ソフトウェアサプライチェーン攻撃から身を守る 4 つのベストプラクティス

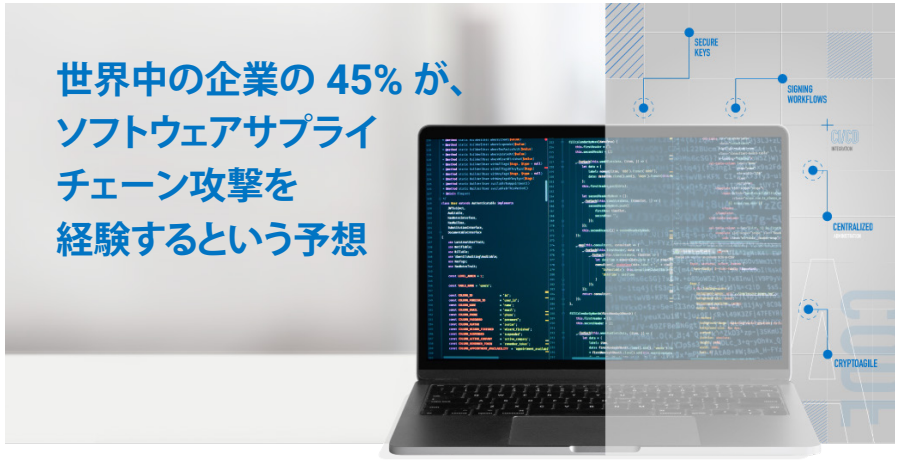
digicert

安全なコード署名、脅威とマルウェアの検出、SBOM 生成を統合

ソフトウェアサプライチェーン攻撃の頻度は増え続けています。ガートナーが発表した最近のレポートによると、2025 年までに世界中の企業の 45% がソフトウェアサプライチェーン攻撃を経験すると予測されています。2021 年比で、実に 3 倍です。

企業は、規模の大小を問わずこうした攻撃を経験しています。これまでにも、SolarWinds、CircleCI、3CX で発生したサプライチェーン攻撃については、ほとんどの方が耳にしたことがあるでしょう さらに最近も、Micro-Star International（MSI）が、同社の BIOS に使用されているコード署名の秘密鍵と Intel の BootGuard に使われている鍵が侵害されるという攻撃を受けました。

ゼロトラストの方針などの基本的なセキュリティ対策が、企業にとって第一線の防御策となるのは確かですが、企業のソフトウェアサプライチェーンとソフトウェア開発ライフサイクルを保護するには、それ以上の対策が必要です。



対象読者：

自社のソフトウェア開発インフラストラクチャを、侵害、攻撃、その他の脆弱性から保護する責務を負っている、またはそれに関与しているセキュリティ、DevSecOps、ソフトウェアエンジニアリングの各専門家。

課題

悪意のある攻撃者は、ソフトウェアサプライチェーンに対する攻撃に多種多様な手口を使います。侵害されたソフトウェア開発者のパスワードを狙ってソフトウェア開発システムに侵入する、保護されていないコード署名の秘密鍵を盗んだり悪用したりする、上流で普及率の高いオープンソースソフトウェアパッケージにマルウェアを挿入する、あるいはこれらすべての戦術を組み合わせて使う、といった行為です。2019 年、SolarWinds に対する Sunburst 攻撃で明らかになったように、攻撃戦術はきわめて巧妙に複雑になっているので、このような攻撃に対抗するさらに高度な手法が求められています。それを示したのが図 1 です。

図 1: サプライチェーン攻撃はどの段階でも起こりうる

攻撃は、ソフトウェア開発ライフサイクルのどの段階でも起こりえます。既存のスタンドアローンのセキュリティ対策ツールでは、一定の種類の攻撃は検出できても、それ以外の攻撃は見逃してしまうかもしれません。したがって、ソフトウェア開発ライフサイクルの全体を通じてセキュリティ対策を追加する「シフトアップ」のアプローチが必要です。

そのほか、組織的および文化的な課題もあります。ソフトウェア開発チームは、煩雑だとか、自動化されている CI/CD ビルドパイプラインを損ねるとか、ソフトウェアリリースプロセスが遅れるといった理由で、セキュリティ対策ツールに難色を示すことがあるのです。

セキュリティチームには、別の面でも課題があります。セキュリティに関するガイダンスを示し、コンプライアンスを徹底する必要があるにもかかわらず、ソフトウェアチームの多くは組織全体で散在し、たいていは地理的に分散しているということです。そのうえ、そうしたチームは、使用するプログラミング言語や環境が異なり、ソフトウェアを展開するプラットフォームも違えば、ソフトウェア構築のインフラストラクチャやソフトウェアプロセスも違うことが少なくありません。このような異種混在のために、企業全体を通じてソフトウェアセキュリティポリシーを定義、実施、監視するのはかなり困難になることがあります。

最後に、組織にとって新たな課題となっているのが、ソフトウェアサプライチェーン攻撃への対処を意図した各種の政府規制や業界規制、たとえばソフトウェアの透明性を求める規制などへの対応です。

ソフトウェアサプライチェーン保護のベストプラクティス

なかには、ベストプラクティスだけでなく、ソフトウェアサプライチェーンの完全性を保証するために企業が従うべき要件も定めている規制や業界の推奨事項があります。たとえば、米国の「国家のサイバーセキュリティの向上に関する大統領令」や EU の「サイバーレジリエンス法案」では、ソフトウェアの透明性を示す SBOM の作成が求められています。NIST の「Security Considerations for Code Signing（コード署名に関するセキュリティ上の考慮事項）」では、安全なコード署名のために従う手順の概略が示されています。

本ガイドでは、このような課題に対処するうえで企業が活用できる 4 つのベストプラクティスと、デジサートがお届けする支援の形について概説します。ここで取り上げるベストプラクティスで重要なのは、SDLC 上の単一のステージに焦点を当てるのではなく、ソフトウェア開発ライフサイクル全体を通じて実施するということです（図 2 を参照）。

図 2: ソフトウェアの信頼 : 左へ、右へ ... そして上へ

SDLC の複数の側面に焦点を当てたソフトウェアサプライチェーン攻撃に対抗するには、多面的でありながら統合された次世代のアプローチが必要です。

#1: 企業全体を通じた安全なコード署名

コード署名は、30 年以上にわたって使われている基本的なセキュリティ対策です。暗号を使ってソフトウェアパッケージ（実行可能ファイル、ライブラリ、アプリなど）に電子署名することで、ソフトウェアが本物であり（たとえば、デジサートによって署名されたソフトウェアは、間違いなくデジサートから提供されている）、第三者によって改ざんされていないことを証明します。iOS、macOS、Windows など多くのオペレーティングシステムは、パッケージをインストールして実行する前にソフトウェアの電子署名を確認します。

ソフトウェア署名はきわめて効果的なので、悪意のある攻撃者はコード署名システムを狙って、企業のコード署名証明書や秘密鍵を盗んだり、侵害あるいは悪用したりしています。

コード署名を保護するうえで明らかな第一歩は、コード署名の秘密鍵をすべて安全に保管することです。認証局 / ブラウザ（CA/B）フォーラムによる最近の要件では、公的に信頼されているコード署名の秘密鍵を、ハードウェアセキュリティモジュール（HSM）のような FIPS 140 レベル 2、コモンクライテリア EAL 4+ デバイスに保管するよう要求されています。

安全なストレージを使うのは必須の第一歩ですが、企業全体でコード署名を安全に行うと言う目的には十分とは言えません。ユーザー名 / パスワードが侵害されていると、悪意のある攻撃者はその安全なストレージにアクセスできる可能性があるからです。

そのかわりに推奨したいのが、企業レベルの強力なコード署名システムを使用することです。以下のようなセキュリティ対策を追加して実施します。

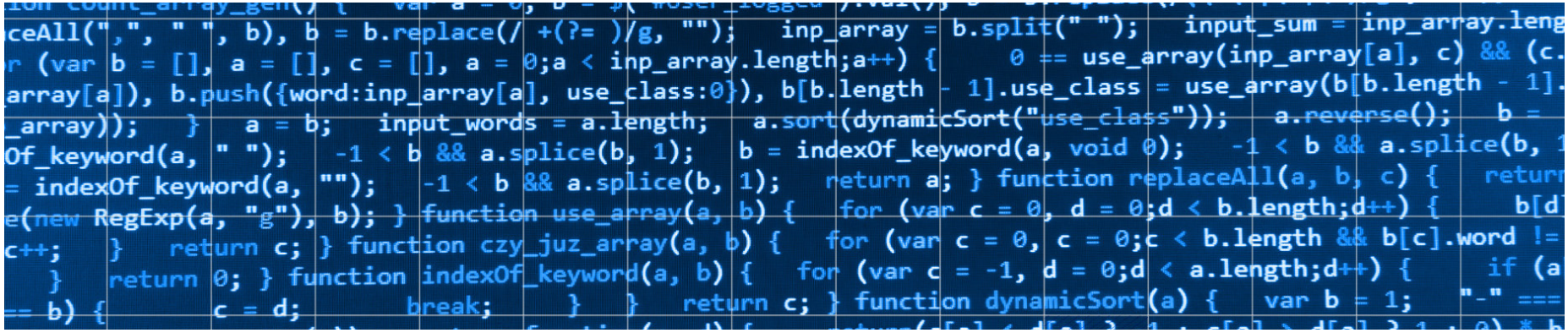
- 発行と失効を含めて、コード署名用証明書管理を自動化する。
- 誰がコード署名鍵にアクセスできるかだけではなく、誰がそのアクセスを承認しなければならないか、どんな状況で鍵にアクセスできるか（時間帯、ビルドマシンなど）まで指定する承認プロセス。
- 署名操作のために特定のコード署名用証明書と鍵にアクセスできるのは誰か、署名操作を承認する権限を付与されるのは何か、監視を許可されるのは何かを指定するロールベースのシステム。
- コード署名活動すべてについて、万一にも証明書が危殆化した場合に参照できる、反論の余地のない記録の保持。
- 大量のコード署名操作、多種多様なソフトウェア展開および開発プラットフォームやツールをサポートする単一のコード署名システム。
- ソフトウェアのリリース前に満たさなければならない管理措置に対するポリシーベースのアプローチ。
- コミット前のソースコード、ビルドスクリプトやレシピ、コンテナその他のソフトウェアインフラストラクチャファイルなど、SDLC 全体にわたる中間的なソフトウェア成果物への署名。
- さまざまなソフトウェアチームの所在にかかわらず、一元化的な場所から実施できる全社的な可視化と、ポリシー定義の実施。

#2: 包括的な脅威検出

ソフトウェアリリース後の改ざんはコード署名によって防止できますが、署名とリリースの前に改ざんされた場合のソフトウェアはどうなるでしょうか。たとえば、悪意のある攻撃者がビルドシステムに侵入すると、企業のソースコードリポジトリにマルウェアを挿入できるかもしれません。あるいは、ソフトウェア開発者がオープンソースのパッケージをダウンロードして、そこにマルウェアが隠されている可能性もあります。隠れていてこれまでは発見されなかった脆弱性が発見されることもありえます。

ソフトウェア静的解析ツール（SAST）など、市販されている各種のセキュリティツールも、こうした問題の一部には対応しています。しかし、こうしたツールが焦点を当てるのは、たいてい単一の側面（オープンソースソフトウェアの脆弱性など）にすぎません。

そうではなく、最終的なスキャンソフトウェアバイナリの中から脅威、脆弱性、埋め込まれた秘密情報やマルウェアの存在を見つけ出す総合的な対策が必要です。このスキャンは、オープンソースコード、サードパーティの商用ライブラリ、あるいは企業が書いたプロプライエタリコードなど、ソフトウェアのコンポーネントの元々のソースとは関係なく実施されるべきです。



#3: ソフトウェアの透明性

前述したように、政府および業界の規制では、ソフトウェアの内容を明らかにすることをソフトウェア開発者に求める傾向が強くなっています。食品の原材料を示す栄養表示と同じです。SBOM と呼ばれるこの文書は、コンポーネントのソース（オープンソース、サードパーティライブラリなど）にかかわらず、特定のソフトウェアに含まれるコンポーネントすべてをカタログ化したものです。食品の栄養表示と同じく、SBOM はソフトウェアの単なる「原材料」リストを示すだけではなく、使用されているバージョンや既知の脆弱性など、「原材料」の品質評価にも利用できます。

コンプライアンスに必要ではないとしても、以下の理由から組織は SBOM を作成したほうがいいでしょう。

- **ソフトウェアの攻撃対象領域に関する評価と優先順位付け**：ライブラリやコンポーネントの脆弱なバージョンや狙われるバージョンを探し、ソフトウェアの依存性を調べて、必須のセキュリティパッチが適用されているかどうかをチェックします。
- 脆弱性対策が欠落していたり、安全でないコード署名を実施していたりなど、**古くなったコンポーネントを探します**。
- 検出チームや脅威インシデント対応チームをサポートし、脅威モデリングを実施するソフトウェア構造を利用して、**先取的な防御策を講じます**。

#4: シフトアップ：企業と SDLC の全体にわたる管理と可視性

セキュリティ対策の多くは、SDLC の特定の部分を対象としています。DevOps プロセスでは、セキュリティのシフトレフトとシフトライトという言葉をお聞きになったことがあるでしょう。これは、森の中で一本一本の木を見るようなものです。しかし、攻撃が複雑になるのに伴って、組織は一歩下がって「シフトアップ」しなければなりません。

- SDLC プロセス全体を視野に入れ、ソフトウェアを全体的に見渡す。
- 企業全体でセキュリティポリシー、対策、結果を可視化する。
- 企業全体でセキュリティポリシーを定義し、実施する。
- 最終的なソフトウェアの実行可能ファイルが署名されてリリースされる直前に、そのファイルについて脅威と脆弱性の分析を実行する。

これを効果的に実行し、ソフトウェア開発チームに受け入れてもらうには、セキュリティツールが既存のソフトウェアプロセスにシームレスに統合されること、そしてソフトウェアリリースプロセスを遅延させないことが必要です。ソフトウェア開発者にとって使いやすく、さらには透明性を備えていなければなりません。

DigiCert Software Trust Manager

DigiCert® Software Trust Manager は、ソフトウェアサプライチェーンの全体でソフトウェアの完全性を保護するデジタルトラストソリューションです。コードが危殆化するリスクを低減し、企業や規制上のポリシーを適用して、コード署名時の鍵の使用とアクセス制御をきめ細かく実現します。

開発、ビルド、リリースに伴うセキュリティ侵害やマルウェア拡散のリスクを最小限に抑えられ、企業はソフトウェアサプライチェーン攻撃から身を守りながら、政府の規制を遵守できます。

DigiCert Software Trust Manager をご利用になれば、以上 4 つのベストプラクティスの導入も簡単に実現可能です。

詳細については、https://www.digicert.com/jp/software-trust-manager をご覧ください。

図 3: Software Trust Manager

DigiCert Software Trust Manager は、安全なコード署名、包括的な脅威検出、ソフトウェアの透明性、企業の可視性と管理を実現することによって、ソフトウェア開発ライフサイクルの全体を保護します。