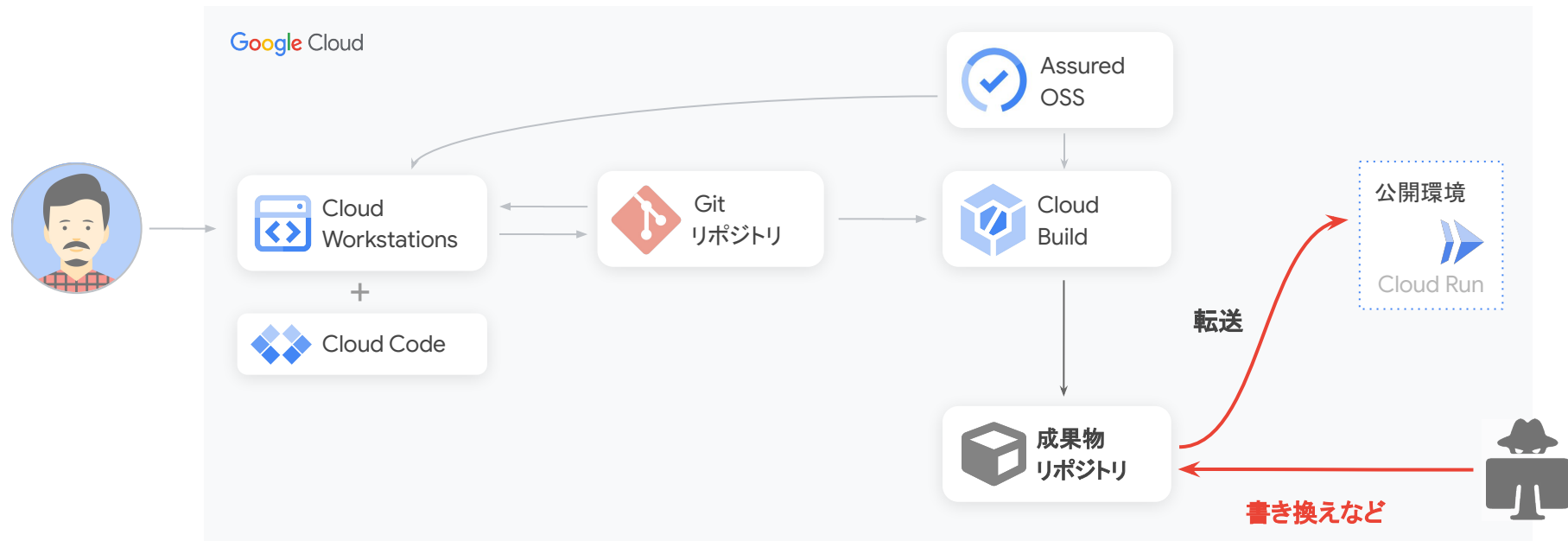
来歴 [🔗](#)

```
{"_type": "https://in-toto.io/Statement/v0.1",  
 "subject": [  
   {  
     "name": "https://asia-northeast1-docker.pkg.dev/sample-na/  
     "digest": {
```

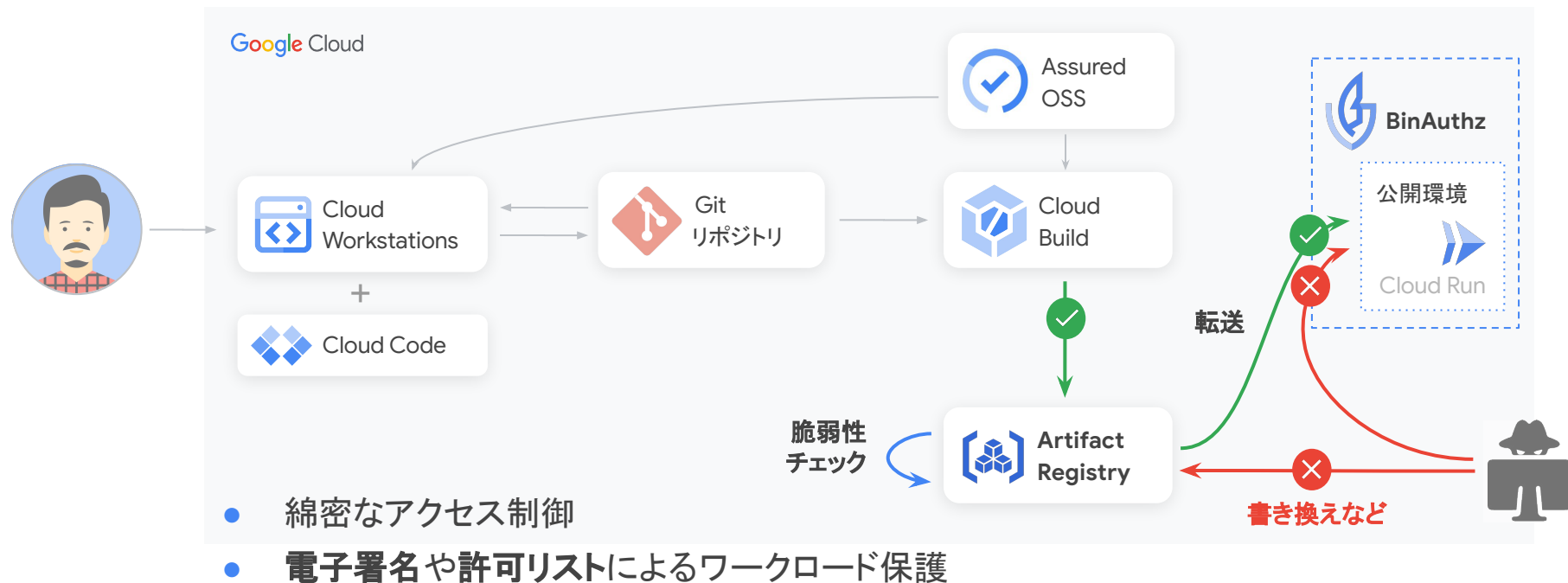
4. 成果物への攻撃



4. 成果物への攻撃



4. 成果物への攻撃に対する防御例

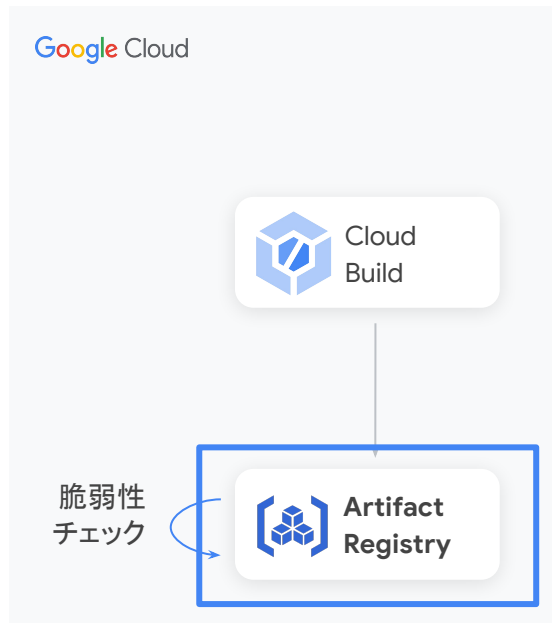




Artifact Registry

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

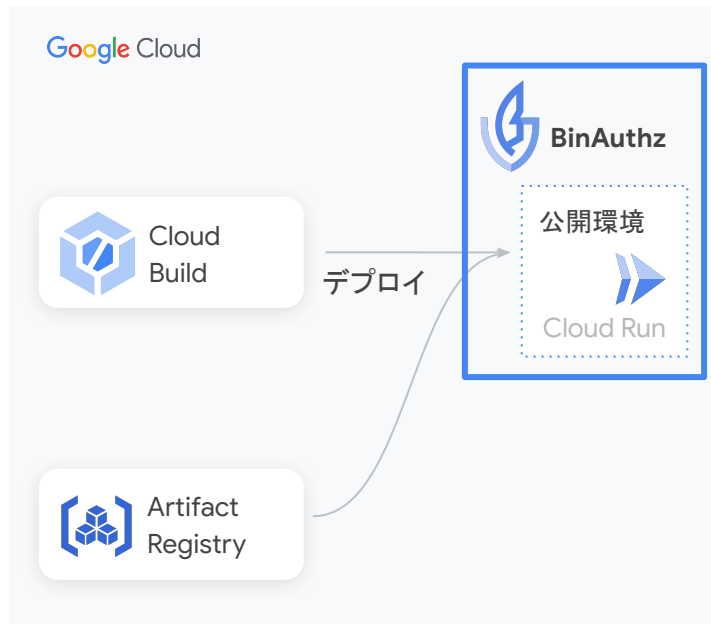




Binary Authorization

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？





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

リビジョンをデプロイしています

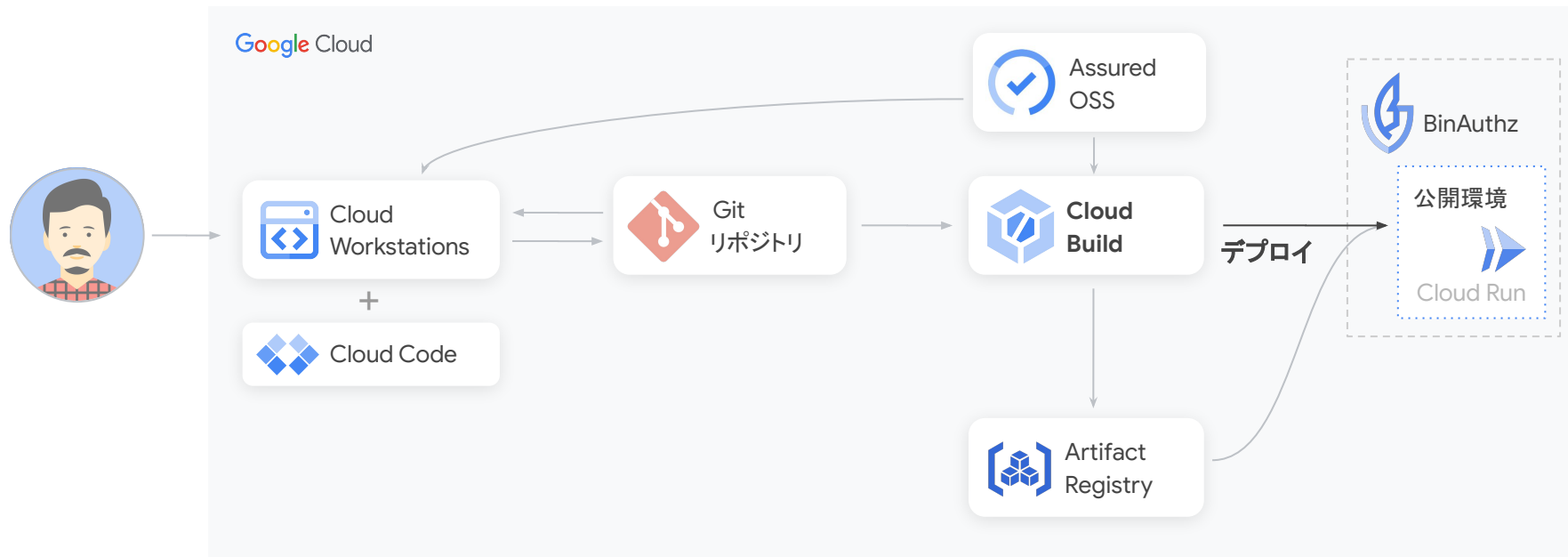
Service update rejected by Binary Authorization policy: Revision 'hello-00026-wox' is not ready and cannot serve traffic. Container image 'us-docker.pkg.dev/cloudrun/container/hello@sha256:f85693e789a09ea6629b0a7717102db1e7bea45d6c8616e5543e224f9edf5107' is not authorized by policy. Denied by an ALWAYS_DENY admission rule

BREAKGLASS

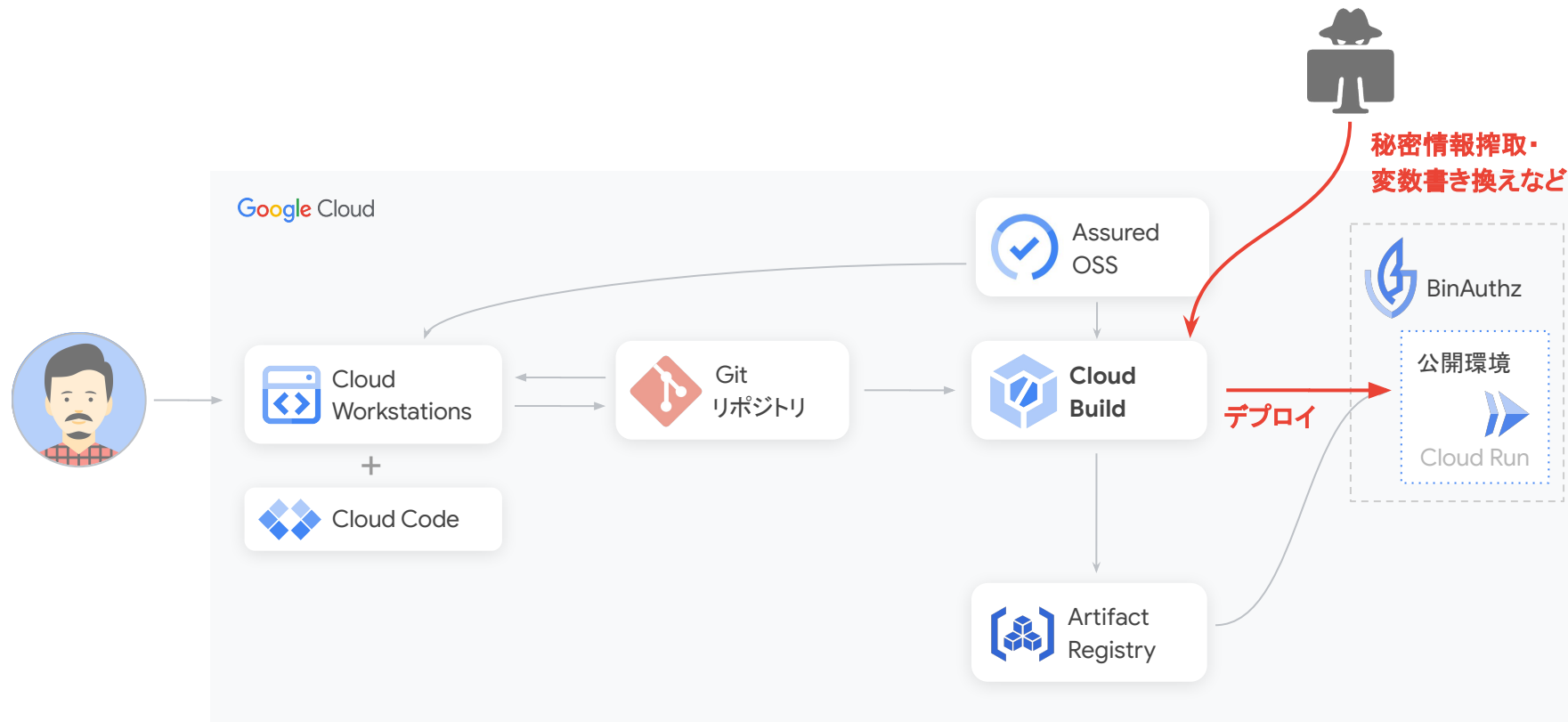
Pending

```
admin@cloudshell:~ (rnak-altostrat)$
admin@cloudshell:~ (rnak-altostrat)$ gcloud run deploy hello --region asia-northeast1 --image us-docker.pkg.dev/cloudrun/container/hello --binary-authorization=default
Deploying container to Cloud Run service [hello] in project [rnak-altostrat] region [asia-northeast1]
X Deploying... Revision 'hello-00026-wox' is not ready and cannot serve traffic. Container image 'us-docker.pkg.dev/cloudrun/container/hello@sha256:f85693e789a09ea6629b0a7717102db1e7bea45d6c8616e5543e224f9edf5107' is not authorized by policy. Denied by an ALWAYS_DENY admission rule
OK Creating Revision...
X Routing traffic... Revision 'hello-00026-wox' is not ready and cannot serve traffic. Container image 'us-docker.pkg.dev/cloudrun/container/hello@sha256:f85693e789a09ea6629b0a7717102db1e7bea45d6c8616e5543e224f9edf5107' is not authorized by policy. Denied by an ALWAYS_DENY admission rule
Deployment failed
ERROR: (gcloud.run.deploy) Revision 'hello-00026-wox' is not ready and cannot serve traffic. Container image 'us-docker.pkg.dev/cloudrun/container/hello@sha256:f85693e789a09ea6629b0a7717102db1e7bea45d6c8616e5543e224f9edf5107' is not authorized by policy. Denied by an ALWAYS_DENY admission rule
admin@cloudshell:~ (rnak-altostrat)$
```

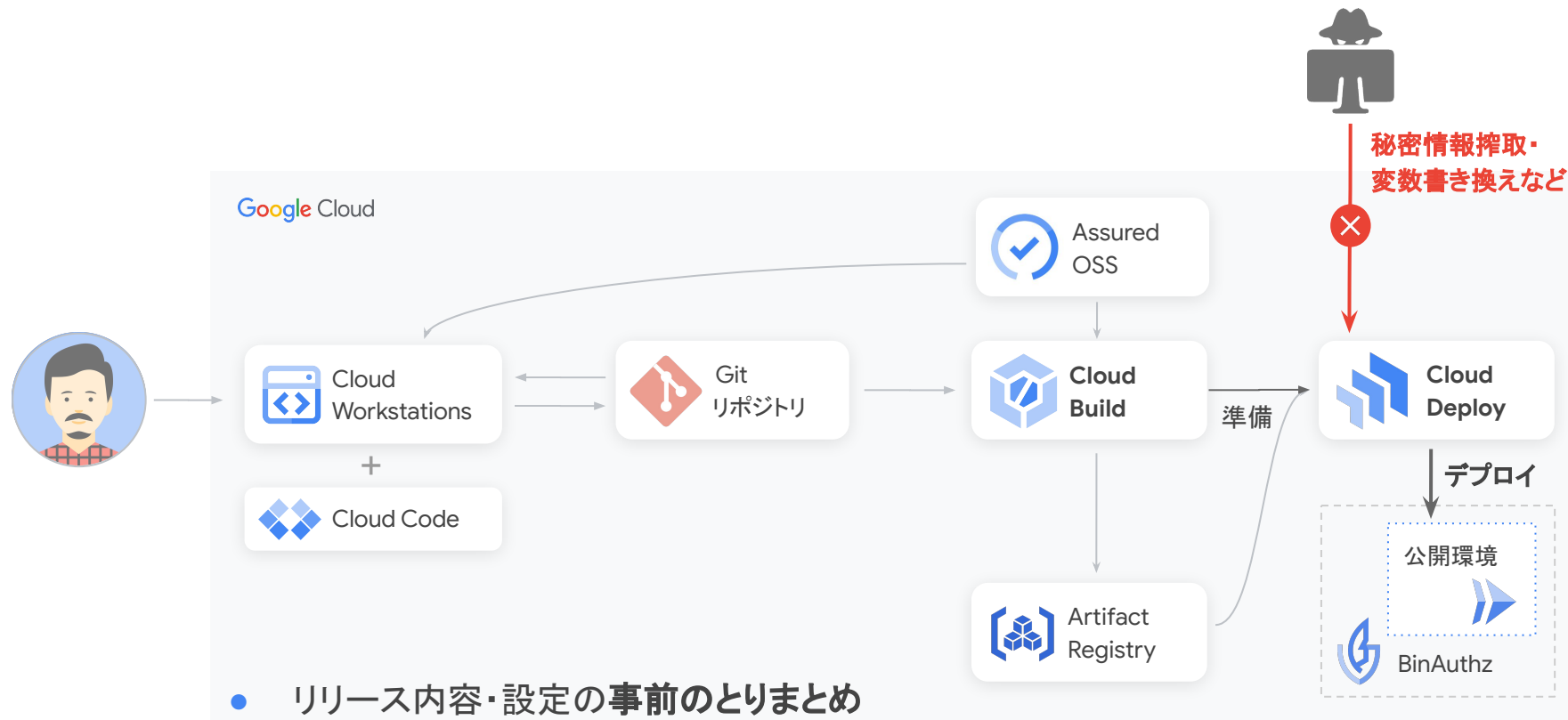
5. デプロイシステムへの攻撃



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

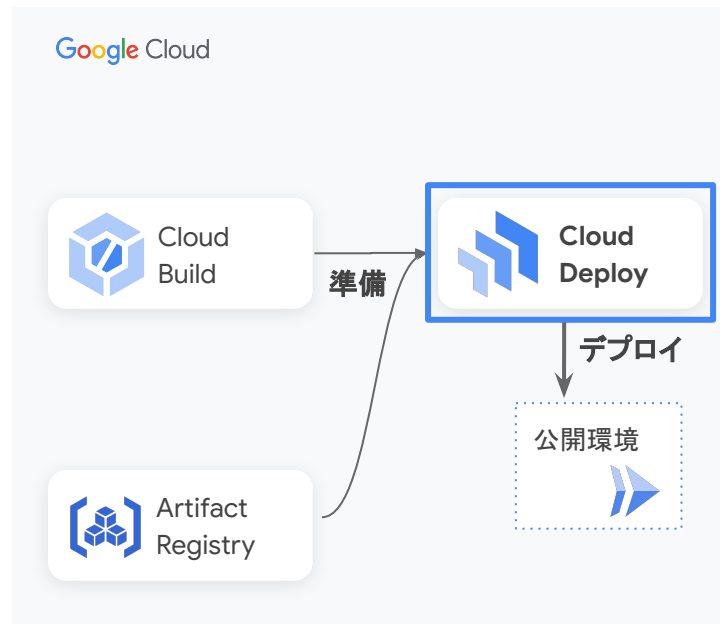




Cloud Deploy

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





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

リリースの詳細

名前: git-9fa636d-20230712-171214
作成日: 2023/07/13 2:12:20
最新のロールアウト先: dev

すべて表示

ロールアウト: アーティファクト | パイプライン インスタンス

レンダリング ソース

アーカイブ ↑ | ダウンロード | gs://572c3e0f363142bfb220b49ad2e6b57_clouddeploy/source/1689181939.151171-87e907aa6094459a9bd78903f8167b67bz | ダウンロード ↓

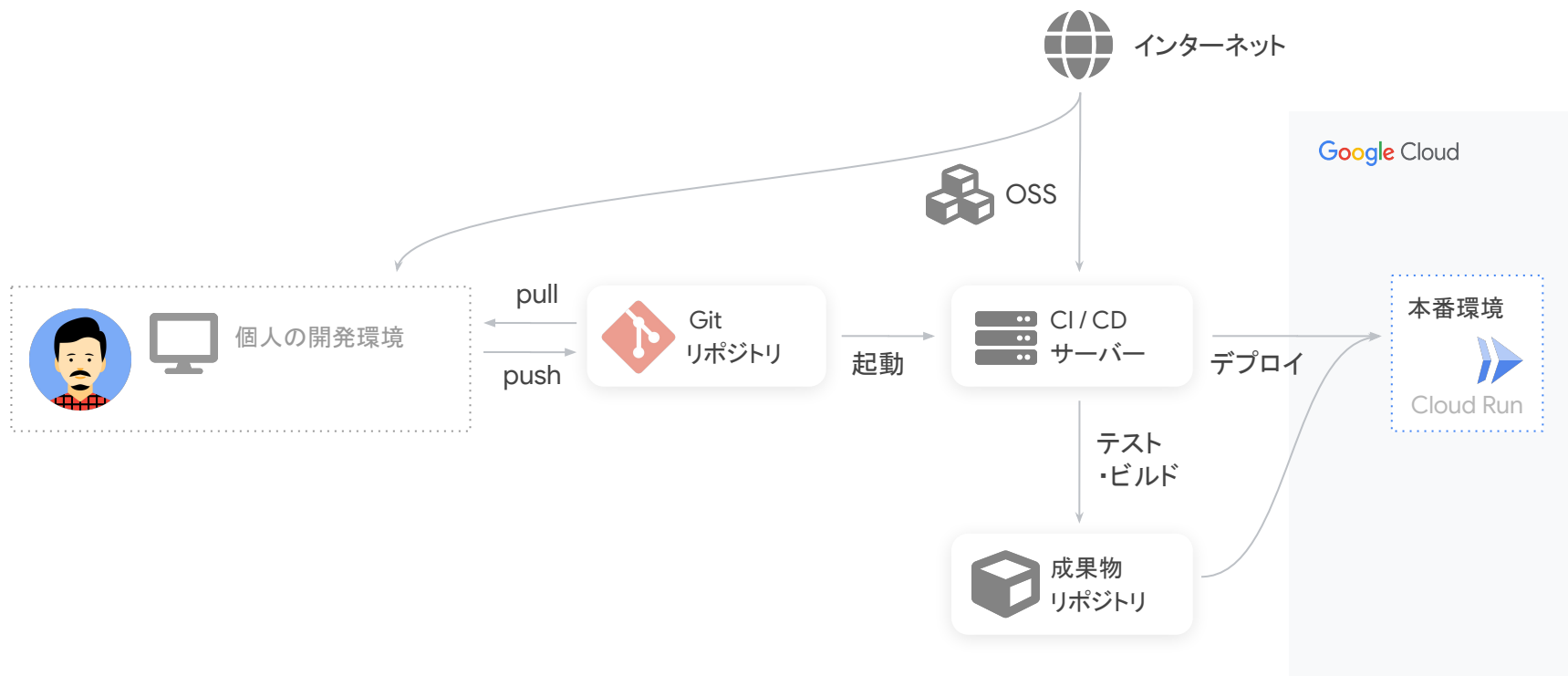
ビルドアーティファクト

アーティファクト名 ↑	タグ
api	asia-northeast1-docker_pkg_dev/rnak-altostrat/my-apps/api:9fa636d@sha256:45698eac14479591b7318c15f9a0b15cf5fbaec958e4...11ca728c
web	asia-northeast1-docker_pkg_dev/rnak-altostrat/my-apps/web:9fa636d@sha256:9aaef6df1441bef4a7d73c68a1728384c83497139d36...2ab3ed66

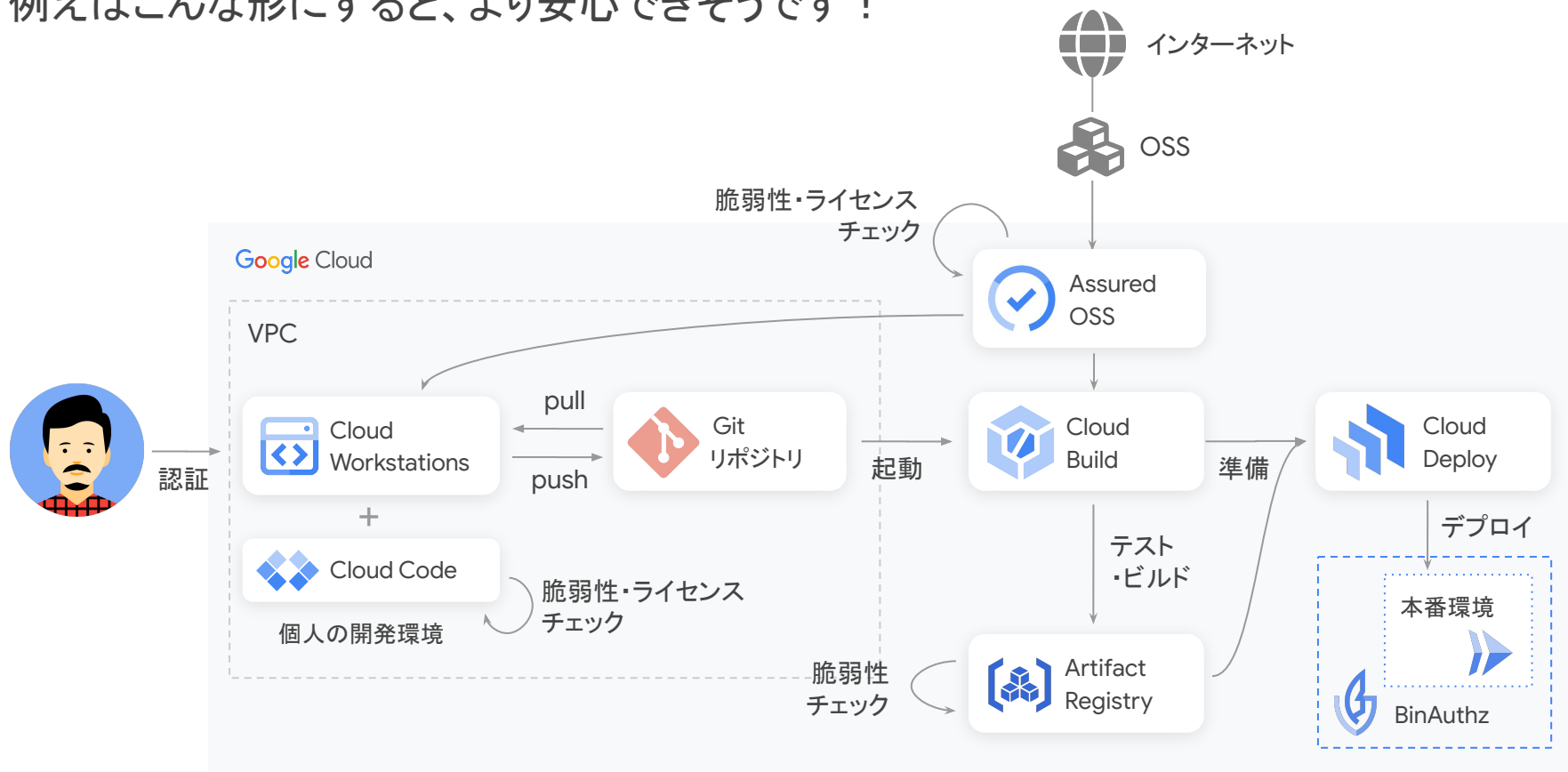
ターゲットのアーティファクト

ターゲット	レンダリング ステータス	レンダリング ログ	レンダリングされたアーティファクト	ストレージの場所	リリースインスペクタ
dev	成功	ビルド e8b853...	scaffold.yaml, manifest.yaml	gs://asia-northeast1.deploy-artifacts.rnak...	アーティファクトを表示 ✓
prod	成功	ビルド f0fea5c...	scaffold.yaml, manifest.yaml	gs://asia-northeast1.deploy-artifacts.rnak...	アーティファクトを表示 ✓

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！



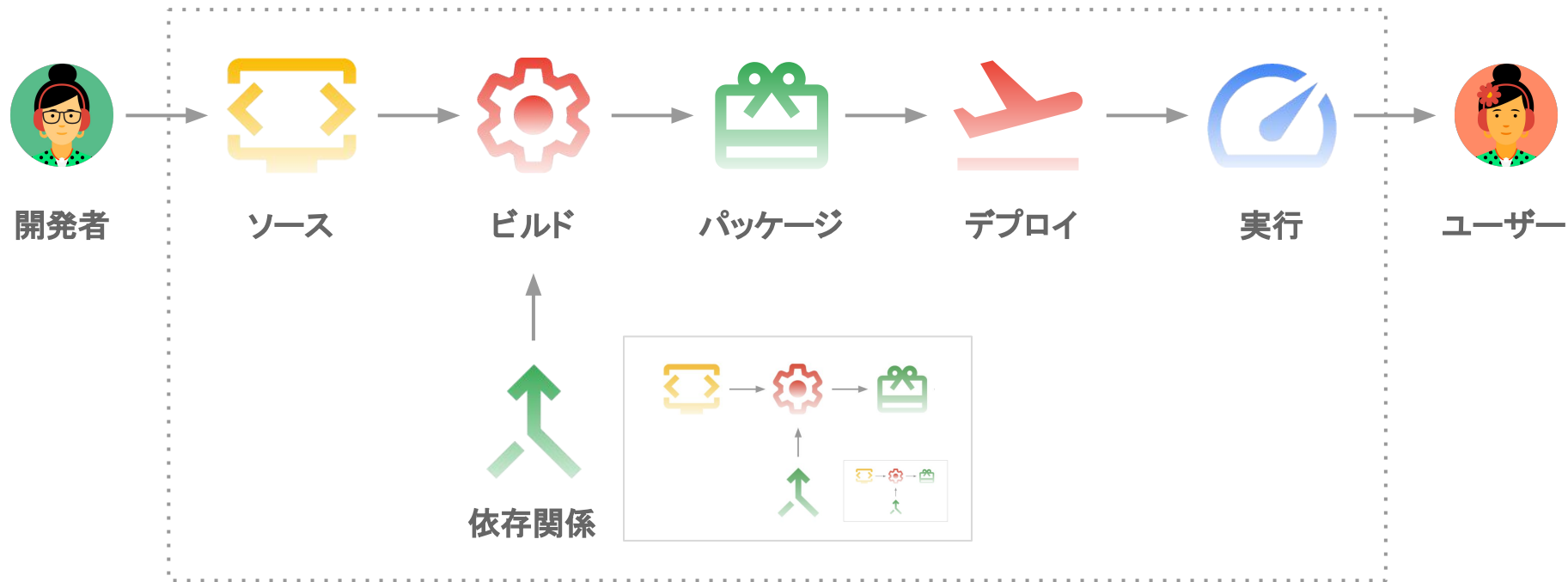
03

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/>

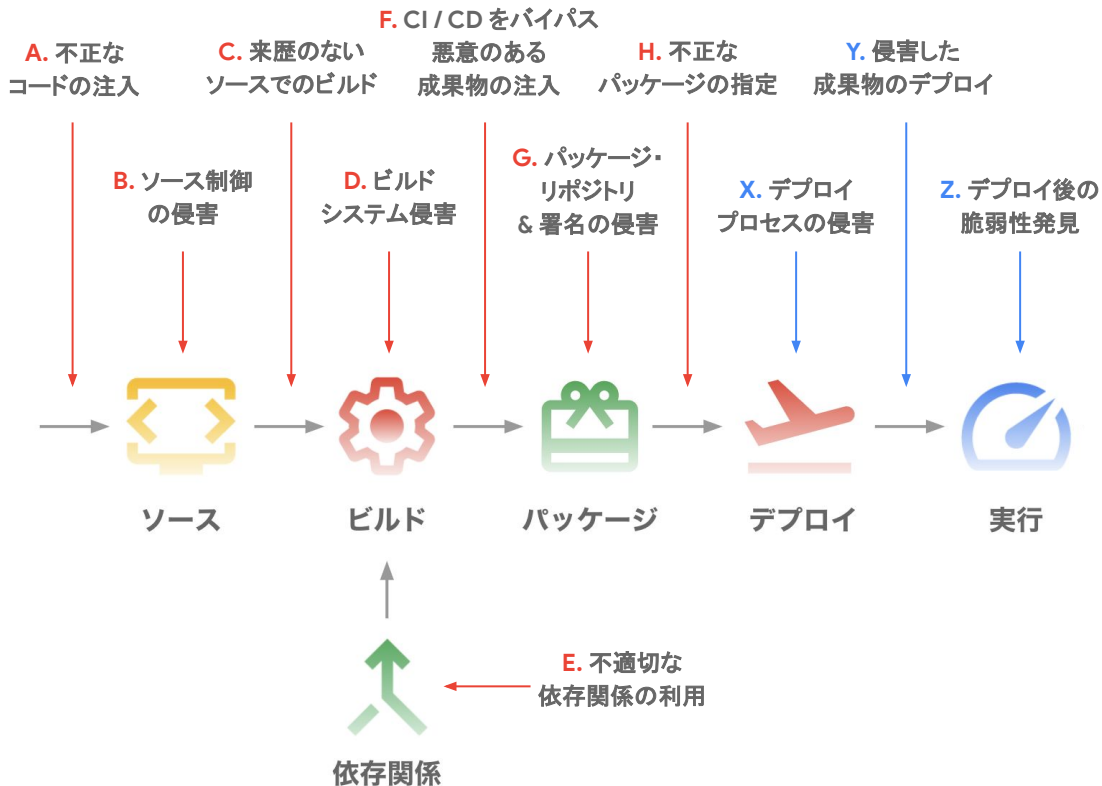
45% の組織は 2025 までサプライチェーン攻撃を受けると予測

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。



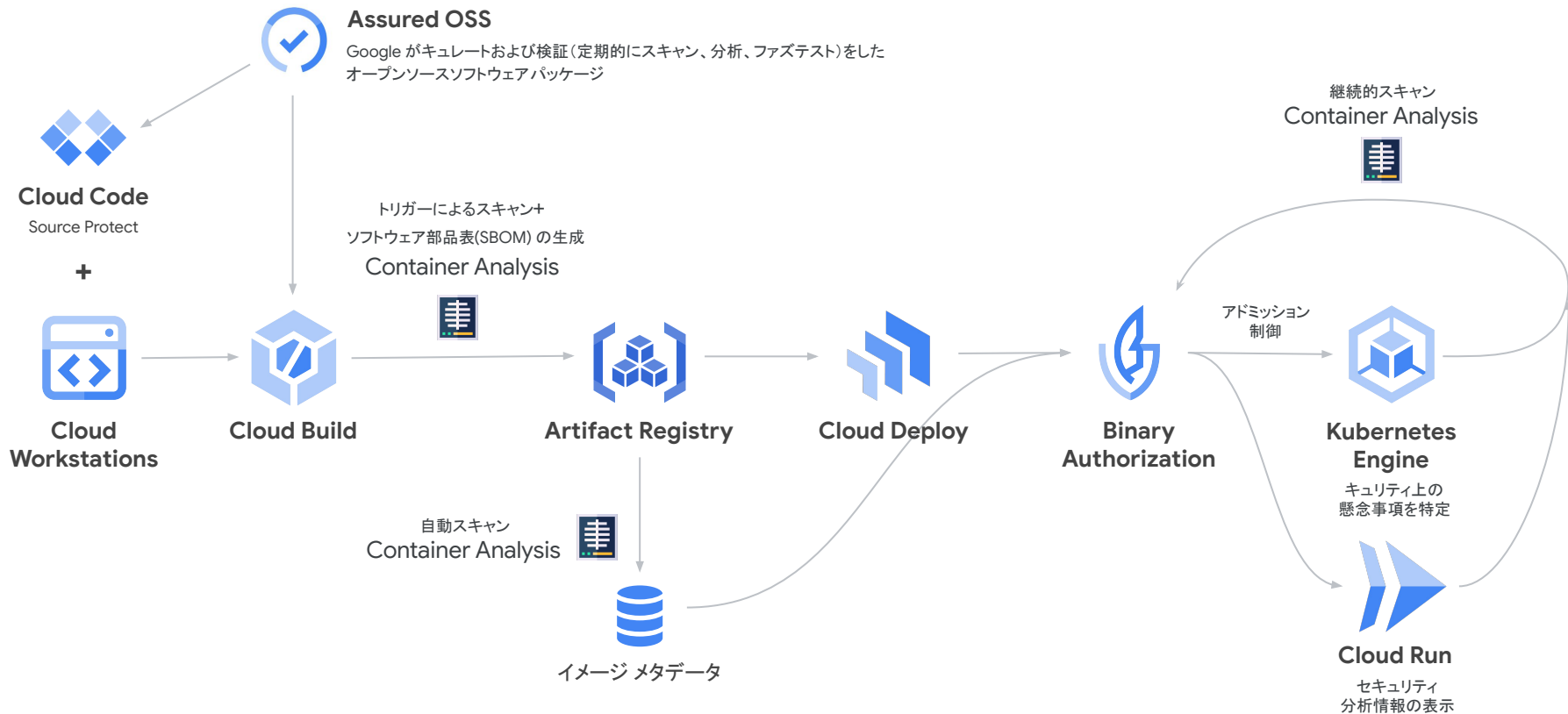
Software Delivery Shield

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.