BLACKDUCK®

ガイド

# 可視性を向上して AI 駆動型開発のリスク管理を強化

「You can't fix what you can't see.（見えないものを直すことはできない）」。このシンプルな格言は人生のどこにでも当てはまり、DevSecOps プログラムもその 1 つです。このように、ソフトウェア・リスクを管理するには、開発者のデスクトップからビルド・パイプラインと CI パイプラインまで、ソフトウェア開発ライフサイクル（SDLC）のあらゆる段階にわたってエンドツーエンドの可視性を確保することが不可欠です。AI コーディング・アシスタント（GitHub Copilot、Tabnine など）が過去に例のないスピードで大量のコードを生成する現在では、特にこれが当てはまります。

すべてのチームが、問題の見通しを、できる限り速やかに一貫して確保することを主な目標とすべきですが、これには主なコントリビューター間での協調した取り組みが必要になります。

- **開発チーム：**セキュリティ上の問題点を早期に可視化することが非常に重要です。開発者がコードを書き、依存関係を宣言し、成果物をソース・コード管理（SCM）システムおよびレポジトリにチェックインする際に問題に気付く仕組みが必要です。これにより後工程での手戻りを回避し、次の機能ブランチに速やかに取り掛かることができます。
- **DevOps チーム：**SDLC 全体で一貫した可視性が確保されると、DevOps チームはパイプラインを通過する成果物のセキュリティ状態を理解できるようになります。これにはビルド時に解決される直接依存関係および推移的依存関係に加えて、機能テストおよびユニット・テストの結果が含まれている必要があります。DevOps チームは可視性を得ることで、タイムリーでセキュアなソフトウェア・リリースを実現するために、十分な情報に基づく調整を徐々に加え、ソフトウェアのリリース期限を常に守ることができます。
- **アプリケーション・セキュリティ（AppSec）チーム：**包括的な可視性によって、AppSec チームはすべての開発プロジェクトを認識して、開発ツールとビルド・パイプラインを理解し、注意喚起によって、ソフトウェア・サプライ・チェーンを通じて取り込まれる資産に気づくことができます。AppSec チームは可視性によりアタック・サーフェスを把握し、統合された自動化メカニズムを確立して、できる限り速やかに問題を検出して解決できます。

残念ながら、プロジェクト、ワークフロー、セキュリティの問題点に対する可視化不足は根強い問題であり、この問題はリスクを検知できる機会が減少する AI 活用開発では悪化する一方です。

可視性を確保する際の邪魔になり、DevSecOps チームが組織全体でリスク管理を改善するのを妨げる主な要因は次の 2 つです。

- **明らかに高速な開発アクティビティ：**各プロジェクトには独自開発のコード、オープンソース・ライブラリ、サードパーティの依存関係が含まれており、これらはすべて、開発者の好みの IDE およびビルド・パイプラインで、各種の言語とフレームワークを使用して作成されます。さらに、アプリケーションはさまざまな目的のために構築することができます。例えば、オンプレミスまたはクラウドでの実行、モノリシック、コンテナ化、またはサーバーレスのアーキテクチャなど、各開発者が一部の技術と機能に焦点を合わせていることも多々あります。これがサイロを生み出し、明瞭な可視性の妨げとなります。このような多様なツールおよびパイプライン構成を十分に明らかにしない限り、着実に問題を把握することは困難になる場合があります。
- **職務の分離と組織のサイロ化：**開発者だけがアプリケーション・セキュリティの責任を追うわけではありませんが、開発者が組織のセキュリティ・リスク態勢に与える影響は重大です。DevSecOps に対するビジョンが共有されていない場合、セキュリティ・ゲートが開発ワークフローを妨げる困難な障害になりかねません。これにより、AppSec チームの見えない場所に開発者が二次的なビルド・パイプラインとテスト環境を構築してしまい、リスクの評価と優先修正策の提案ができなくなる可能性があります。

DevSecOps には、サイロの解消と、さまざまなプロジェクト、パイプライン、リスクにまたがる一貫した可視性の確保が欠かせません。このためには、AppSec プロセスを開発者に合わせ、開発ツールおよびワークフローにセキュリティ・テストを直接統合して自動化する必要があります。また、AppSec チームは AI コーディング・アシスタントを人間の開発者のように扱う必要があります。人間の開発者は、間違いやすくタスク指向であるものの、適切な権限を与えればきわめて重要なセキュリティの後ろ盾となります。そうすることで初めて各コントリビューターがリスク態勢を正確に評価し、ワークフローを適宜調整できるようなり、長期的なセキュリティと効率の向上につながります。

AppSec チームは AI コーディング・アシスタントを人間の開発者 - 間違いやすくタスク指向であるものの、適切な権限を与えればきわめて重要なセキュリティの後ろ盾となる - のように扱う必要があります。

## あらゆる箇所を「監視」して可視性を向上

ソフトウェア・リスクに対する可視性を最も簡単に拡大する方法は、CI パイプラインのあらゆる段階に静的アプリケーション・セキュリティ・テスト（SAST）とソフトウェア・コンポジション解析（SCA）を組み込むことです。これにより、プロジェクトにリスクが入り込む余地があるすべての箇所に重要なリスク検出メカニズムを配置し、優先順位付けと修正を開発プロセスの近くに維持します。その結果、以下が実現します。

- セキュリティ、開発、DevOps のチーム間でのリスク認識の一致
- 継続的な改善の実現と問題のバックログの削減
- デプロイの遅延の防止
- 期限を遵守し、リスク許容度も妥協しない

さらに、これらのテストをパイプライン・トリガーに基づいて自動化することで、AI 活用開発の速度に合ったセキュリティを実現できます。セキュリティ・テストが DevOps ワークフローには避けられない当然の一部になると、開発者にリアルタイム・フィードバックが提供され、コード変更からセキュリティ評価までの時間が短くなります。検出された問題はトリアージされ、事前定義されたリスク許容ポリシーに基づいて修正の優先順位が付けられます。このポリシーはプロジェクト別にカスタマイズされ、セキュリティ、スピード、コンプライアンスのバランスを取るように改良されています。

重要なのは、統合テストは開発者がセキュアなコードを作成し、AI コーディング・アシスタントによる生成コードをクロスチェックするのに役立たなければならないということです。IDE プラグイン、SCM 拡張機能、柔軟な AppSec プラットフォームを使用すると、これを最も効果的に実現でき、開発者のワークフローが中断されることはありません。

統合テストは開発者がセキュアなコードを作成し、AI コーディング・アシスタントによる生成コードをクロスチェックするのに役立たなければなりません。

## IDE プラグイン：ソースでの可視性

開発者はしばしば、セキュリティの問題やソフトウェア・ライセンスの競合に対する防御の最前線と見なされます。開発者（または使用している AI コーディング・アシスタント）がプロジェクト・ファイルに持ち込んだ問題を自分で確認できれば、新たなリスクに注意を向けられるようになります。さらに、修正ガイダンスが IDE 内で提供されることで検出された問題を修正し、将来同じ問題を持ち込むことを回避するのが理想的です。

ブラック・ダックの **Code Sight™ IDE プラグイン**は開発者にとって「セキュリティのスペルチェッカー」となり、リアルタイムの可視性を強化して実用的な修正ガイダンスを提供します。Code Sight は効果的かつ自然な方法で開発者のコーディングを促進し、以下を含む機能を提供します。

- **SAST および SCA の統合：**Code Sight は、開発者がワークフローを変更したり別のツールにアクセスしたりしなくても、人間の開発者または AI コード生成機能によって作成されたコードに含まれる欠陥を検出し、脆弱なオープンソースの依存関係を識別します。
- **オンデマンド・スキャン：**開発者は開発プロセス中いつでも自分のコードをチェックし、作業を最適化するためにプロジェクト全体または単一ファイルをスキャンできます。
- **自動スキャン：**スキャンは指定した間隔で実行するか、コードのコミット、ビルド、プル・リクエストなどの所定のイベントによってトリガーできます。
- **スピードと詳細度のバランス：**開発者は開発の段階とコードの重要度に基づいて、各種のスキャン・モードから適切な詳細レベルを選択できます。
- **開発および AppSec 向けに統合された可視性：**Code Sight にその他のブラック・ダック・ソリューション（Black Duck Polaris™ Platform など）を接続することで、チームは IDE 外部のパイプライン・スキャンで検出されたリスクを確認し、ポリシーに基づいた修正の優先順位と、修正のために割り当てられた問題とを開発者に提供します。IDE 内でスキャンが開始されると、関連付けられたプロジェクトのテスト結果が自動的に Polaris に表示されるため、AppSec チームは常に新しい開発アクティビティを認識できます。

Code Sight を開発環境に取り込むことで、開発者が早い段階で効率的にセキュリティの課題を検出して対処できるようになるため、全体的な可視性が向上し、脆弱性が本番に持ち込まれるリスクが軽減されます。

## SCM 拡張機能：あらゆる変更点での可視性

ブラック・ダックは GitHub、GitLab、Azure DevOps、Jenkins、Bitbucket などの主要 SCM プラットフォーム向けに、すぐに使えるプラグインを提供しています。開発者はこれを使用することで、リスク許容ポリシーに準拠したまま、作業中のプロジェクトに合わせてテスト・ワークフローを最適化できます。このプラグインは、テンプレート化された SCM ワークフロー自動化スクリプトに含まれる事前設定済みのパラメータを調整するだけで使用できます。このスクリプトを使用すると以下のようなアクティビティを簡単に実行できます。

- SAST または SCA スキャンの起動
- 既存のリスク許容ポリシーとの比較
- スキャンの失敗またはポリシー違反後に実行するアクションの自動化（ビルドの中止など）
- 問題の詳細と修正ガイダンスに関する開発者へのフィードバックの仕組みの自動化（プル・リクエストのコメント、修正プル・リクエストのオープンなど）

SCM 拡張機能の利点で最も注目すべきなのは、（個人開発者のアクティブな作業中ファイルではなく）組織の共有コードベースに対する可視性が向上し、コードのレビュー / マージ / リリース段階でのコラボレーションが強化される点でしょう。

IDE プラグインと SCM 拡張機能の両方を使用することで、開発者はコーディング中に手元で速やかな支援を受けられ、コードベースに対するチーム・レベルの一元的な管理と可視性も実現されます。これにより、摩擦を軽減しながらセキュリティを強化できます。

これらの仕組みを組み合わせることで、セキュリティ承認済みのパイプラインが最も抵抗の少ない手段となるため、未承認で監視されていない二次的パイプラインを使用する動機を解消できます。

**可視性を高めるシナリオ**

**IDE の統合**

1. 開発者がコードを書くと、IDE の SAST スキャンにより潜在的な SQL インジェクションの脆弱性がハイライト表示される。
2. リアルタイムの警告と問題点を修正するための詳細なガイダンスが開発者に示される。
3. 開発者がオープンソースの依存関係を宣言し、ローカル・ビルドを実行する。
4. 宣言済み依存関係と推移的依存関係のリストが、脆弱性と、対応が必要な可能性のある関連ソフトウェア・ライセンスとともに開発者に示される。

**SCM 拡張機能**

1. 開発者が（収益を上げている）商用アプリケーション向けに何らかの要素を編集し、コードを GitHub にコミットする。
2. これにより、Black Duck Security Scan の GitHub Action を使用して設定されていたセキュリティ・スキャンがトリガーされる。
3. パイプライン・スキャンにより、クロスサイト・スクリプティング攻撃につながるコーディングの欠陥と、商用アプリケーション向けの企業ポリシーに違反するソフトウェア・ライセンスが検出され、ビルドが中止される。
4. ポリシー違反に関して注意喚起された開発者は、ソフトウェア・ライセンス違反の可能性についての自動生成されたプル・リクエスト・コメントを読み、ブラック・ダックの AI セキュリティ・アシスタントによって自動生成された、コードの欠陥に対する修正プル・リクエストを確認する。

## アプリケーション・セキュリティ・プラットフォーム：AI とともに拡張できる可視性

最近の AppSec ツールはセキュリティ上の問題点の検出とポリシーの適用を自動化することで、組織が高度なセキュリティとコンプライアンスを達成し、開発プロセスを効率化し、開発者への負担を軽減できるようにしています。また、これらのツールは AI 活用開発ワークフローのニーズの変化に合わせて拡張できます。

Polaris はクラウドベースの AppSec プラットフォームで、開発チームおよび DevSecOps チームのニーズに合わせて最適化されています。ブラック・ダックの市場をリードするセキュリティ解析エンジンを統合することで、以下の機能を提供します。

- **統合されたアプリケーション・セキュリティ・テスト・プラットフォーム。**Polaris は、SAST、SCA、動的アプリケーション・セキュリティ・テスト（DAST）の結果として信頼できる唯一の情報源を提供します。CI パイプラインと統合されると、Polaris はセキュリティ・チームが定義したポリシーに基づいて自動的に解析およびトリアージを実行し、何を優先して修正すべきかを開発者に知らせます。
- **一元的なポリシー。**Polaris により、すべてのセキュリティ・ポリシーを 1 か所で定義して管理できるため、初期開発からコード・コミットや最終デプロイまでのあらゆる CI/CD パイプライン段階にわたって、一貫してポリシーを適用することが容易になります。また、業界固有の規制や内部のセキュリティ・ガイドラインなど、組織に特有のニーズに合わせてポリシーを調整できます。
- **適用の自動化。**Polaris では、パイプラインのトリガーやスキャンの完了、ポリシー違反に基づいて自動化アクションを実行できます。これらのアクションの例には、重大なセキュリティの問題が検出された場合にパイプライン・アクティビティを自動的に阻止する、問題解決のための Jira チケットをオープンして担当開発者にアサインする、などがあります。こういった適用メカニズムにより、Polaris は SDLC 全体を通じて、一貫性がありながら柔軟なセキュリティ・ゲートを確立できます。
- **AI を活用した AppSec。**Polaris はセキュリティが最適化された独自の AI アシスタントを使用して問題の詳細を明確にし、開発者が目前の課題を完全に理解できるように支援し、SAST で見つかった問題の修正コードを生成します。

## チームに合わせた包括的な可視性の確立

これらの戦略、プラクティス、ツールを採用することで、問題発生後可能な限り速やかに検出と緩和のための優先順位付けが行えるようになります。さらに、次のステップの曖昧さをなくして、複数チームにまたがるフィードバック・ループを完結させることで、ワークフローでの AI コーディング・アシスタントの活用が進んだ場合も DevSecOps プログラムの効率と回復力を維持できます。

## ブラック・ダックのソリューションが役立つ理由

ブラック・ダックは、ベスト・オブ・ブリードのアプリケーション・セキュリティ・テスト・ソリューションから包括的なポートフォリオを提供しており、これらは SDLC の全段階にわたる緊密な統合と専用プラグインで強化されています。ブラック・ダックの市場をリードするテスト・エンジンにより、組織はテスト結果の取りまとめ、集約、標準化、優先順位付けを通じて自社のソフトウェア・リスク状況を全体的に把握できます。ますます多くの企業がソフトウェア・セキュリティ防御の最前線として開発チームを利用していますが、ブラック・ダックの Code Sight IDE プラグインと SCM セキュリティ自動化テンプレートおよび拡張機能（GitHub、GitLab、Azure DevOps などの主要プラットフォームに対応）は、このような企業をサポートします。

**準備はできていますか。ブラック・ダックのソリューションを使用し、セキュリティの可視化を通じてリスク管理を改善する方法について、ぜひお問い合わせください。**

# ブラック・ダックについて

ブラック・ダックは、True Scale Application Security によって、モダン・ソフトウェアの経営レベルのリスクに対応し、規制された、AI を活用した世界におけるソフトウェアの信頼性を保証します。ブラック・ダックのソリューションは、セキュリティ、規制、ライセンスに関するリスクを排除しつつ、組織のスピード、精度、コンプライアンスのトレードオフから解放します。 クラウドでもオンプレミスでも、コードが実行されるあらゆる場所でミッション・クリティカルなソフトウェアを保護するには、ブラック・ダックこそが選択肢となるのです。ブラック・ダックを活用することで、セキュリティ・リーダーはよりスマートな意思決定を行い、自信を持ってビジネス・イノベーションを推進することができます。詳しくは、www.blackduck.com/jp をご覧ください。

**ブラック・ダック・ソフトウェア合同会社**
www.blackduck.com/jp