



Developer Training

GitHub Advanced Security

日本語版

Agenda



The State of Security Today

今日のセキュリティの現状とGitHub Advanced Securityの機能を利用したセキュリティソリューションのビジョンを共有



Supply Chain

サプライチェーンにおける依存関係と適切な更新を管理



Source Code

ソースコードを保護するためにシークレット/コードスキャンを有効にして設定する方法





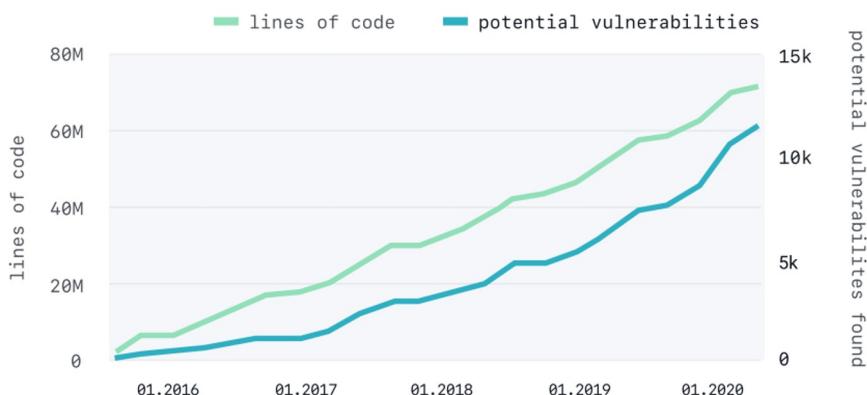
The State of Security Today

セキュリティの現状



The state of AppSec

Potential vulnerabilities found in source code scale with lines of code written



“ソースコードから発見された潜在的な脆弱性は、記述されたコード行数に比例する”

何十億ドルもの投資にもかかわらず…

85%のアプリケーションにセキュリティ上の問題が残っている

2020年に書かれたコードは、
2016年に書かれたコードと同様
に、セキュリティ問題を引き起こす
可能性が高い



Flaws in applications are consistently the #1 attack vector for breaches

Source: Verizon Data Breach Investigations reports 2016, 2017, 2018, 2019 and 2020.

アプリケーションの欠陥は、常に侵害の攻撃ベクトルの第1位である。

The state of AppSec

開発の現状からさらに遅れている



1:100 セキュリティチームメンバーと開発者



知識不足がAppSecの主な課題に

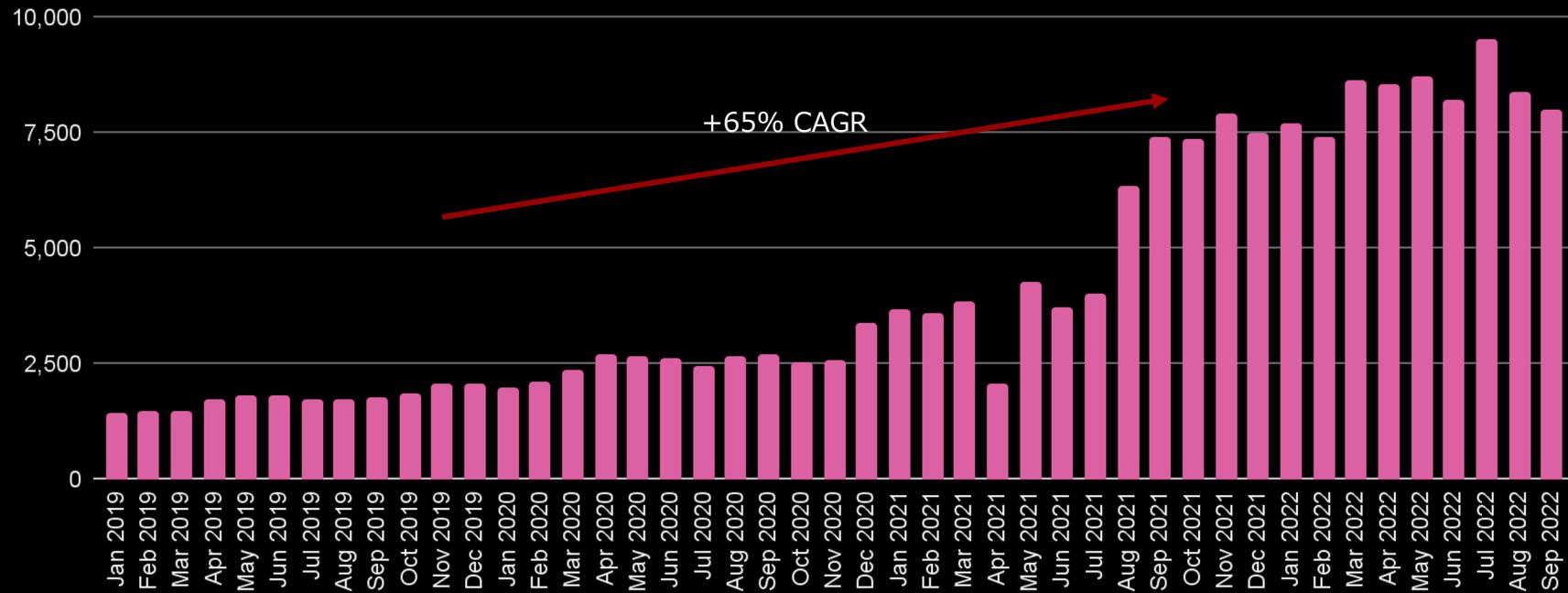


修復のトレンド (Remediation Trends)は停滞



Credentialの漏洩は増加傾向にあります

GitHub access tokens leaked in public repositories

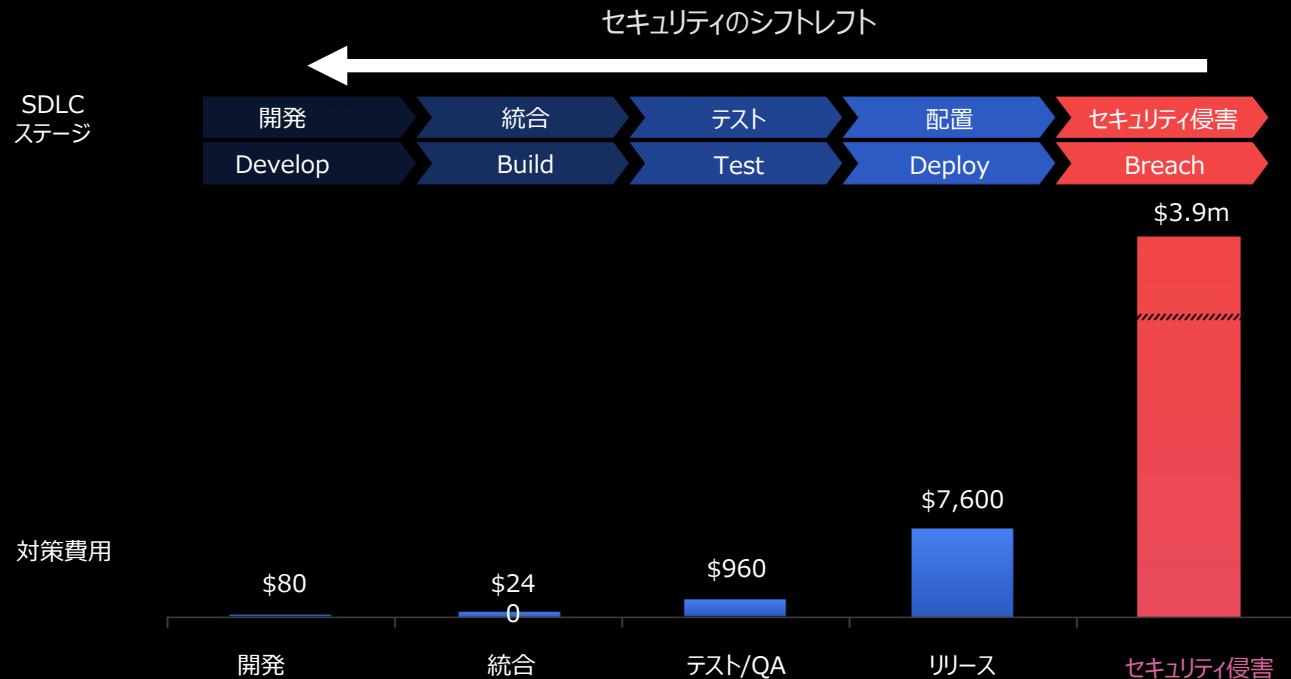


Source: GitHub data

July, 2022



誰もがセキュリティ対策を開発早期に実施したい…



Source: Ponemon Institute Cost of a Data Breach 2020

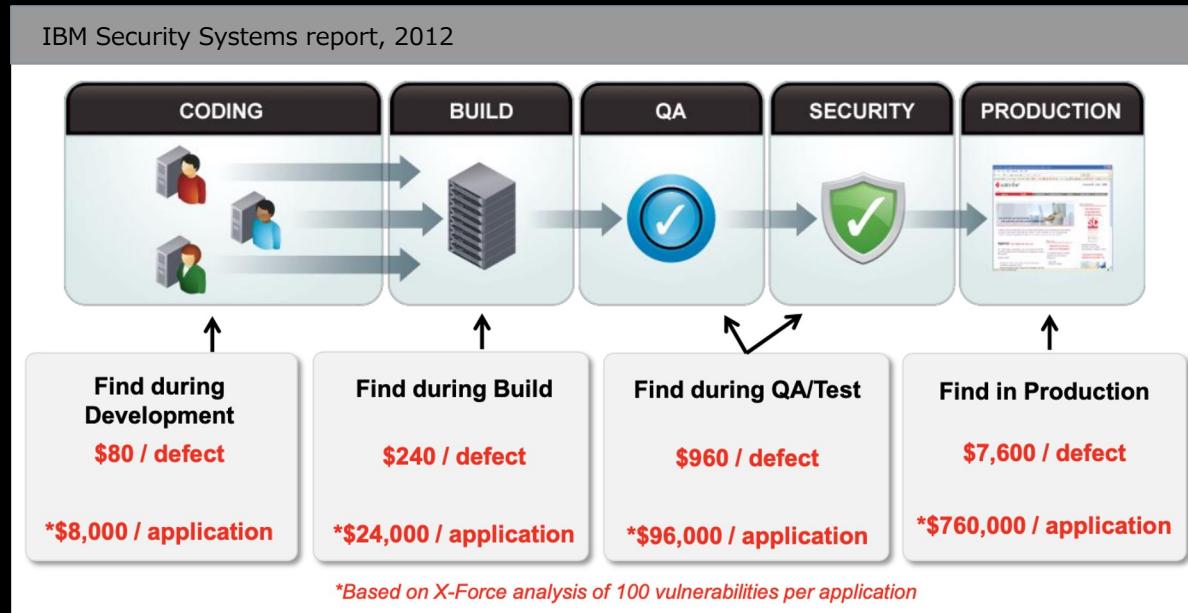
July, 2022

Advanced Security Developer Training

Japanese



ここ10年、企業はシフトレフトを試行しているものの...



GitHub believes that making this shift requires a developer-first approach to all our security products

GitHubは、このシフトを実現するには、すべてのセキュリティ製品において開発者ファーストのアプローチが必要だと考えています

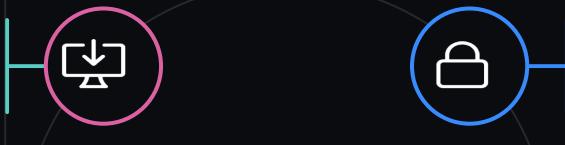


The GitHub Difference

開発者にセキュリティ権限を与え、数カ月ではなく数分で修正することが可能となります

Native

初期段階から出荷までの組み込み
自動デプロイ
詳細な変更履歴
すべてのコードにわたるセキュリティ状況の可視化



Trust-By-Design

静的コード解析
サプライチェーン解析
シークレットスキャン
セキュリティラボ
拡張可能なプラットフォーム

Fix in minutes

インコンテキストフィードバック
自動修正
専門用語を使わない言語
非難しない文化



Community Driven

数百万人の開発者とセキュリティ研究者に支えられている
主要な同業者によるレビュー



GitHub セキュリティソリューション



サプライチェーン

- Dependency graph
依存関係を視覚化
- Advisory database
依存関係に関わる脆弱性のデータベース
- Security alerts and updates
依存関係の脆弱性の通知と修正のプルリクエスト作成
- Dependency review
PRで新たな依存関係と脆弱性を検出



コード

- Secret scanning
リポジトリ内で漏洩される可能性があるAPIトークンや他のシークレットを検出
- Code scanning
CodeQLによる静的解析を提供開発ワークフローに統合、PRをトリガ



開発サイクル

- Branch protection
ブランチへのpushやPRによる統合における要件を強制
- Commit signing
全てのコミットについて署名を強制





Supply Chain





Dependency Graph



依存関係の視覚化

- 以下のマニフェスト、もしくはlockファイルの要約:
 - リポジトリに依存しているエコシステムとパッケージ
 - 依存関係にあるリポジトリとパッケージ
- サプライチェーンセキュリティの中核
- 直接、推移する依存関係
- 言語、パッケージのサポートを拡張



Dependabot

脆弱な依存関係や古い依存関係を自動的にアップデート



セキュリティとバージョン更新のための自動プルリクエスト

脆弱なコンポーネントや古いコンポーネントを監視することで、プロジェクトの安全性と最新性を維持します。アップデートの提案が見つかった場合、自動的に修正提案を含むプルリクエストをオープンします。

開発者のワークフローに統合

Dependabotは開発者のワークフローに直接統合され、摩擦のないエクスペリエンスと迅速な修正を実現します。



豊富な脆弱性データ

GitHubは、セキュリティ研究者、メンテナー、およびGitHub Advisory Databaseで発見可能なNational Vulnerability Databaseからのデータを使用して、サポートされているパッケージマネージャのパッケージの脆弱性を追跡します。



The screenshot shows a GitHub pull request titled "Bump proton-j from 0.12.0 to 0.30.0 in /maven #8". The pull request was opened by dependabot and merged into the master branch from dependabot/maven/maven/org.apache.qpid-proton-j-0.30.0. The message indicates that this automated pull request fixes a security vulnerability. It also notes that only users with access to security alerts can see this message. The pull request has 0 conversations, 1 commit, 0 checks, and 1 file changed. A comment from dependabot bot states: "Bumps proton-j from 0.12.0 to 0.30.0." Below the comment, it says "compatibility unknown". A note from Dependabot says: "Dependabot will resolve any conflicts with this PR as long as you don't alter it yourself. You can also trigger a rebase manually by commenting @dependabot rebase." At the bottom, there are sections for Dependabot commands and options, and a commit history showing the addition of the dependencies label.

作業を妨げない 通知



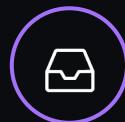
Email



User Interface



CLI



GitHub Inbox



Dependency Review

- 依存関係の追加、削除、編集を表現
- リリース日、評価、脆弱性情報をリスト表示
- PR上でリッチな差分表示
- Dependency Review GitHub ActionはPR上で依存関係のレビューを行い、脆弱性問題が当該リポジトリに混入することを阻止

The screenshot shows two views of GitHub's dependency review feature. The top view is a list of vulnerabilities found in a package-lock.json file, including 'follow-redirects' at version 1.13.0 (released 2 years ago) with a 'High severity' issue (GHSA-74fj-2j2h-c42q) and 'Exposure of sensitive information in follow-redirects'. The bottom view is a GitHub Actions log for a pull request, showing the 'dependency-review' job failing due to vulnerable packages.

Follow-redirects 1.13.0 released 2 years ago

Exposure of sensitive information in follow-redirects (GHSA-74fj-2j2h-c42q)
High severity Patched version: 1.14.7

Exposure of Sensitive Information to an Unauthorized Actor in follow-redirects (GHSA-pw2r-vq6v-hr8c)
Moderate severity Patched version: 1.14.8

chalk 4.1.2 released 9 months ago

is-buffer 1.1.6 released 5 years ago

axios updated to 0.21.2 released 8 months ago

follow-redirects updated to 1.14.9 released 2 months ago

loader-utils updated to 2.0.2 released 6 months ago

vue-loader-v16 updated to npm:vue-loader@16.8.3

is-buffer 1.1.6 released 5 years ago

Give feedback on dependency review

dependency-review failed 16 days ago in 4s

Set up job

Checkout Repository

Dependency Review

Error: This pull request introduces vulnerable packages.



Dependabot

Enabling Dependabot alerts (Dependabot通知有効化)

Reviewing the dependency graph(Dependency graphレビュー)

Viewing and managing results(結果参照と管理)

Enabling Dependabot security updates(セキュリティ更新有効化)

Reviewing dependency updates(セキュリティ更新レビュー)



Secret Scanning



Secret scanning

ハードコードされたシークレットの検索と管理



可能な限り早期にシークレットを特定する
シークレット（Azure シークレットを含む）が GitHub にプッシュされた瞬間に発見し、発見次第直ちに開発者に通知します。



Secret scanning partners コミュニティ
ポジトリに行われたすべてのコミットとその完全な git 履歴について、シークレットスキャンパートナーからのシークレットフォーマットを探します。



定義されたカスタムパターン
リポジトリ全体で Organization 内部のパターンをスキャンします



Public と private のリポジトリをサポート
Secret scanningは、Public リポジトリとPrivateリポジトリの両方に潜在的なシークレット脆弱性がないか監視する。

The screenshot shows the GitHub interface for a repository named 'dsp-testing / code-scanning-demo'. The 'Security' tab is selected. On the left, there's a sidebar with sections like Overview, Security policy, Security advisories, Dependabot alerts, Code scanning alerts (which is highlighted), CodeQL, and Detected secrets. The main area displays a code snippet from 'test.ts' with several lines highlighted in yellow, indicating detected issues. One specific line is shown in detail:

```
8  */
9  const sendRedirect = async (res: ServerResponse, url: string, statusCode: number) =>
10    res.statusCode = statusCode;
11    res.setHeader('Location', url);
```

The highlighted line is 'res.setHeader('Location', url);'. A tooltip for this line states: 'Untrusted URL redirection due to user-provided value.' Below the code, there's a table with columns for Tool, Rule ID, and Query. The entry for this alert is:

Tool	Rule ID	Query
CodeQL	js/server-side-unvalidated-url-redirection	View source

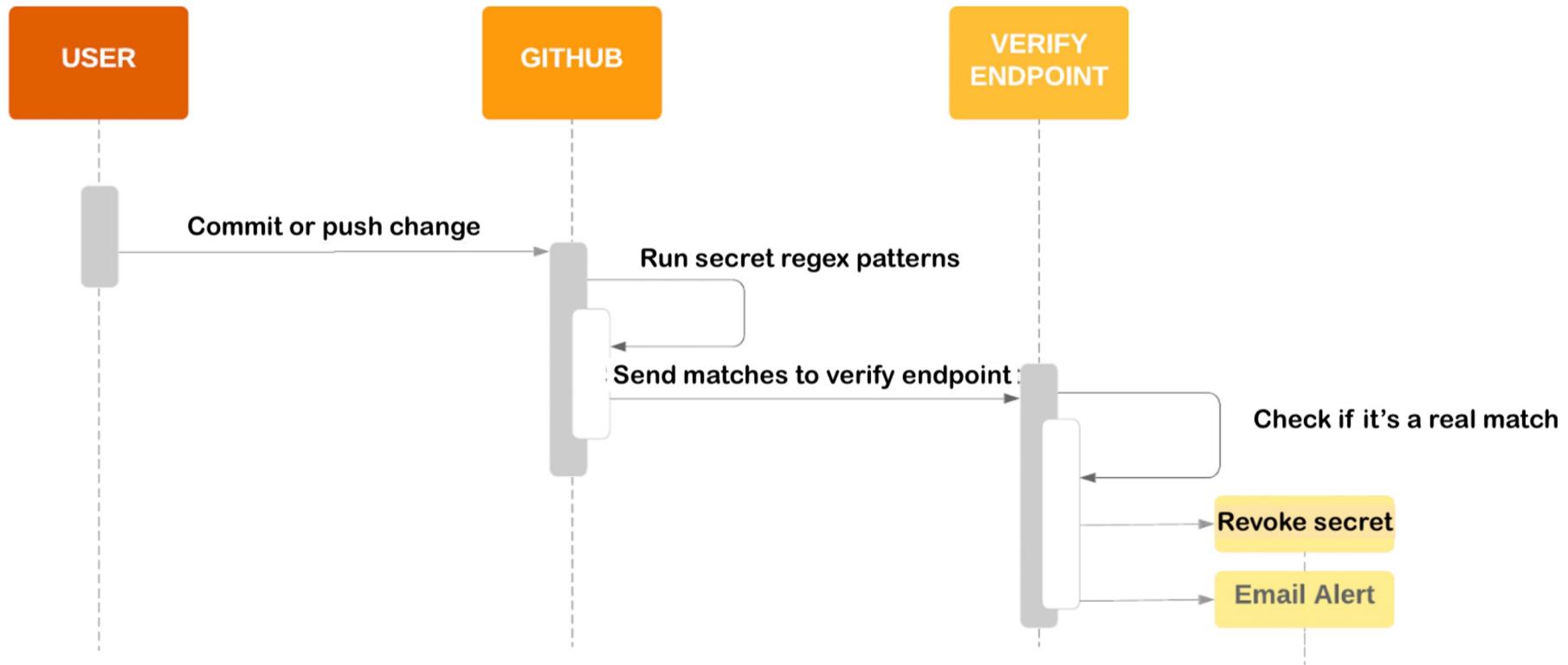
Below the table, a note reads: 'Directly incorporating user input into a URL redirect request without validating the input can lead to security vulnerabilities like SSRF attacks, unsuspecting users can be redirected to a malicious site that looks very similar to the legitimate one, which is controlled by the attacker.' There are also links for 'Show more' and 'First appeared in commit 1a36781 on Apr 9'.



Secret Scanning パートナー



Secret Scanningのプロセスダイアグラム



カスタムパターン

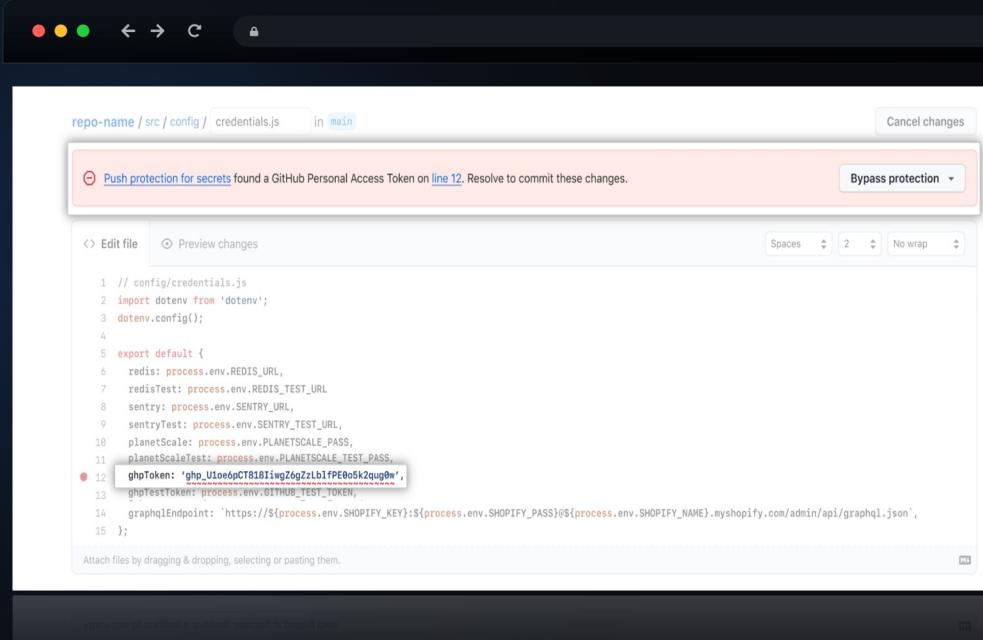
- 企業組織、リポジトリ単位でカスタムパターンを定義可能
- パターンは正規表現で指定し、HyperScanライブラリを使用

The screenshot shows a web-based configuration interface for creating a new custom pattern. The title bar reads "Security & analysis / New custom pattern".
Pattern name: A text input field contains "New custom pattern". Below it, a note says "This cannot be edited after saving."
Secret format: A text input field contains the regular expression "my_custom_secret_[a-z0-9]{3}".
Test string (required) - 1 match: A text input field contains "my_custom_secret_123".
At the bottom, there is a note: "Provide a sample test string to make sure your configuration matches the patterns you expect." A green button labeled "Save and dry run" is visible.



プッシュ保護

- Organizationやリポジトリに秘密がプッシュされないようにする
- Organizationレベルまたはリポジトリレベルで有効にできる



The screenshot shows a GitHub commit interface for a file named 'credentials.js'. A red warning box highlights a GitHub Personal Access Token on line 12. The token starts with 'ghp_U1oeopCT8181wgZ6gZt1b1FPEo5k2quq0w'. Below the code editor, there's a note: 'Attach files by dragging & dropping, selecting or pasting them.'

```
// config/credentials.js
import dotenv from 'dotenv';
dotenv.config();

export default {
  redis: process.env.REDIS_URL,
  redisTest: process.env.REDIS_TEST_URL,
  sentry: process.env.SENTRY_URL,
  sentryTest: process.env.SENTRY_TEST_URL,
  planetScale: process.env.PLANETSCALE_PASS,
  planetScaleTest: process.env.PLANETSCALE_TEST_PASS,
  ghpToken: 'ghp_U1oeopCT8181wgZ6gZt1b1FPEo5k2quq0w',
  ghpTestToken: process.env.GITHUB_TEST_TOKEN,
  graphQLEndpoint: 'https://${process.env.SHOPIFY_KEY}:${process.env.SHOPIFY_PASS}@${process.env.SHOPIFY_NAME}.myshopify.com/admin/api/graphql.json',
};
```





Secret Scanning

Enabling secret scanning (有効化)

Enabling push protection(push保護有効化)

Applying a custom pattern(カスタムパターン適用)

Viewing and managing results(結果参照と管理)

Excluding files from secret scanning(スキャン対象除外ファイル)

Managing access to alerts



Code Scanning



Code Scanning

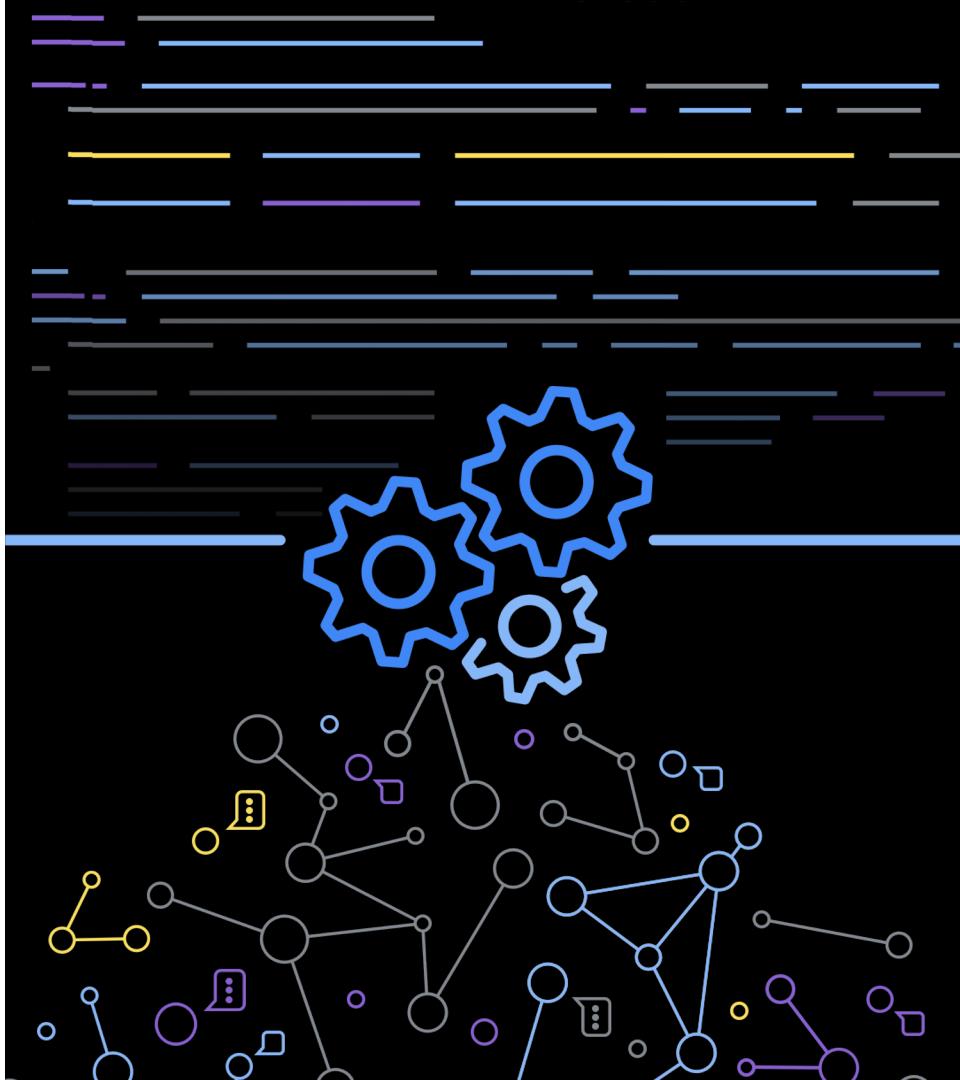
- CodeQL自動スキャンを利用してコードベースに統合する前に脆弱性を検出
- 開発ワークフローに直接結果を統合
- カスタムクエリ及び、GitHubコミュニティのクエリセットを実行
- 3rdベンダー等その他のSASTツール利用

The screenshot shows the GitHub Code Scanning interface for a repository named 'dsp-testing / code-scanning-demo'. The 'Security' tab is selected. On the left, there's a sidebar with sections like Overview, Security policy, Security advisories (0), Dependabot alerts (0), Code scanning alerts (1), CodeQL, and Detected secrets (0). The main area displays a file named 'test.ts' with a specific line highlighted in yellow: 'res.setHeader('Location', url);'. A tooltip for this line states: 'Untrusted URL redirection due to user-provided value. CodeQL'. Below the code, a detailed view of the alert shows the tool used (CodeQL), rule ID (js/server-side-unvalidated-url-redirection), and a query link. It also includes a description of the vulnerability: 'Directly incorporating user input into a URL redirect request without validating the input can facilitate phishing attacks. In these attacks, unsuspecting users can be redirected to a malicious site that looks very similar to the real site they intend to visit, which is controlled by the attacker.' There are also links for 'Show more' and 'Tool version'.



CodeQL: 革新的なセマンティックコード エンジン

- Oxford大学の30人の研究者が13年にわたり研究をベースにしたコード解析エンジン
- 脆弱性検出の精度を高めるためにクエリをカスタマイズ可能
- 特定の脅威トポロジーに適応するために簡単にカスタマイズ可能
- コミュニティ主導のクエリは世界中のセキュリティチームのプロジェクトに提供



Code Scanning: パートナー

GHEC

GHES

- Marketplaceの統合機能を使用し、コンテナ、Infrastructure as Code、その他の言語をコードスキャン機能を使ってスキャンすることも可能です

The screenshot shows the GitHub Marketplace interface with several listed items:

- Anchore Container Scan** by Anchore: Produce container image vulnerability and compliance reports based on the open-source Anchore container image scanner. [Set up this workflow](#)
- Codacy Security Scan** by Codacy: Free, out-of-the-box, security analysis provided by multiple open source static analysis tools. [Set up this workflow](#)
- CodeScan** by CodeScan Enterprises, LLC: CodeScan allows for better visibility on your code quality checks based on your custom rulesets. [Set up this workflow](#)
- CxSAST** by Checkmarx: Scan your code with Checkmarx CxSAST and see your results in the GitHub security tab. [Set up this workflow](#)
- DefenseCode ThunderScan** by DefenseCode: Scan your code with ThunderScan® SAST to detect security vulnerabilities in more than 30 programming languages. [View in marketplace →](#)
- Fortify on Demand Scan** by Micro Focus: Integrate Fortify's comprehensive static code analysis(SAST) for 27+ languages into your DevSecOps workflows to build secure software faster. [Set up this workflow](#)



SARIF形式に対応した 3rdパーティツールを統合

- 結果を統合するためにはSARIF形式で出力する必要あり
- Sarif 形式はGitHubアクション用 CodeQL解析、もしくはCodeQL CLIで対応済み
- 複数のSARIFファイルをアップロード可能

The screenshot shows the GitHub Code scanning interface. At the top, it displays a summary: "Latest scan yesterday", "Pull request #4", "Workflow CodeQL", "Lines scanned 61 / 61", "Duration 3m 13s", and "Result 0 alerts". Below this is a search bar with the query "Q is:open branch:main". Under the search bar, there are filters for "0 Open" and "0 Closed" issues. To the right of these filters are dropdown menus for "Tool", "Branch", "Rule", "Severity", and "Sort". A "Filter by tool" dropdown is open, showing "CodeQL" with a count of "0". Below the filters, a blue padlock icon with a checkmark is displayed, indicating that the branch hasn't been scanned yet. The text "This branch hasn't been scanned yet." is shown in bold, along with a link "For more information about code scanning, see [here](#)".





Code Scanning

Enabling code-scanning(有効化)

Reviewing a failed analysis job(失敗した解析ジョブのレビュー)

Customizing the build process in the CodeQL workflow(ビルドプロセスのカスタマイズ)

Reviewing and managing results(結果レビューと管理)

Triaging a result in a PR(PRの中で結果をトリアージ)

Customizing CodeQL configuration(コンフィグをカスタマイズ)

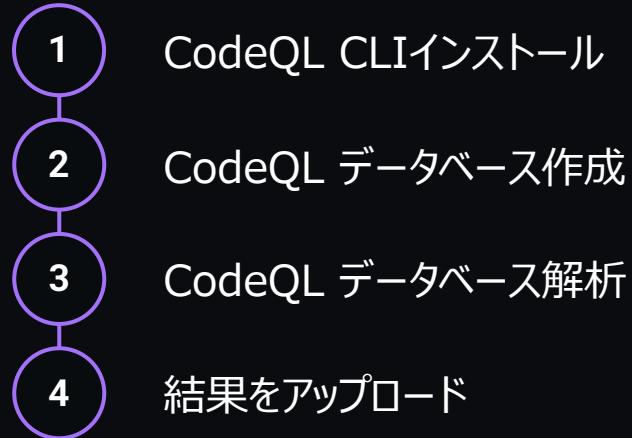
Adding your own code scanning suite to exclude rules(自身のコードスキャンを追加)



CodeQL CLI for Code Scanning



CodeQL 検出までの流れ





CodeQL

CodeQL CLIを使って始める

- 1 . codeql database create
2. codeql database analyze
3. codeql github upload-results

Q&A

