ソースコードスニペット検出能力を考慮する必要がある理由とは？

現在利用可能なソフトウェア構成分析ツールは、コンプライアンスおよびセキュリティ脆弱性のみを対象とするものから、開発プロセス全体にわたってコンプライアンスプロセスの管理やチェック機能の一本化を行うものに至るまで、その機能は実にさまざまです。

自分自身のポリシーやプロセスに従ってワークフローを自由にカスタマイズできるツールがある一方で、一連の能力がキュレーションされたモノリシックなアプローチのツールもあります。どのツールを選ぶのが会社の技術・事業・法律面における目標に最も良く適しているかは、その会社の成熟度と専門性によって異なります。しかし、利用可能なツールを比較するにあたって、重要な注意点が一つあります。それは、コードスニペット検出能力があまりサポートされていない、あるいは完全に欠けている、という点です。

しかし、オープンソースのコンプライアンスという文脈で、なぜこれが重要なのでしょうか？なぜ、セキュリティ脆弱性の検出プロセスの一環として、スニペットサポートが欠かせないのでしょうか？

ソースコードが会社に入ってくる経路は？

ソースコードは、多様なソースを利用してコードを作成・統合する自社の開発者を通じて、またはサプライチェーンを介して会社に入ってきます。どちらの場合も、開発者は以下の組み合わせを行っています。

1. オリジナルコードの作成、
2. 多様なソースからのコードの再利用、および
3. コードの統合。

オープンソースのコンプライアンスとセキュリティの脆弱性検出にまつわる問題は、コンポーネントをそのまま使用する場合には発生しません。というのも、コンポーネント全体の検出・追跡は比較的容易に行えるからです(ここで、コンポーネント全体とはソフトウェアパッケージ全てを指すものとします。これに対し、スニペットはオープンソースソフトウェアパッケージからコピーした複数のコード行を指します)。真の問題が発生するのは、開発者がオープンソースのコードスニペットをコピーして、より大きなコード行本体の中に組み込んだ場合です。コードスニペットが組み込まれた途端に、市場に出回っている多くの既存ツールでは、このスニペットの本来の派生元とライセンス、および既知のセキュリティ脆弱性の識別と警告が困難になるのです。

コンポーネント全体

オープンソースプロジェクトは常につつき回され、再利用されるため、一部のスキャナは関係のない二次的一致のリストも含め、レポートをうるさいほど生成しがちです。**FossID を使用すれば、現在の形式(コード、アーカイブ、バイナリなど)に関係なく、コンポーネントの本来の派生元を自動的に識別する高速識別メソッドにより、多大な時間と分析作業を削減することができます。**

フルファイル
ファイルを自発的または自動的に変更すると、一致の識別がより困難になり、ライセンスのコンプライアンスアクションが必要になる場合があります。**FossID の検索アルゴリズムは、ファイルが編集されていてもファイルを見つけ出して、派生元とオリジナルライセンスに一致させます。**

コードスニペット
新機能の実装時やバグの修正時は、Web からコードをコピー&ペーストするのが一般的です。**FossID ならオープンソースのコードスニペットとそのライセンスを検出してくれるため、自社の企業ガイドラインに沿うこともオープンソースに適用されるライセンスに準拠することも容易に行え、自社製品およびサービスに真価をもたらすものに集中することができます。**

ソースコードスニペット検出能力を考慮する必要がある理由とは？
オープンソースコンプライアンスとセキュリティ脆弱性検出は、主にリスク管理に関連した問題です。製品やサービスに含まれるすべてのソースコードに適用されるライセンスにきちんと準拠し、セキュリティの脆弱性も回避したいものです。

会社としては自社の開発者がオープンソースソフトウェアを使用できるようにしたいと考え、開発者としてはコンポーネント全体の使用や、オープンソースプロジェクトに由来するファイルや部分的なコードスニペットの再利用を柔軟に行いたいと考えます。

スニペット検索と識別は、コンプライアンスおよびセキュリティという双方の観点におけるリスクを最小限に抑えながらオープンソースを利用するという柔軟性を得るための、中核をなす命題でありイネーブラーであると、私たちは信じています。

セキュリティ脆弱性の検出とスニペットとの関連は？
市場に出回っている現行のツールは、コンポーネント全体が使用されていることを発見した場合のセキュリティ脆弱性検出をサポートしています。例えば zlib (架空の例)を使用していて、ソースコードに既知の脆弱性がある場合、ツールはそのコードにフラグを付けて、脆弱性に関する詳細情報を提供します。ツールは、コンポーネント全体が使用されていると想定し(ソースコードが部分的にコピー&ペーストされている場合はカバーされません)、スキャンしたコンポーネントについて開示されている脆弱性と一致するファイルにフラグを付けます。ここで、zlib コードが何千というコンポーネントのすべてに(そのほとんどが異なるライセンスで)再利用されていると考えてみてください。一致候補として確認する必要のある偽陽性は、一体何件に上るでしょうか。

そのうえ、開発者がzlibコードベースからコードスニペットをコピーし、それを製品レポに組み込んでいたとしたら、どうなるでしょう？ツールは、そもそもこのようなスニペットを見つけることができません。そのため、コードとともにコピーされた既知の脆弱性にフラグを付けることもできないのです。つまり、コンプライアンスの問題と、発見されずに未解決となっているセキュリティの脆弱性という、2つの問題が隠れたまま残っていることとなります。

スニペットレベルでコードを追跡できる FossID なら、このような状況を回避できます。私たちは、外部ソースのコードスニペットを、既知のセキュリティ脆弱性を含むスキャン済みのソースコードにコピーして使用する方たちに、フラグ機能を提供します。また、コードの派生元、オリジナルコンポーネントの親のバージョン番号、ライセンス名およびバージョン、ならびに既知のセキュリティ脆弱性への参照のほか、該当する場合には詳細情報や可能な修正方法を示すポインタも提供します。

FossIDは、開発プロセスにシームレスに統合し、コードベース内からフリー&オープンソースソフトウェア(FOSS)の断片(コンポーネント全体にはじまりコードスニペットにいたるまで)を検出する最先端のオープンソーススキャナーを提供します。FossIDのソフトウェアは、ライセンス義務とコンプライアンスの問題を明らかにし、御社が優れた製品の開発に集中できるように貢献します。

www.fossid.com
@fossid_ab
linkedin.com/company/fossid-ab

GET IN TOUCH!

Discover all FossID
products and services at
www.fossid.com

FossID AB
Gåsgränd 3
111 27 Stockholm
Sweden

FossID K.K.
1-10-3-200 Roppongi, Minato-ku
Tokyo 106-0032
Japan