GitHub Actionsのセキュリティ実践と課題の混合研究法的分析

久保 佑介[1,2,a)]　金井 文宏[1]　秋山 満昭[3]　若井 琢朗[2]　森 達哉[2,4,5]

概要：継続的インテグレーション／継続的デリバリ (CI/CD) の普及に伴い，GitHub Actions はソフトウェア開発において広く活用されている．一方で，そのセキュリティ確保は重要な課題であり，GitHub は開発者向けにセキュリティプラクティスを公開している．しかし，それらが実際にどの程度実践され，開発者がどのような判断や認識のもとでそれらを実践しているかについては十分に理解されていない．本研究では，338,812 件の公開リポジトリを対象とした大規模な観測調査と 102 名の開発者へのユーザ調査を組み合わせた混合研究法的分析により，GitHub Actions におけるセキュリティプラクティスの実態を分析した．その結果，(1) セキュリティプラクティスが十分に実践されていないこと，(2) リポジトリの所有者のアカウント種別などの特性と実践状況に統計的な関連があること，(3) 実践を阻害する主要な要因として，認知不足，理解不足や誤解，運用コストやリソースに対する懸念の三点が存在することを明らかにした．さらに，これらの知見に基づき，ドキュメントの改善やプラットフォーム・ツールによる支援の必要性について議論する．

キーワード：GitHub Actions, セキュリティプラクティス

A Mixed-Methods Approach to Examining Security Practices in GitHub Actions

Yusuke Kubo[1,2,a)]　Fumihiro Kanei[1]　Mitsuaki Akiyama[3]　Takuro Wakai[2]　Tatsuya Mori[2,4,5]

Abstract: GitHub Actions has become a widely used and influential platform in modern software development due to the widespread adoption of Continuous Integration/Continuous Delivery (CI/CD). Ensuring the platform's security is a critical and ongoing challenge that has prompted GitHub to publish security practices for developers. However, it is still unclear how widely these practices are adopted and how developers perceive and apply them in practice. This study used a mixed-methods approach combining a large-scale measurement study of 338,812 public repositories and a use study of 102 developers to examine the state of security practices in GitHub Actions. Our findings show that (1) security practices are not widely adopted, (2) adoption is statistically associated with repository characteristics, such as the owner's account type, and (3) three main barriers hinder adoption: lack of awareness, lack of understanding and misconceptions, and cost and resource concerns. Based on these findings, we discuss the necessity of improving documentation and providing stronger support through platforms and tools.

1. はじめに

継続的インテグレーション／継続的デリバリ（CI/CD）は，現代のソフトウェア開発において不可欠な手法となり，コードのビルドやテスト，デプロイを自動化することで開発効率とリリース速度を向上させている．一方で，その普及

[1] NTT ドコモビジネス株式会社/NTT DOCOMO BUSINESS, Inc.
[2] 早稲田大学/Waseda University
[3] NTT 株式会社/NTT, Inc.
[4] 情報通信研究機構/NICT
[5] 理化学研究所 革新知能統合研究センター/RIKEN AIP
[a)] yuusuke.kubo@ntt.com

は新たな攻撃対象を生み出しており，2020 年 SolarWinds [1] や 2021 年の Codecov [2] のインシデントに示されるように，CI/CD を標的とした攻撃が深刻化している．

数ある CI/CD プラットフォームのうち，GitHub Actions [3] は GitHub リポジトリとの統合や設定の容易さから主流のサービスとして台頭している [4]．一方，GitHub Actions の利用拡大に伴い，セキュリティ上の課題が顕在化してきた．たとえば，2025 年の tj-actions/changed-files の事例 [5] では，リリースタグが改ざんされ悪意あるコードが注入されたことにより，このアクションをワークフローで利用していた多数のリポジトリに影響が及んだ．また，Koishybayev らの研究 [6] では，過剰な権限設定や外部コード実行などのリスクが体系的に指摘されている．

これらのセキュリティ脅威への対策として，GitHub は公式サイトでセキュリティプラクティスを公表している [7]．しかし，それらが実際の開発現場でどの程度採用され，またどのような要因が実践を左右しているのかは明らかではない．そのため，GitHub Actions のセキュリティプラクティスが十分に機能しているか否かは不透明な状況にある．

そこで本研究では，GitHub Actions におけるセキュリティプラクティスの実態を明らかにするため，次の 3 つの研究的問い (RQ) を設定した．

- **RQ1:** GitHub Actions のセキュリティプラクティスは，開発現場でどの程度実践されているのか?
- **RQ2:** セキュリティプラクティスを実践しているリポジトリと，実践していないリポジトリには，どのような特性の違いがあるのか?
- **RQ3:** 開発者のセキュリティプラクティスの実践を妨げる要因は何か?

これらに答えるため，我々はリポジトリデータの大規模観測調査と開発者へのユーザ調査を組み合わせた混合研究法を採用した．観測調査では，SEART-GHS データセット [8] から収集した 338,812 件の公開リポジトリを対象に，5 つのセキュリティプラクティスの実践を検出し，統計モデルによりリポジトリ特性との関連を分析した．ユーザ調査では，リポジトリデータセットから抽出した 102 名の GitHub Actions を使用している開発者を対象にオンラインアンケートを実施し，セキュリティプラクティスの実践を阻害する要因を調査した．

本研究の貢献は以下の通りである：

- GitHub Actions のセキュリティプラクティスの実践を判定する自動検出手法を開発した．本手法は公開リポジトリのファイルやメタデータといったオープンデータを利用して判定しており，開発者や研究者が検証や分析に活用できる．成果物は，開発者や研究者が利用できるよう公開を予定している．
- 338,812 件の公開リポジトリを分析し，GitHub Actions におけるセキュリティプラクティスの実装状況を定量的に評価した．その結果，実装率は 0.6 %〜52.9 %と低水準にとどまっており，さらにワークフロー数やリポジトリ所有者のアカウント種別といった特性が実装状況と統計的に有意な関連を持つことを明らかにした．
- 102 名の開発者に対するアンケート調査から，認知不足（21.3 %〜71.6 %がプラクティスを知らない），理解不足や誤解（一部の開発者は誤解により未実践），運用コストの増加とリソース不足（一部のプラクティスにおいて 25.3 %がコストを懸念）という 3 つの主要な阻害要因を特定した．
- 調査結果に基づき，ドキュメントにおける実践対象やリスクの明確化，プラットフォームによるプラクティス実践の自動検知と通知，開発環境への支援技術の統合を提案する．これらの提言は，プラットフォーム改善に資する具体的な知見として整理し，GitHub に共有する予定である．

2. 背景

2.1 GitHub Actions

GitHub Actions の中心機能は，リポジトリ内で定義されるワークフローにあり，開発プロセスを自動化するための単位として機能する．図 1 にワークフローファイルの例を示す．ワークフローは，リポジトリの `.github/workflows` 配下に YAML 形式のファイルとして定義され，プッシュやプルリクエストなどのイベントをトリガーとして実行される． ワークフローは，1 つ以上のジョブから構成され，それぞれのジョブが特定の処理を担う単位となっている．また，各ジョブは，複数のステップで構成されており，ステップはシェルコマンドの実行 (`run`) やアクションの呼び出し (`uses`) など，実際の処理内容を記述する最小の単位である．また，GitHub が提供する組み込み変数群である GitHub コンテキストを用いることで，イベント種別やリポジトリ名といった実行時のメタ情報にアクセスできる．これは，`${{ github.xxx }}` の形式で参照でき，条件分岐やコマンドの動的実行を可能にする．

アクションは，特定の処理を再利用可能な単位として定義するコンポーネントであり，開発者は `uses` を用いて実行できる．アクションの実装方式には JavaScript，Docker，Composite の 3 種類があり，リポジトリ内に定義して利用するほか，公開リポジトリやコンテナイメージを参照して利用できる．公開リポジトリから参照する場合は `owner/repo@ref` の形式を用い，タグ・ブランチ・コミット SHA により参照先（`ref`) を指定する．このような公開リポジトリから利用するアクションを「パブリックアクション」と呼ぶ．なお，パブリックアクションには，GitHub 公式のアクション（例：`actions/checkout`）だけでなく，外部の開発者や組織が公開したものも含まれる．本研究では，GitHub 公式もしくは利用者以外の第三者が開発した

ものを，「サードパーティアクション」と定義する．アクションは，GitHub Marketplace を通じて公開されており，開発者はそこから容易に検索・利用できる．Marketplace 上では，GitHub が開発者を検証したアクションに Verified バッジ が付与される仕組みがあり，開発者がアクションを選定する際の信頼性指標として使われている．

```yaml
name: Example Workflow # ワークフロー名
on:
  pull_request: # ワークフローのトリガーとなるイベント
    branches:
      - main # 対象ブランチ
jobs:
  build: # ジョブ名
    runs-on: ubuntu-latest  # 実行環境
    steps:
      - name: Check out code
        uses: actions/checkout@v4 # アクションの使用
        with: # アクションへの入力値
          path: main
      - name: Show Pull Request Title
        env: # 環境変数の設定
          PR_TITLE: ${{ github.event.pull_request.title
          ↪  }} # GitHubコンテキストの使用
        run: echo "$PR_TITLE" # コマンドの実行
```

図 1　ワークフローファイルの例

2.2　関連研究

CI/CD におけるセキュリティリスクや脆弱性に関する体系的な調査は，GitHub Actions のセキュリティに関する議論において重要な先行研究である．Koishybayev ら [6] は，CI プラットフォームにおける権限設定の不備や不正コードの実行といったセキュリティリスクを体系的に整理し，GitHub Actions と他プラットフォームを比較評価した．また，GitHub Actions のワークフローの実態調査を通じて，99.8%のワークフローが過剰な権限を持ち，さらに 23.7%が悪意あるプルリクエストに起因する攻撃リスクを抱えることを示した．一方，Gu ら [9] は CI ワークフローにおけるトークン管理の観点から，主要な CI プラットフォームを分析し，トークン漏洩，過剰な権限設定，長期間有効なトークンといったリスクがリポジトリへの不正アクセスや改ざんにつながることを実証した．

また，GitHub におけるセキュリティ対策の利用実態についても調査が進められている．Ayala ら [10] は，GitHub Advisory Database に登録された脆弱性を持つ OSS プロジェクトのメンテナを対象に，アンケート調査（N=80）と半構造化インタビュー（N=22）を実施した．その結果，多くのメンテナがサプライチェーンの信頼性不足や脆弱性管理の自動化の欠如に直面しており，その背景には認識不足や機能の複雑さがあることが明らかにした．

既存研究は主に，CI/CD プラットフォーム のリスクや脆弱性 [6], [9]，あるいは GitHub というコードホスティングサービスにおけるセキュリティ機能 [10] に焦点を当てており，CI/CD プラットフォームのセキュリティプラクティスを対象とした研究は存在しない．さらに，観測調査とユーザー調査を組み合わせた研究も存在せず，CI/CD セキュリティの包括的な理解は十分とはいえない．

3.　調査方法

3.1　混合研究法の概要

本研究では，GitHub Actions におけるセキュリティ実践の実態と，その導入を妨げる要因を明らかにするため，混合研究法を採用した．具体的には，大規模リポジトリデータを対象とした観測調査と，開発者を対象としたユーザ調査を組み合わせた．観測調査では，セキュリティプラクティスの実践率（RQ1）と，リポジトリ特性との関連（RQ2）を定量的に分析する．ユーザ調査では，開発者の認識や導入障壁（RQ3）を調査し，定量・定性の両面から分析を行う．

3.2　セキュリティプラクティス

GitHub が公開する 16 種類のセキュリティプラクティス [7] のうち，観測調査およびユーザ調査の双方において一貫して調査可能な 5 種類を選定し，調査対象とした．以下では，各プラクティスに付与した識別子と名称とともに概要を説明する．なお，本研究におけるセキュリティプラクティスの内容は，2025 年 6 月 12 日時点で公開されていた情報に基づいている．

P1: CODEOWNERS. CODEOWNERS は，リポジトリ内の特定のファイルやディレクトリに責任者を割り当てる機能である．これにより，変更は必ず指定レビュアーの確認を経るため，不正やミスを防止できる．設定は CODEOWNERS ファイルにルール（対象パス，レビュアー）を記述することで行われ，該当箇所の変更時に自動的にレビュワーが割り当てられる．

P2: Mitigating Script Injection. スクリプトインジェクションは，攻撃者が GitHub コンテキストに悪意のあるコマンドを挿入し，ワークフローがそれをコードとして実行することで発生する．これを防ぐために，コマンド実行（`run`）で GitHub コンテキストを直接使用する方法は非推奨とされている．代わりに，コンテキストを (1) JavaScript アクションに入力値（`with`）として渡す，または (2) 環境変数（`env`）に割り当てた上でその環境変数をコマンド実行（`run`）に使用する，という 2 つの方法が推奨されている．

P3: OpenSSF Scorecard. Scorecard は，リポジトリを分析し，セキュリティリスクをスコア化するツールである．ワークフロー設定や依存関係，ブランチ保護など複数の観

点で評価を行い，開発者にセキュリティレベルと改善点を提示する．パブリックアクション（ossf/scorecard）やワークフローのテンプレートが公開されており，これらを使用することで容易に導入できる．

P4: Pinning Third-party Actions. サードパーティアクションが悪意を持って作成された場合や侵害された場合，その実行により機密情報の漏洩やリポジトリの不正操作といったリスクが生じる．このリスクを低減するために，アクションの参照（owner/repo@ref の ref）に関する 2 つの方法が推奨されている [*1]．：

- **P4-1**: 完全なコミットハッシュ（SHA）で参照する．将来の変更や改ざんを防ぎ，常に同一コードの実行を保証できる．最も信頼性が高いが，ハッシュ値を確認して記述する手間がある．
- **P4-2**: 開発者が信頼できる場合に限り，タグで参照する．可読性や簡便性に優れる一方，タグは付け替え可能であり，侵害時には悪意あるコミットへ改ざんされるリスクがある．

P5: Dependabot. ワークフローで利用するアクションの継続的な管理・更新はセキュリティ維持に不可欠であり，その手段として GitHub Actions では Dependabot の利用が推奨されている．Dependabot は，設定ファイルで github-actions を管理対象として指定することで，アクションに重要な更新があった際に自動でプルリクエストを作成し，開発者に更新を促す仕組みを提供する．

3.3 観測調査

3.3.1 リポジトリデータセット

本研究では，SEART-GHS [8] を元にデータセットを作成し，観測調査に使用した．作成の手順を以下に示す．

(1) SEART-GHS から GitHub リポジトリのリストを取得する．この際，分析対象として重複する Fork リポジトリや実質的にリスクが存在しない Archive リポジトリは除外する．
(2) GitHub API を使用して，各リポジトリのワークフローの有無を調べる．
(3) ワークフローが存在する場合，GitHub Actions を使用しているリポジトリと判定し，ローカルのサーバにクローンする．
(4) クローンしたリポジトリ内にワークフローファイルが存在しない場合は除外する．これは，GitHub Pages などの機能を利用している場合，実際にはファイルがなくとも GitHub API 上ではワークフローが存在すると判定されるためである．

我々は, 2025 年 5 月 12 日に, SEART-GHS から 1,675,884 件のリポジトリデータを収集した．このデータセットから，GitHub Actions を使用しているリポジトリを特定し，2025 年 6 月 6 日から 7 日にかけて，合計 338,812 件のリポジトリをクローンした．これは SEART-GHS 全体の約 20.2%に相当する．

3.3.2 自動検出手法

RQ1 を明らかにするために，セキュリティプラクティスの実践状況を自動的に検出する手法を開発した．本手法は，GitHub の公式ドキュメントおよび関連機能・ツールの仕様に基づき定義されたパターンに基づいて設計した．また，各プラクティスの検出手法は，実践対象となるリポジトリの判定と，実際に実践されているかの判定の 2 つから構成されている．表 1 に，本研究における判定条件を示す．

3.3.3 統計分析

RQ2 を明らかにするために，ロジスティック回帰分析を実施した．分析では，各プラクティスの実践有無をバイナリ変数（1: 実践，0: 未実践）に変換して従属変数とした．一方，独立変数には，SEART-GHS や GitHub API から取得したコントリビュータ数やコミット数などのリポジトリの特徴量を使用した．

本分析の主眼はモデルの予測性能ではなく，オッズ比を通じてどのようなリポジトリ特性がセキュリティプラクティスの実践に影響を与えるかを把握する点にある．オッズ比の解釈は独立変数の種類によって異なる．本分析においては，連続変数に対するオッズ比が 1 より大きいことは，変数の値とセキュリティプラクティスの実践の間に正の相関があることを意味する．同様に，バイナリ変数およびカテゴリ変数に対するオッズ比が 1 より大きいことは，比較元カテゴリと比較してセキュリティプラクティスが実践されていることを意味する．

ただし，オッズ比の大きさ（効果量）が必ずしも統計的有意性を意味するわけではない．一般に，統計的有意性を評価する際には p 値が用いられる．ただし，サンプルサイズが大きい場合には，ごく小さな差異であっても p 値が有意水準を下回る可能性がある．このため，p 値だけに依存した判断は適切ではない．そこで本研究では，統計的有意性を評価するために, p 値と信頼区間の両方を用いた．具体的には，p 値が 0.05 未満であり，かつ 95 %信頼区間が 1 を含まない場合を，統計的に有意と見なした．

3.4 ユーザ調査

3.4.1 質問紙設計

RQ3 に回答するために，GitHub Actions の利用経験のあるソフトウェア開発者を対象としたアンケート調査を実施した．アンケートでは，参加者が関与する GitHub リポジトリの基本的な属性や統計情報，GitHub Actions の主な用途や運用方法を質問した後に，セキュリティプラクティスに関する質問を行った．セキュリティプラクティスに関

[*1] ユーザ調査においては P4 を P4-1 と P4-2 に分割して扱うため，サブ識別子を付与する．

表 1 セキュリティプラクティスごとの判定条件

	対象判定	実践判定
P1	コントリビュータが 1 人以上存在	CODEOWNERS ファイルが有効なディレクトリ (`.github/`, リポジトリルート, `docs/`) に存在し，当該ファイルが 1 つ以上のルールを含む
P2	少なくとも 1 つ以上のワークフローで，`run`（コマンド実行），`with`（アクションの入力値），`env`（環境変数）のいずれかを通じて GitHub コンテキストを使用	すべてのワークフローにおいて，`run`（コマンド実行）で GitHub コンテキストを直接使用する使い方をしていない
P3	パブリックリポジトリである	少なくとも 1 つのワークフローで，`ossf/scorecard` アクションを使用
P4	少なくとも 1 つのワークフローで，サードパーティアクションを使用	使用されるすべてのサードパーティアクションの参照方法が，(1) 40 文字の完全な SHA で参照，または (2) Verified バッジ付きアクションをタグで参照，のいずれかに該当
P5	少なくとも 1 つのワークフローで，パブリックアクションを使用	`dependabot.yml` または `dependabot.yaml` が有効なディレクトリ (`.github/`) に存在し，当該ファイルにおいて `github-actions` を管理対象として設定

する質問では，プラクティスの内容を説明する文章を提示した後に，その実践状況を尋ねた．プラクティスを実践していないと回答した参加者には，その理由を選択肢形式で尋ねた．なお，選択肢において「その他」を選んだ参加者には自由記述形式で具体的な理由の記述を求めた．これらの質問は，3.2 節で説明した各プラクティス（P1〜P5）ごとに行った．ただし P4 については，P4-1 と P4-2 に細分化した．両者はどちらもアクションのセキュアな参照方法に関するプラクティスであるが，具体的な実践内容が異なり，それぞれへの開発者の認識が異なることが予想されることから区別したものである．上記に加えて，セキュリティプラクティスを実践するうえで重視する側面についても選択肢形式（複数選択可能，最大 3 つまで選択可能）で質問した．この質問は，個別のプラクティスではなくプラクティス全体に対して適用した．

アンケートにおける質問や選択肢は，既存研究 [10]，および 5 名の開発者へ対して行われたパイロットインタビューの結果を元に設計した．また，8 名の開発者へのパイロット調査を通じて，質問紙や選択肢のレビューと修正を繰り返し行い，アンケートの回答負荷の低減に努めた．アンケートは日本語で作成された後に英語に翻訳され，パイロット調査の参加者 2 名（英語ネイティブおよび日英バイリンガル）による言語表現の確認を行なった．

3.4.2 参加者募集

我々のユーザ調査は，Qualtrics を用いて 2025 年 7 月に実施した．我々は，GitHub 上から収集した開発者の連絡先へ，アンケート回答リンク付きの募集メールを送信することで参加者を募集した．ただし，不特定多数の開発者に調査を依頼することは，調査目的として不適切な参加者（例：GitHub Actions を全く利用していない開発者）が募集されてしまうことや，大量の開発者への募集メールが送信されることによる “調査疲れ” を誘発する懸念がある．そのため，以下の条件で募集対象の開発者の絞り込みを行った．まず， 3.3.1 節で説明したデータセットに含まれるリポジトリの中で，コントリビューションの数が上位 100 に入る 1,439,072 件の GitHub アカウントを抽出した．そこから，(1) プロフィール上で明示的に連絡先メールアドレスを公開しており，(2) GitHub Actions の設定に関わっている，(3) Bot ではないアカウント，に該当する 3,405 件を抽出した．さらに，最も貢献しているリポジトリの所有者がユーザアカウントか組織アカウントかで均等に分かれるよう，抽出された GitHub アカウントに対するランダムサンプリングを実施した．最終的に 1,800 件の開発者に対して募集メールを送信し，102 件の有効な回答を収集した．本調査では，参加者への金銭的な報酬は提供せず，希望者に結果の概要を後日共有することをインセンティブとした．

3.4.3 データ分析

我々は各設問の回答に対して，各選択肢が選ばれた件数/割合を算出した．回答が無回答であった場合，それらは割合を算出する際の母数から除外した．なお，参加者の数が限られていることから，我々は量的分析において統計検定を行わず，件数/割合の算出および比較のみを行った．これは，不十分な検出力の統計検定が行う際の問題 [11] を避けるためである．セキュリティプラクティスを実践していない理由において「その他」を選択した人の自由記述回答は，以下の手順で 2 名の著者によってコーディングを実施した．まず，各コーダーは回答にコードを付与することで独立にコードブックを作成した．その後，コーダー同士で各自のコードブックを比較し，議論を通じてコーディングの不一致を解消した．この対話的プロセスにより，コーダー間の合意を反映した最終コードブックを作成した．なお，コーディングは統計的な一致ではなくコーダー間の合意に基づいて実施されたため，既存研究 [12] における推奨に従い，コーダー間信頼性の報告は行わない．

表 2　セキュリティプラクティスごとの実践率

セキュリティプラクティス	対象数	実践率
P1 CODEOWNERS	337,163	7.1% (23,852)
P2 Mitigating Script Injection	43,516	52.9% (23,005)
P3 OpenSSF Scorecard	338,812	0.6% (1,965)
P4 Pinning Third-party Actions	233,124	16.2% (37,693)
P5 Dependabot	332,928	10.7% (35,588)

4. 調査結果

4.1 RQ1. セキュリティプラクティスの実践率

各セキュリティプラクティスの実践率を表 2 に示す．ここで示す実践率は，各プラクティスごとに実践対象と判定されたリポジトリを分母として算出している．セキュリティプラクティスごとの実践率は 0.6〜52.9%とばらつきがあるが，すべてのセキュリティプラクティスが十分に実践されていなかった．特に，セキュリティ機能およびツールの利用に関するセキュリティプラクティス（P1，P3，P5）の実践率は 11%以下であり，顕著に低いことがわかった．

4.2 RQ2. リポジトリ特性と実践傾向

ロジスティック回帰分析の結果を表 3 に示す．なお，スター数，コードサイズ，コミット数，リポジトリの存続日数といった変数も同様に分析に含めたが，いずれのセキュリティプラクティスに対しても統計的に有意な結果は確認されなかった．そのため，これらの変数は表から省略した．

分析の結果，最近の活動が見られる，すなわち継続的にメンテナンスされているリポジトリほど，セキュリティプラクティスが実践されている傾向が明らかになった．さらに，ワークフローの数が増えると，ワークフローの構成に関するプラクティス（P2，P4）が実践されにくくなる傾向が見られた．これは，管理すべきワークフローが増加することで，それらのプラクティスを安定的に適用することが難しくなる可能性を示している．また，コントリビュータ数では明確な傾向が確認できなかった一方で，ワークフローの開発者数では一定の関連を確認した．これは，ワークフローの実装や維持に携わる開発者がセキュリティプラクティスの実施において中心的な役割を果たしている可能性が考えられる．加えて，リポジトリのオーナータイプはセキュリティプラクティスの実践に対して顕著な結果を示した．特に，組織アカウントが所有するリポジトリでは，セキュリティ関連の機能やツールの利用に関するプラクティス (P1，P3，P5) が実践されていた．これは，組織的なガバナンスがセキュリティプラクティスの実践を促進している可能性を示す．

4.3 RQ3. 実践を妨げる要因

表 4 に，ユーザ調査における参加者のセキュリティプラクティスの実践率，およびセキュリティプラクティスを実践していない上位の理由を示す．

認知不足．プラクティスを実践していない参加者の 21.3%〜71.6%が，そもそもそれらの存在を知らなかったと回答しており，認知不足は GitHub Actions におけるセキュリティプラクティスが実践されない主要な要因の一つであると言える．また，P4-1 を実践していない理由は他のプラクティスと傾向が異なっており，保守/運用コスト増加への懸念が 2 番目に多かった．上記に関連して，パイロットインタビューの参加者から，P4-1 を実践すると，参照先のアクションが更新された際に手動での更新対応が必要となり，運用の負荷が高まる点が指摘された．加えて，P4-1 を実践しない理由に対する自由記述回答において，リソース不足を挙げている回答があった．このような実践コストやリソース不足への懸念も，セキュリティプラクティスが実践されない理由の一つであると言える．

理解不足および誤解．セキュリティプラクティスを実践していない理由に対する自由記述回答にて，それらが適用対象外であるという意見や，開発プロセスに悪影響を及ぼすという懸念がみられた．一方で，それらの中にはセキュリティプラクティスの効果や適用対象の誤解により，結果としてそれらが実践されていないケースが含まれていた．例えば，P5 を実践していない理由として，Dependabot が特定の言語をサポートしていない点を挙げている自由記述回答が 3 件あった．このプラクティスは GitHub Actions における依存アクションの更新に関するものであり，言語のサポート状況には依存せず，パブリックアクションを利用している全てのリポジトリで適用可能である．このような事例は少数であるが，セキュリティプラクティスに対する理解不足や誤解が，実践を妨げる要因となり得る事を示唆している．

コストおよびリソースに関する懸念．参加者がセキュリティプラクティスを実践する際に最も重視するのは「保守/運用コストの低さ」（N=81, 79.4 %）であり，他の要素と比較して選ばれた割合が特に高かった．これは，一部のセキュリティプラクティスにおいて，「保守/運用コスト増加」が実践を妨げる主要な要因として挙げられていた事と一貫した結果である．2 番目に多かった要素は「容易に設定/適用可能であること」（N=47, 46.1%），3 番目は「関連するセキュリティリスクを明確に軽減できること」（N=38, 37.3%）であった．3 番目の要素に関連する意見として，パイロットインタビューのある参加者は，「各セキュリティプラクティスがどのようなセキュリティリスクを解決するのか不明確である」と述べていた．これらの結果から，セキュリティプラクティスがどのようなリスクに対応するかが明確であることの重要性が示唆される．

5. 議論

本研究の調査により，GitHub Actions のセキュリティ

表 3 セキュリティプラクティスの実践に対するロジスティック回帰分析の結果

独立変数	P1	P2	P3	P4	P5
コントリビュータ数	**1.0017*** [1.0014, 1.0020]**	**.9990*** [0.9986, 0.9993]**	**1.0023*** [1.0016, 1.0030]**	.9996* [0.9992, 0.9999]	**.9982*** [0.9979, 0.9986]**
直近の活動の有無	**1.4154*** [1.3745, 1.4574]**	.9825+ [0.9436, 1.0229]	**2.6913*** [2.4159, 2.9982]**	**1.0750*** [1.0509, 1.0997]**	**2.3390*** [2.2819, 2.3976]**
ワークフロー数	**1.0518*** [1.0479, 1.0557]**	**.9729*** [0.9683, 0.9775]**	**1.0305*** [1.0253, 1.0358]**	**.9725*** [0.9680, 0.9769]**	**1.0586*** [1.0548, 1.0624]**
ワークフローの開発者数	**1.1009*** [1.0967, 1.1050]**	**.9808*** [0.9766, 0.9851]**	**1.0442*** [1.0381, 1.0502]**	**1.0141*** [1.0104, 1.0178]**	**1.1242*** [1.1200, 1.1285]**
オーナータイプ	**.2500*** [0.2409, 0.2596]**	**1.0549* [1.0097, 1.1022]**	**.3556*** [0.3157, 0.4004]**	**1.0617*** [1.0372, 1.0867]**	**.8969*** [0.8746, 0.9197]**

表内の数値はオッズ比，角括弧内の数値は信頼区間を示す．p 値の判定基準：+ $p > .05$; * $p < .05$; ** $p < .01$; *** $p < .001$．太字は統計的に有意であり，オッズ比が 1.000 から 0.001 以上離れている場合．コントリビュータ数，ワークフロー数，ワークフローの開発者数（連続変数）．直近の活動の有無（バイナリ変数）：2025/1/1 以降にコミット履歴がある（ベースライン）．オーナータイプ（カテゴリ変数）：リポジトリの所有者がユーザアカウント（ベースライン），組織アカウント．有意水準: + $p > .05$; * $p < .05$; ** $p < .01$; *** $p < .001$.

表 4 プラクティスの実践率およびプラクティスを実践しない理由

ID	実践率	プラクティスを実践しない理由	参加者の割合（理由ごと）
P1	17.6%	認知不足	41.4% (N=29)
	(N=18)	不要/過剰	40.0% (N=28)
		保守/運用コスト増加	4.3% (N=3)
P2	46.1%	認知不足	50.0% (N=19)
	(N=47)	不要/過剰	31.6% (N=12)
		保守/運用コスト増加	5.3% (N=2)
		解決されるリスクが不明確	5.3% (N=2)
P3	5.9%	認知不足	71.6% (N=63)
	(N=6)	不要/過剰	20.5% (N=18)
P4-1	19.4%	不要/過剰	30.7% (N=23)
	(N=19)	保守/運用コスト増加	25.3% (N=19)
		認知不足	21.3% (N=16)
P4-2	61.2%	認知不足	38.7% (N=12)
	(N=60)	不要/過剰	22.6% (N=7)
		保守/運用コスト増加	12.9% (N=4)
P5	43.4%	認知不足	36.2% (N=17)
	(N=43)	不要/過剰	25.5% (N=12)
		保守/運用コスト増加	8.5% (N=4)

実践率は，セキュリティプラクティスの実践状況を尋ねる質問において，実践していると回答した人の割合を示す．セキュリティプラクティスを実践しない理由には，上位 3 位以内かつ 2 名以上が選択した理由のみを示す．セキュリティプラクティスを実践していない理由の割合は，当該プラクティスを実践していないと回答した参加者に対する割合を示す．

プラクティスは十分に普及しておらず，その背景には複数の阻害要因が存在することが明らかになった．これらの課題を克服し，実践を促進するためには具体的な対応が求められる．本論文では，GitHub 公式ドキュメント [7] の改善およびプラットフォームやツールによる支援の 2 つの観点から推奨事項を提示する．

5.1 ドキュメントの改善

セキュリティプラクティスの実践対象の明確化．セキュリティプラクティスが十分に実践されない一因として，技術的な認識の誤りがある．ユーザ調査でも，Dependabot に関する誤解の事例が確認された．GitHub の機能に関するドキュメントでは，「この機能を使用できるユーザーについて」の項目を設け，対象ユーザや適用条件を明示している場合がある．このような記述は，開発者が自身のプロジェクトにとって機能の適用可否を判断する助けとなる．セキュリティプラクティスについても同様に，推奨される状況や条件を明記することで，導入に関する判断を支援できる．具体的には，表 1 に示した対象判定の内容がその条件に相当する．

セキュリティプラクティスとリスクの対応付け．ユーザ調査において，セキュリティプラクティスを実践する上で重視する要素として，「関連するセキュリティリスクを明確に軽減できること」を挙げた参加者が 37.3% 存在した．また，パイロットインタビューにおいて，対応するリスクが明確ではない点について指摘があった．これらの結果から，各セキュリティプラクティスがどのリスクに対応しているのかを明示することの重要性が示唆される．リスクを明確に示すことは，開発者が当該プラクティスの価値を理解し，導入を判断する際の重要な情報となる．

5.2 プラットフォームとツールによる支援

セキュリティプラクティス実践の自動検知と通知．GitHub Actions が実践されていない理由として，「知らなかった」という回答が最も多く，21.3%〜71.6% を占めた．また，アンケートの参加者からは，本調査を通じてセキュリティプラクティスを学習する機会となったことを示すポジティブな意見が寄せられた．これらの結果は，開発者の自発的な理解や知識に依存するだけでは不十分であり，外部からの働きかけによって認識を促す仕組みが求められることを示している．具体的には，対象プロジェクトにおける実践状況を自動的に検知し，開発者へ通知する仕組みが有効であると考えられる．この仕組みにより，開発者は熟練度に依存せず，通知を通じてプラクティスを知る機会を得られる．

開発環境への支援技術の統合．観測調査ではワークフロー数が多いリポジトリほどワークフローの構成に関するプラクティス（P2，P4）の実践率が低下する傾向が確認され，ユーザ調査では P4-1 の維持管理負荷に対する懸念が示された．特に P4-1 では各アクションにコミット ID（SHA）の指定が求められるため，更新頻度が高い場合には管理コストが増大する．この負担を軽減するには，コード反映後にプラットフォームが通知する仕組みだけでは不十分であり，開発環境内でのリアルタイム支援が必要となる．たとえば，Visual Studio Code の拡張機能として候補となるアクションの開発者情報や `owner/repo@ref` の `ref` を自動提示すれば，開発者はコーディングの過程で自然にプラクティスを実践できる．このように，開発段階で支援を組み込むシフトレフト型のアプローチが，実践率向上の鍵となる．

6. 研究倫理

本研究は著者所属機関において倫理審査免除とされたが，観測調査・ユーザ調査ともに倫理的配慮を徹底した．

観測調査では，サーバ負荷低減と非侵入的な調査を徹底した．具体的には，GitHub API の利用制限を超えない範囲でアクセスし，メタデータはパブリックデータセット（SEART-GHS）から取得した．また，対象リポジトリはローカル環境にクローンして分析し，GitHub 上のリポジトリに直接的な改変や書き込みを行わなかった．

ユーザ調査では，参加者募集，同意取得，プライバシー保護，インセンティブに配慮した．具体的には，関連性の高い参加者（コントリビュータ上位，明示的なメールアドレス公開，GitHub Actions 設定への関与，非ボット）を厳選することで，開発者を対象とする研究で近年問題視されている “調査疲れ” を軽減した．また，調査前の適切な同意取得を実施し，不要な個人情報を収集せず，回答は厳重に保管した．参加は完全に任意であり，金銭的報酬の代わりに希望者へ調査結果を共有する予定である．

最後に，本研究で得られた知見と提案は GitHub に共有するとともに改善に向けた協力を計画している．

7. まとめと今後の課題

本研究では，公開リポジトリの大規模な観測調査と開発者へのユーザ調査を組み合わせた混合研究法的分析により，GitHub Actions におけるセキュリティプラクティスの実態を明らかにした．観測調査の結果，セキュリティプラクティスの実装率は全体として低水準にとどまっており，さらにワークフロー数やリポジトリの所有者のアカウント種別といった特性と実践状況との間に統計的に有意な関連が確認された．また，ユーザ調査の結果，認知不足，理解不足や誤解，運用コストやリソースに対する懸念という 3 つの主要な阻害要因を特定した．これらの結果は，セキュリティプラクティスの普及に向けた対応の必要性を示しており，本研究ではドキュメントの改善やプラットフォームおよびツールによる支援の具体的な方向性を提案した．

今後の課題としては，セキュリティプラクティスの実践状況を継続的に追跡するとともに，他の CI/CD プラットフォームへの拡張や，本研究で提案した推奨事項の効果を評価することが挙げられる．特に，自動通知や開発環境への統合支援については，技術的な実装に加え，ユーザ調査を組み合わせて検証を行うことが重要である．

参考文献

[1] SolarWinds: SolarWinds Security Advisory, , available from ⟨https://www.solarwinds.com/sa-overview/securityadvisory⟩ (accessed 2025-08-02).

[2] Codecov: Bash Uploader Security Update, , available from ⟨https://about.codecov.io/security-update/⟩ (accessed 2025-08-02).

[3] GitHub: GitHub Actions, , available from ⟨https://github.co.jp/features/actions⟩ (accessed 2025-08-03).

[4] Golzadeh, M., Decan, A. and Mens, T.: On the rise and fall of CI services in GitHub, *In Proc. IEEE SANER 2022*.

[5] GitHub Advisory Database: CVE-2025-30066: tj-actions/changed-files Compromised by Malicious Code Injection, , available from ⟨https://github.com/advisories/GHSA-mrrh-fwg8-r2c3⟩ (accessed 2025-08-06).

[6] Koishybayev, I., Nahapetyan, A., Zachariah, R., Muralee, S., Reaves, B., Kapravelos, A. and Machiry, A.: Characterizing the Security of Github CI Workflows, *In Proc. USENIX Security 2022*.

[7] GitHub: Secure use reference, , available from ⟨https://docs.github.com/en/actions/reference/security/secure-use⟩ (accessed 2025-08-03).

[8] Dabic, O., Aghajani, E. and Bavota, G.: Sampling Projects in GitHub for MSR Studies, *In Proc. IEEE/ACM MSR 2021*.

[9] Gu, Y., Ying, L., Chai, H., Qiao, C., Duan, H. and Gao, X.: Continuous Intrusion: Characterizing the Security of Continuous Integration Services, *In Proc. IEEE S&P 2023*.

[10] Ayala, J., Tung, Y.-J. and Garcia, J.: A Mixed-Methods Study of Open-Source Software Maintainers On Vulnerability Management and Platform Security Features, *In Proc. USENIX Security 2025*.

[11] Ortloff, A.-M., Tiefenau, C. and Smith, M.: SoK: i have the (developer) power! sample size estimation for Fisher's exact, Chi-Squared, McNemar's, Wilcoxon rank-sum, Wilcoxon signed-rank and t-tests in developer-centered usable security, *In Proc. USENIX SOUPS 2023*.

[12] McDonald, N., Schoenebeck, S. and Forte, A.: Reliability and Inter-rater Reliability in Qualitative Research: Norms and Guidelines for CSCW and HCI Practice, *Proc. ACM Hum.-Comput. Interact.*