ソフトウェア サプライ チェーンの現状レポート 2025

拡大する脅威がソフトウェアの整合性を脅かす

目次

はじめに

ソフトウェア サプライ チェーン全体の管理とセキュリティの確保は、信頼できるソフトウェア リリースを提供するための基本的要件です。しかし、「言うは易く行うは難し」です。ソフトウェア セキュリティに特化した企業として、専用のセキュリティ研究部門を持ち、これまで 15 年以上にわたり開発チームとセキュリティ チームを支援してきた JFrog は、今日の組織が直面する脅威と課題を理解しています。AI 時代の到来とともに、これらの課題はさらに加速しており、多くの DevSecOps チームは、この変化にどのように対応すればよいのか、日々考えています。

このレポートは、何百万人ものユーザーから得た JFrog 使用状況データ、JFrog セキュリティ リサーチ チームによる CVE (Common Vulnerabilities and Exposures、共通脆弱性識別子) 分析、1,400 人のセキュリティ、開発、運用担当者を対象とした第三者機関による調査データを組み合わせることにより、これらの課題に対する答えを提供します。この分析から、2025 年時点でのソフトウェア サプライ チェーンと開発環境の状況を把握し、永続的なリスクと新たなリスクを明らかにして、ソフトウェア サプライ チェーンを保護するために必要な情報を得ることができます。

このレポートが皆様のお役に立つことを期待しています。フィードバックがありましたら、data_report@jfrog.com までお送りください。

エグゼクティブ ブリーフ

ソフトウェア サプライ チェーンは前例のない速さで進化しており、組織は対処することが困難なペースで新たな脅威にさらされる可能性があります。「より多く」が、必ずしもサプライ チェーン全体のリスクを軽減するための最善のアプローチとは限りません。

「がむしゃらに働くのではなく、スマートに働く」という古い格言に基づいて、ツール チェーンとプロセスを簡素化することが、迅速な行動、新しいテクノロジの導入、競争優位性を求める組織にとって、最善のアプローチと言えます。

ソフトウェア サプライ チェーン – より大きく、より速く、より複雑に

オープンソース エコシステムの成長が減速する兆候はなく、革新に意欲的な組織は最新テクノロジを活用するべく迅速に動いています。

- 組織の 64% は 7 つ以上の、44% は 10 以上のプログラミング言語を使用しています。これは前年比でそれぞれ 53% および 31% の増加です。
- パブリック リポジトリは引き続き成長しています。注目すべき点は、2024 年に Docker Hub が 190 万のイメージを、Hugging Face が 100 万のイメージをそれぞれ追加したことです。
- 一般的な組織は年間 458 の新規パッケージを導入しています。平均すると、月に 38 の新規パッケージを導入していることになります (開発者の数により異なります)。

リスクの増加と透明性の減少

CVE の潜在的な影響を理解することは依然として複雑な作業ですが、これはリスクという氷山の一角に過ぎません。

- NVD (National Vulnerability Database) には膨大なバックログがありますが、新たな脆弱性の公開は止まりませんでした。2024 年には 33,000 を超える新たな CVE がレポートされ、前年比で 27% 増加しました。
- JFrog セキュリティ リサーチ チームは、パブリック レジストリで 25,229 (前年比で 64% 増加) の漏洩したシークレット/トークンを検出し、そのうち 6,790 はアクティブでした。
- JFrog セキュリティ リサーチ チームが 183 の重要な CVE を詳細に分析した結果、63 の CVE は JFrog Cloud のお客様のスキャン対象アプリケーションで悪用できないことが判明しました。

セキュリティ リスクに取り組むときに、基本を軽視しない

組織はさまざまなレベルのセキュリティ フレームワークを採用し、多くのセキュリティ ツールを使用していますが、その過程で、いくつかの基本的なベスト プラクティスを見落としています。

- 71% は、組織が開発者にインターネットからパッケージを直接ダウンロードすることを許可していると回答しています。
- 組織の 73% は 7 以上の、49% は 10 以上のセキュリティ ソリューションを使用しています。それぞれ、昨年の 47% と 33% から増加しています。
- 組織がコード レベルとバイナリ レベルでスキャンを行っていると回答したのは半数未満 (43%) です。
- 回答者の 40% は、本番環境で実行しているソフトウェアの出所を完全に可視化できていません。

AI の導入は新たな段階へ

AI サービスを本番環境に導入するための選択肢はこれまでになく増えていますが、その結果、組織は新たな懸念事項に対処する必要が生じています。

- 今年、Hugging Face には 100 万を超える新しいモデルとデータセットが追加されましたが、悪意のあるモデルも 6.5 倍に増加しました。
- チームの 64% はホスト型モデルに移行しつつありますが、組織の約半数は、独自のモデルとオープンソースのモデルの両方を何らかの形でセルフホスティングしています。
- 現在、組織の 37% は、承認されたモデルのリストのキュレートと保守を手作業で行って、モデル アーティファクトの使用を管理しています。

ソフトウェア サプライ チェーンには何が含まれているか?

現代のソフトウェア サプライ チェーンは、グローバルで拡張性が高く、複数のテクノロジとソースを統合しています。最も人気の高いテクノロジ エコシステムには、毎年何百万もの新しいパッケージとライブラリが追加されています。ソフトウェア開発組織は、かつてないほどの多くの言語と、それらに対応するパッケージ エコシステムを活用しています。従来のテクノロジが依然として広く利用されている一方で、定評のある新しいオープンソース エコシステムの採用には、可能性とリスクがあります。このレポートでは、これらの可能性とリスクについて詳しく調査します。

開発組織で使用しているプログラミング言語の数

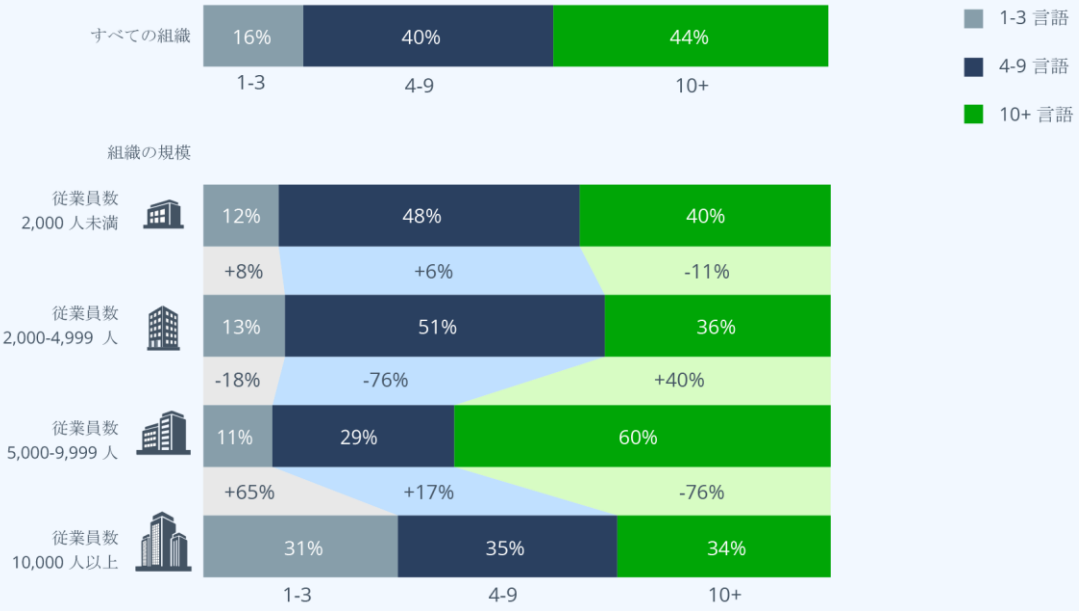


図 1.1. ソフトウェア開発組織で使用しているプログラミング言語の数は? (委託調査、2024 年)

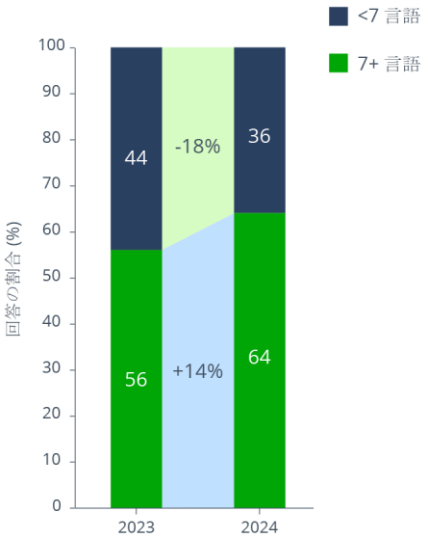


図 1.2

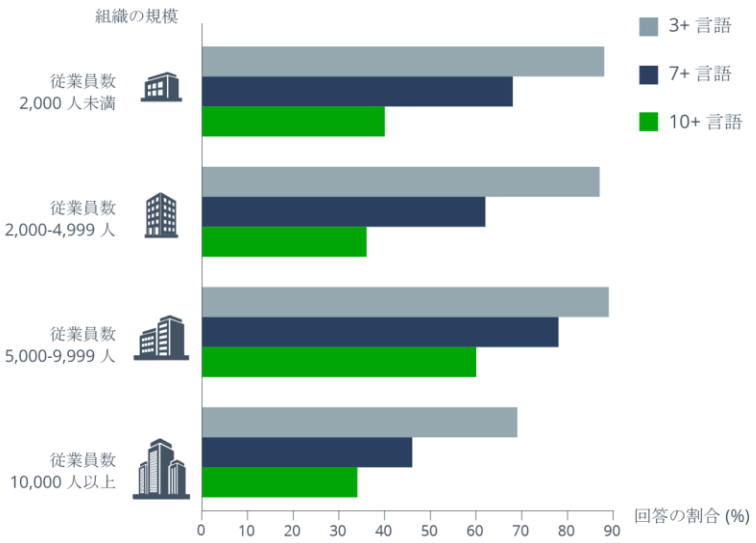


図 1.3

テクノロジ専門家の約 3 分の 2 (64%) が、所属組織で 7 つ以上のプログラミング言語を使用していると回答しています。昨年は、回答者の半数強 (56%) でした。この増加は、ソフトウェア サプライ チェーン全体で複雑性が全面的に増加していることを反映しています。組織の規模が大きくなるにつれて、使用している言語の数も増えています。これは予想された傾向です。しかし、組織の規模が従業員 1 万人以上になると、使用している言語の数は減少しています。この減少は、組織が転換点を迎え、開発管理でより積極的なアプローチを取り、特定のテクノロジを使用して標準化することによりスプロールを抑制する必要があることに気付いたことを表していると考えられます。大規模な組織では、実証されたレガシー アプリケーションを継続して使用することが多く、追加のテクノロジ エコシステムの使用が必要な新規プロジェクトが少ないことも考えられます。

パッケージ タイプ別の年間の新規パッケージ数

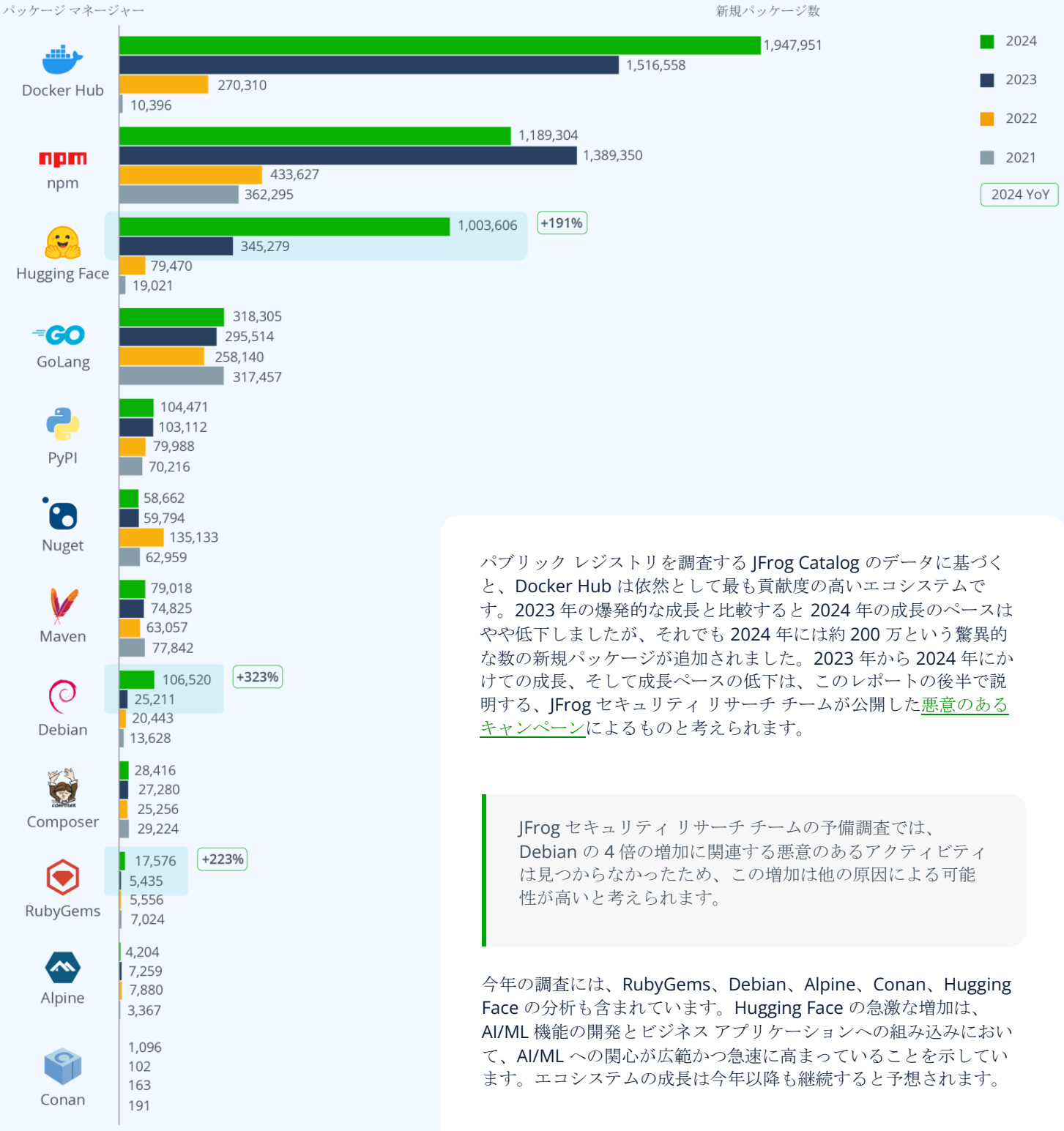


図 2. パッケージ タイプ別の年間の新規パッケージ数
(JFrog Catalog データベース、2024 年)

パブリック レジストリを調査する JFrog Catalog のデータに基づくと、Docker Hub は依然として最も貢献度の高いエコシステムです。2023 年の爆発的な成長と比較すると 2024 年の成長のペースはやや低下しましたが、それでも 2024 年には約 200 万という驚異的な数の新規パッケージが追加されました。2023 年から 2024 年にかけての成長、そして成長ペースの低下は、このレポートの後半で説明する、JFrog セキュリティ リサーチ チームが公開した悪意のあるキャンペーンによるものと考えられます。

> JFrog セキュリティ リサーチ チームの予備調査では、Debian の 4 倍の増加に関連する悪意のあるアクティビティは見つからなかったため、この増加は他の原因による可能性が高いと考えられます。

今年の調査には、RubyGems、Debian、Alpine、Conan、Hugging Face の分析も含まれています。Hugging Face の急激な増加は、AI/ML 機能の開発とビジネス アプリケーションへの組み込みにおいて、AI/ML への関心が広範かつ急速に高まっていることを示しています。エコシステムの成長は今年以降も継続すると予想されます。

組織で使用している主要パッケージ テクノロジ

パッケージ タイプ	リクエスト*	リポジトリ数	アーティファクト
Maven	33.52%	104,955	2,567,881,564
npm	30.45%	48,549	674,010,130
Docker	15.45%	112,366	2,264,459,098
YUM	2.68%	14,669	20,785,724
PyPI	2.68%	22,352	66,838,230
Helm	1.61%	26,125	13,231,209
Nuget	1.45%	28,497	131,164,087
Debian	1.35%	8,184	8,066,185
Conan	1.33%	3,420	143,404,846
Gradle	0.99%	9,073	102,198,342
RubyGems	0.93%	3,736	46,728,889
Go	0.75%	9,034	16,511,299
OCI	0.47%	862	8,662,480
Cargo	0.13%	1,261	526,851
Sbt	0.12%	2,239	14,908,497
Helm OCI	0.07%	1,633	201,440
Ivy	0.06%	2,283	31,786,069
Composer	0.05%	2,413	614,957
Terraform	0.03%	3,566	675,684
Opkg	0.02%	529	33,812,836
Conda	0.02%	2,168	1,538,832
P2	0.02%	316	1,010,616
Pub	0.01%	363	166,878
Swift	0.01%	524	1,345,299
Alpine	0.01%	1,550	111,231
Cocoapods	<0.01%	1,400	2,973,045
Cran	<0.01%	2,403	816,170
VCS	<0.01%	273	1,692
Chef	<0.01%	1,530	150,462
Vagrant	<0.01%	680	7,326
Terraform Backend	<0.01%	2,307	395,004
Bower	<0.01%	985	44,161
Ansible	<0.01%	107	4,470
Puppet	<0.01%	1,530	17,758
Hugging Face	<0.01%	551	12,638

*第 4 四半期の 570 億のリクエストのうち各タイプのリクエストの割合

2024 年の年末 (第 4 四半期) にスナップショットを作成し、JFrog がサポートしている 35 以上のテクノロジ タイプの中で最も人気の高いテクノロジをより正確に把握しました。npm、Docker、Maven などの確立されたテクノロジ エコシステムが引き続き普及している一方、YUM と Cargo の人気が大幅に上昇しました。

ここ数年、特に政府機関がよりメモリ セーフな開発を推進したことにより、Cargo の人気は着実に高くなってきました。Rust の人気が頭打ちになるのか、Java のようなより確立された言語のレベルまで広範に利用および採用されるようになるのかはまだ分かりません。

OCI と Helm OCI の使用量も注目に値します。JFrog は 2024 年初めに OCI 専用リポジトリを導入し、多くのお客様が既に活用されています。これは、コンテナーやその他のテクノロジ エコシステムでオープン標準への関心が高まっていることを示すものであり、Terraform リポジトリを拡張して OpenTofu をネイティブにサポートした理由です。

一般的なテクノロジの使用状況は業界により異なります。

- **自動車業界や IoT 企業**は、Maven (バックエンド アプリ)、npm (フロントエンド アプリ)、Conan (組み込みデバイス)、Docker、PyPI (AI/ML 向け) を活用して、これらの多くを汎用パッケージ (tar/zip イメージ) にバンドルすることがあります。
- **AI/ML 企業やロボティクス企業**は、Hugging Face や Tensorflow などのパブリック リポジトリから取得した PyPI や ML モデルを活用して、コンテナーや汎用パッケージ (tar/zip) に保存しています。Hugging Face や JFrog の機械学習リポジトリなどのネイティブ リポジトリをモデルに採用することもあります。
- **保険、金融、小売業界**は、Maven、npm、Docker などのテクノロジを組み合わせて活用しています。現在では、AI/ML の普及に伴い、競争力を維持するため PyPI と ML モデルも活用するようになっています。

*JFrog の機械学習リポジトリは 2025 年 1 月に導入されたため、このレポートのデータには含まれていません。

図 3. 組織で使用しているテクノロジ、アクション数、リポジトリ数、各テクノロジに保存されているアーティファクトの合計サイズ (JFrog データベース、2024 年)

人気の高いライブラリ

ランク	Docker	Maven	PyPI	npm
1	library/alpine	org.slf4j:slf4j-api	urllib3	@types/node
2	library/node	commons-io:cozmons-io	requests	semver
3	library/python	commons-codec:commons-codec	certifi	minimatch
4	library/nginx	org.ow2.asm:asm	charset-normalizer	glob
5	library/redis	com.fasterxml.jackson.core:jackson-core	setuptools	electron-to-chromium
6	library/busybox	com.google.guava:guava	idna	lru-cache
7	library/postgres	com.fasterxml.jackson.core:jackson-databind	packaging	caniuse-lite
8	library/ubuntu	com.fasterxml.jackson. core:jackson-annotations	typing-extensions	acorn
9	library/openjdk	org.apache.commons:commons-compress	wheel	debug
10	library/debian	org.apache.commons:commons-lang3	PyYAML	@babel/parser
11	grafana/grafana	org.codehaus.plexus:plexus-utils	python-dateutil	strip-ansi
12	library/golang	junit:junit	numpy	browserslist
13	library/hello-world	org.apache.httpcomponents:httpcore	click	@babel/types
14	library/maven	org.apache.httpcomponents:httpclient	MarkupSafe	tslib
15	library/docker	com.google.code.findbugs:jsr305	pytz	resolve
16	library/eclipse-temurin	com.google.errorprone:error_prone_annotations	cryptography	commander
17	curlimages/curl	commons-logging:commons-logging	cffi	qs
18	library/mongo	net.bytebuddy:byte-buddy	importlib-metadata	@babel/code-frame
19	library/centos	org.objenesis:objenesis	zipp	@babel/generator
20	library/amazoncorretto	org.apache.maven:maven-artifact	attrs	chalk

図 4. JFrog Cloud (SaaS) での Docker、Maven、PyPI、npm のダウンロード数上位 20 のパッケージ (JFrog データベース、2024 年)

多くのパブリック レジストリは、含まれるパッケージのダウンロード メトリクスを提供していますが、これらのメトリクスは、ビルドを実行するたびにクライアントがパッケージを取得するなどの要因の影響を受けることを含め、さまざまな理由から誤解を招く可能性があります。JFrog の調査では、何千ものお客様のアカウントで使用する JFrog SaaS 環境へのリクエストに基づいて、実際に使用しているライブラリを推測しています。

Docker の上位 20 のイメージに最も人気の高いオペレーティング システムと主要な開発言語が含まれているのは当然のことです。これらはおそらく親イメージとして使用されていると思われます。1 つを除いて、Docker 公式のイメージ、または検証済み発行元により提供されているイメージであることは心強いことです。これは、これらのイメージが定期的にメンテナンスされることを示しています。特に、公式の Docker hello-world イメージがこの上位グループに含まれていることは、このコンテナー化が現代のソフトウェア配信で広く普及したことにより、デモ、概念実証、および開発者が Docker の使い方を学習する健全なコレクションとなっていることを示していると考えられます。

Maven、PyPI、npm パッケージの上位 20 には意外なものはありませんでした。ただし、これらのパッケージが直接取り込まれたのか、ソフトウェア開発者により明示的に選択されたのか、依存関係として取り込まれたのか、推移的な依存関係 (つまり、依存関係の依存関係) として取り込まれたのかは不明です。

たとえば、Apache Commons Compress は Maven の 9 位にランクされています。このライブラリを詳しく見ると、Apache Commons IO、Apache Commons Codec、ASM、Apache Commons Lang (2、3、4、10 位) に直接依存していることが分かります。これは、個々の構成要素を評価して、特定のコンポーネントが脆弱になったり、侵害されたり、単にソフトウェア サプライ チェーンで紛失したりした場合の影響の範囲を適切に理解するには、アプリケーションに含まれるソフトウェア アーティファクトの最新のインベントリ (多くの場合、SBOM 形式) の維持が重要であることを浮き彫りにしています。

組織が新規 OSS パッケージを導入するペース

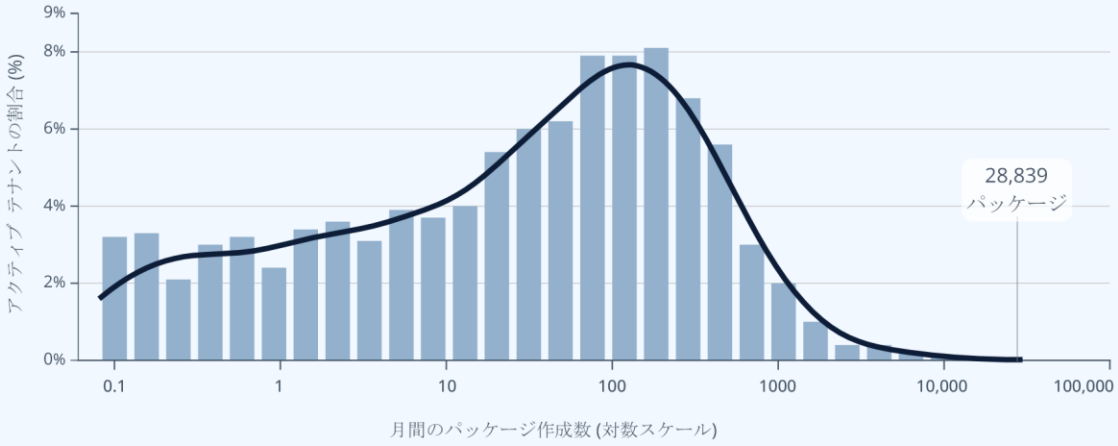


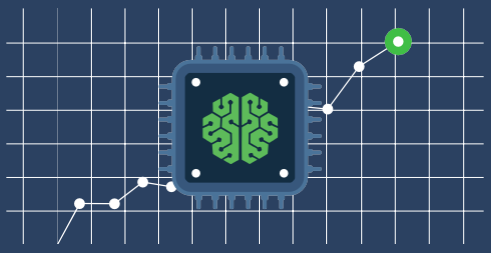
図 5. 2024 年にアクティブ テナント向けに毎月作成された新規パッケージの分布 (JFrog データベース、2024 年)

2024 年に、JFrog のクラウドネイティブ SaaS サービス、JFrog Cloud を使用する組織は、合計 700 万を超える新規パッケージをソフトウェア サプライ チェーンに導入しました。

平均的な組織は年間で約 2,000 のパッケージを導入しますが、この数は少数のヘビー ユーザーにより増加しています。最大の組織が年間で 346,000 の新規パッケージを導入した一方で、中央値の組織ははるかに管理しやすい 231 のパッケージを導入しました。

パッケージを導入しなかった組織を除外すると、新規パッケージ数の中央値は 458 (月あたり 38) まで増えます。データに基づくと、この数は典型的な組織を最も適切に表していると考えられます。新規パッケージが 1 日に 1 つを超える程度のペースでも、組織の環境に導入されるパッケージのセキュリティ、運用リスク、ライセンス コンプライアンスをどのように考慮して管理するかという点で、組織は大きな課題に直面する可能性があります。

重要なポイント



AI の爆発的な増加

入手可能な AI/ML コンポーネントは飛躍的に増加しており、コミュニティや企業がエコシステムに貢献することにより関与するケースが増えています (例: Nvidia は NIM と NVLM をリリース)。組織が AI サービスの自社製品への追加を推進していることは、現時点で JFrog ユーザーが 500 以上の Hugging Face リポジトリを作成していることからも明らかです。オープンソースのモデルとデータセットの利用方法およびセキュリティの確保について、明確なポリシーと戦略を定めることが重要です。この点については、後ほど詳しく説明します。

アプリケーションとその開発の保護は最優先事項

米国政府およびその他の国際的な政治機関は、より安全な開発言語とフレームワークの使用を推進してきました。JFrog のデータでも Rust/Cargo の使用が増加し始めており、組織がアプリケーションを再設計したり、セキュリティ重視の立場から新しいプロジェクトを開始している可能性が示されています。さらに、OCI の人気の高まりは、好みのオープンソース テクノロジが非公開化、ビジネス ソース化されたり、ライセンスが必要になることについて懸念を募らせていることが一因と考えられます。

リスクの急増

組織の 3 分の 2 が 7 つ以上の言語、ほぼ半数が 10 以上の言語を使用しています。それらの組織では、複数の異なる言語、複数の異なるチーム、複数の異なる脅威の原因に対して一貫したパイプラインを確保する必要があるため、組織のリスクは飛躍的に増加しています。本番環境に配信するアプリケーションのセキュリティを確保するには、開発段階で、各エコシステムに存在する特有の脆弱性、悪意のあるアクター、独自の構造を考慮する必要があります。

迅速な対応で攻撃者を寄せ付けない

動きの速い組織は、1 日に 1 つ以上の新しいパッケージやバージョンを導入しており、ソフトウェア サプライチェーンに導入されるこれらのコンポーネントのセキュリティを確保するための、自動化され改善されたプロセスが不可欠です。組織が速度の向上を継続的に目指し、開発者とセキュリティ チームがビジネスの課題に対する斬新なソリューションを見つけられるように支援するにつれて、組織に導入されるパッケージのペースは今後も上昇し続けると考えられます。

ソフトウェア サプライチェーンにおけるリスクの増加

組織は悪意のあるアクターとの競争を続けており、衰える気配のない、数々の主要な要因に対処する必要があります。

- CVE
- 悪意のあるパッケージ
- オープンソース ライセンスのリスク
- 運用のリスク (パッケージ管理の不備、EoL など)
- シークレットの漏洩
- 設定ミス/人的ミス

分析によれば、開発者やセキュリティ専門家が現在使用しているツールは、場合によっては役立ちますが、場合によっては害を及ぼします。たとえば、AI コード アシスタントを適切に使用しなかった場合、特に機能の設定が不十分または不適切な場合、潜在的に悪影響を及ぼす可能性があります。

今年は NVD (National Vulnerability Database、米国国立標準技術研究所 (NIST) が運営する脆弱性データベース) が新たに公開した CVE (Common Vulnerabilities and Exposures、共通脆弱性識別子) の分析とプロパティの割り当てを数ヶ月にわたり行えなかったことと、その結果生じたバックログにより、このセクションで示した CVSS (Common Vulnerability Scoring System、共通脆弱性評価システム) のスコアと CWE (Common Weakness Enumeration、共通脆弱性タイプ一覧) の情報の前年比の比較は、一部が欠けたものとなっています。この NVD のバックログは、この業界における永続的な問題を示す警告と言えます。ライブラリの数、そして CVE の数が増加を続ける中、組織は高まるリスクを持続可能な方法で管理するための最善の方法を検討する必要があります。また、現在の米国の政治情勢から見て、NVD と米国国立標準技術研究所 (NIST) の完全性および将来の存続は保証されていません。

特定のテクノロジやパッケージ タイプで発見された脆弱性

2024 年に、世界中のセキュリティ研究者は約 33,000 の新規 CVE を公開しました。2023 年比では 27 %増加しており、公開される CVE の数は毎年増加しています。新しいオープンソース パッケージの数が常に増加していることを考えれば驚くべきことではありませんが、CVE の増加率 (前年比で 27%) がパッケージの増加率 (前年比で 24.5%) を上回っていることは真剣に受け止めるべきです。

図 6.1. 2024 年にパッケージ タイプごとに公開された CVE の数

注目すべき最初の傾向は、Debian CVE の前年比での大幅な増加です。2024 年にエコシステムに貢献したパッケージが 4 倍に増えたことを考えると、Debian CVE の増加は驚くべきことではありません。幸いなことに、2024 年の Debian CVE で、重大 CVE と高 CVE はそれほど多くありません。

2023 年のデータと同様に、Maven、npm、PyPI、Conan (今年新たに追加) は、Maven と npm の CVE 総数が前年比で減少したにもかかわらず、重大 CVE の割合が最も高くなっています。しかし、データベース全体の年末のスナップショットを見ると、永続的なリスクが存在することから、npm、Maven、PyPI のリスク レベルはやや低いことが分かります。

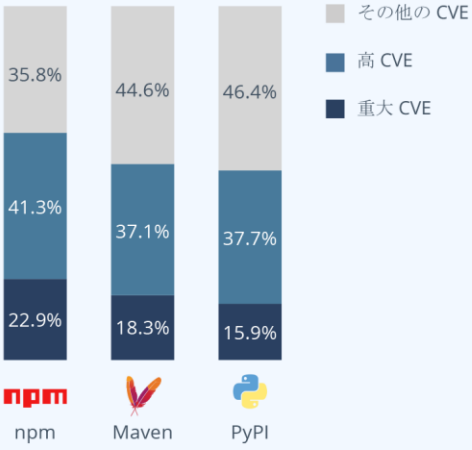


図 6.2. 2024 年末時点の主要エコシステムにおける重大 CVE、高 CVE の割合

削除されたパッケージと非推奨になったパッケージの総数

* この年のデータは未収集

図 7. 削除されたパッケージと非推奨になったパッケージの総数
(JFrog データベース、2024 年)

パッケージはテクノロジ エコシステムに追加されるだけでなく、削除されることもあります。このデータセットで最も注目すべき点は、2024 年と 2023 年に削除された Composer パッケージの数です。JFrog セキュリティ リサーチ チームによる手動レビューの結果、以下の可能性が想定されました。

- 削除されたパッケージのほとんどは、GitHub ソースコード リポジトリが「利用不可」になっています。これは、作成者により削除されたか、非公開にされたことを意味します。
- あるケースでは、削除されたパッケージの名前が変更されただけでした。そのため、`packagist` が名前変更イベントを「削除」としてマークしている可能性があります。
- 一部のパッケージは削除され、「放棄」とマークされていますが、放棄されたパッケージの基準は不明です。
- `packagist` は 2024 年に、GitHub リポジトリが無効なパッケージを削除する新しい自動化を実行したようです。

データ ソースにより削除されたパッケージのレポート方法が異なり、この情報を提供するものもあれば、提供しないものもあります。特定のエコシステムでは、利用可能なすべてのデータを解析し、時系列で比較していますが、この方法では最初のデータ収集よりも前に削除されたパッケージは考慮されません。また、前回の実行以降の情報を含む定期的な更新を受け取り、更新中に削除をレポートするエコシステムもあります。そのため、削除されたパッケージについて完全な情報を常に入手できるとは限りません。

最も一般的な脆弱性のタイプ

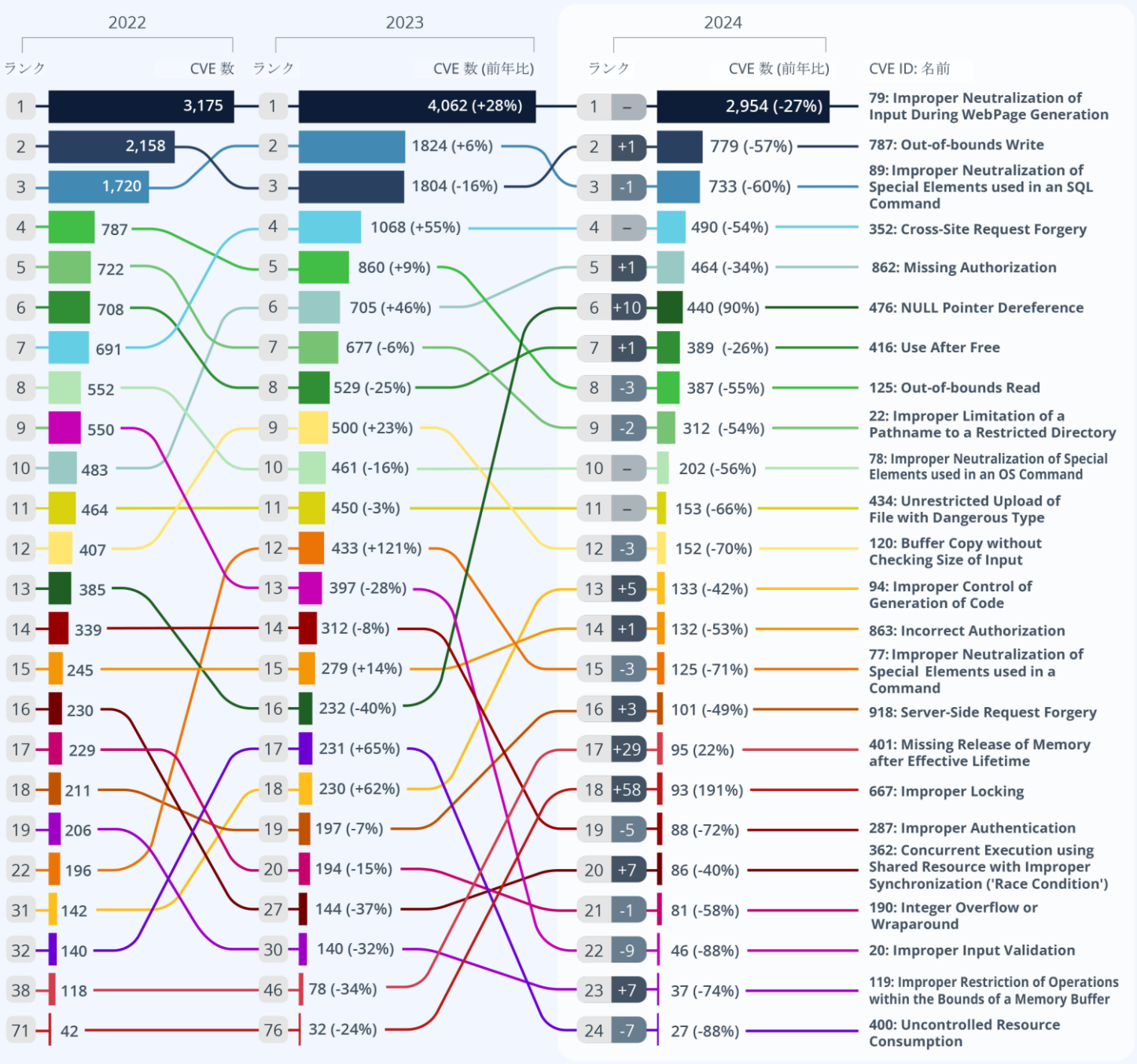


図 8. 2024 年に公開された主な脆弱性 (2023 年、2022 年、2021 年との比較)

2024 年には、243 の固有の CWE ID が CVE に割り当てられ、上位 3 つは前年と変わらず、クロスサイト スクリプティング、境界外書き込み、SQL インジェクションでした。しかし、上位 20 位までに新たに 3 つの脆弱性が加わり、それぞれ急激に増加しました。

401: Missing Release of Memory after Effective Lifetime

2024 **#17** ↑ 2023 **#46**

362: Concurrent Execution using Shared Resource with Improper Synchronization (‘Race Condition’)

2024 **#20** ↑ 2023 **#28**

119: Improper Restriction of Operations within the Bounds of a Memory Buffer.

2024 **#23** ↑ 2023 **#30**

SAST ツールで検出可能な最も一般的な 3 つの CWE (クロスサイト スクリプティング、境界外書き込み、SQL インジェクション) に対処するには、これらのタイプの新たな脆弱性を防ぐため、自動化された SAST ツールを使用してソースコードをスキャンすることを推奨します。また、境界外書き込みは、C/C++ などの低水準 (メモリ セーフでない) プログラミング言語に特有の問題です。

米国政府の提案に従って高水準言語に移行することにより、これらの問題を防ぐことができます。

CWE タイプの前年比の傾向は、ランダムな要因や一時的なイベントの影響を受けやすいことに注意することが重要です。たとえば、NVD のバックログは、2024 年の CWE の普及率の表現にほぼ確実に影響を与えます。すべての CVE が適切にカタログ化されると、これらの数値は変化する可能性があります。たとえば、低水準言語と高水準言語の人気の違いがメモリ破損の脆弱性と高水準/Web の問題数に影響を及ぼす可能性があるため、より長い 20 年間の期間を調査すれば、より有意義な傾向が明らかになるでしょう。特定のテクノロジの人気の変動も、特定のタイプの CWE の影響を受けやすく、これらの傾向に影響を与える可能性があります。

2024 年の注目度の高い CVE の一般的な脆弱性の影響

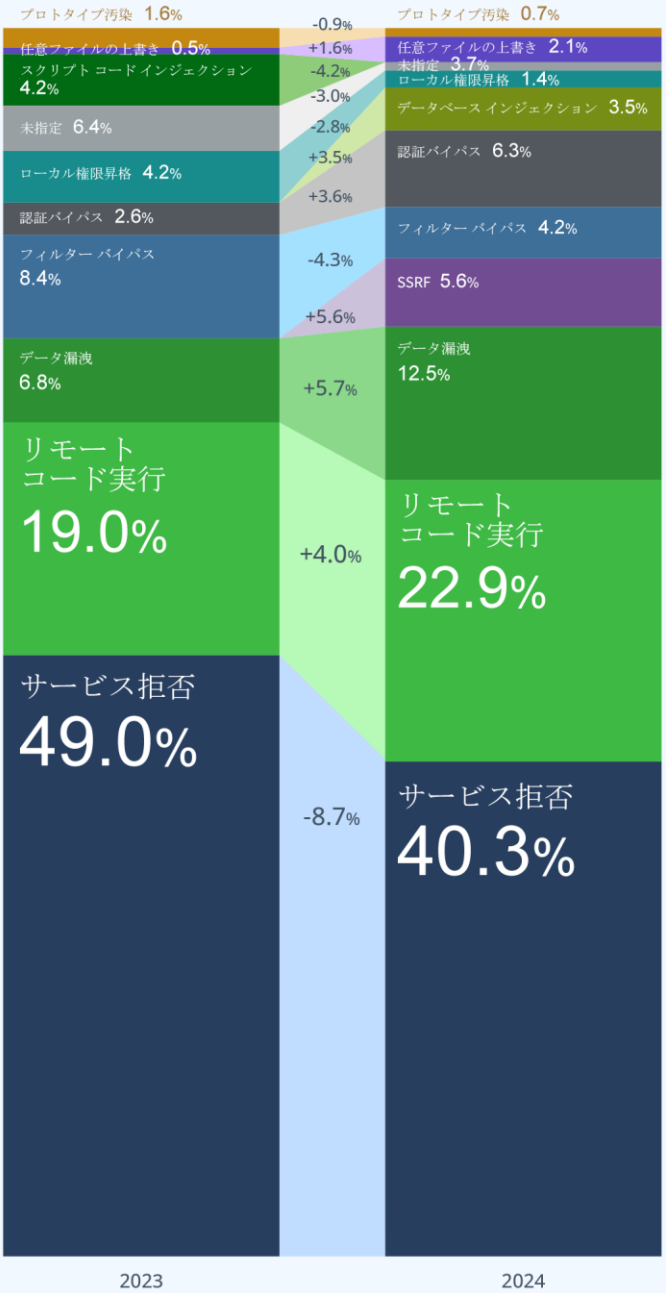


図 9. 2023 年と 2024 年の注目度の高い CVE の一般的な脆弱性の影響

今年、JFrog セキュリティ リサーチ チームは、JFrog のお客様への関連性と潜在的な影響に基づき、140 を超える注目度の高い CVE (HPCVE) を分析しました。サービス拒否は、引き続き一般的な脆弱性の影響の 1 位でした (58)。リモート コード実行 (33) は引き続き 2 位で、割合は 18.9% から 22.9% に増加しました。リモート コード実行はハッカーに壊滅的な制御能力を与える可能性があるため、HPCVE 全体における割合が増加していることは懸念されます。

データ漏洩 (18) は引き続き 3 位で、その割合は急増しました。認証バイパスと、今年の調査で新しく追加された SSRF も増加しました。その一方で、フィルター バイパスは減少しました。

JFrog セキュリティ リサーチ チームは、調査対象の CVE の優先順位付けの際に、いくつかの要因を考慮しています。チームは JFrog のクライアントにとって関連性の高いテクノロジに重点を置いて、主に重大度が「高」および「重大」(CVSS スコアが 7.5 以上) の問題を優先しますが、CVSS スコアが利用できない場合は、機械学習ベースの重大度予測も活用します。また、重大度が「中」または「低」の問題でも、実際に悪用されている脆弱性やメディアで大きく取り上げられた脆弱性は優先します。

ソフトウェア サプライ チェーンに脅威をもたらす脆弱性の重大度

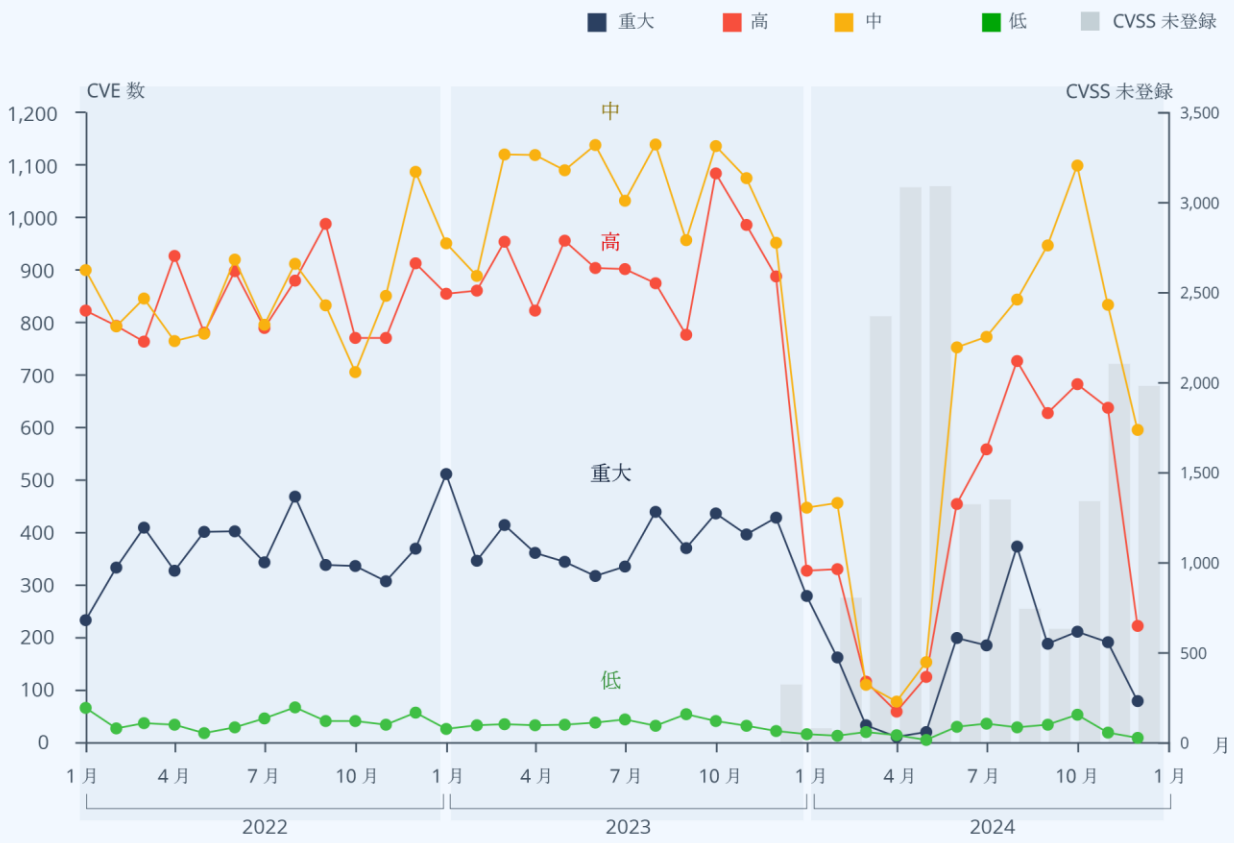


図 10.1. 過去 3 年間の月別および重大度別の CVE 数 (National Vulnerability Database)

データは、2024 年半ばに CVSS の割り当てが大幅に減少し、その後回復したことを示していますが、これは誤解を招く可能性があります。この減少は、NVD が 2024 年 2 月に人員削減に伴う組織再編を行い、CVE の調査を停止したためです。この混乱を解決するため、NVD は 2024 年 6 月に CISA と契約し、CISA の調査を支援していると発表しました。この混乱がなければ、CVSS の数はより予測可能な値になっていたでしょう。リソース不足に対処するため、NVD は現在、CVSS のスコアリングの大部分を Authorized Data Publishers (ADPs、認定データ パブリッシャー) と呼ばれる外部ベンダーに委託しています。現在、CISA が最初で唯一のデータ発行者です。JFrog セキュリティ リサーチ チームは、CISA によりスコアリングされた CVE のスコアリング パターンを継続的に分析しています。初期の分析では、**CISA は NVD よりも誇張された (重大度の重みが高い) スコアを設定している**ことが示されています。将来、他の認定データ パブリッシャーが加わることで、CVE のスコアリングが一致しなくなる可能性も懸念されます。これは、組織がセキュリティ対策の優先順位を決定する際に、考慮しなければならない現実です。

利用可能な NVD データに基づくと、傾向は引き続き一貫しています。重大度が中および高の CVE の数が多く、重大度が低の CVE の数が少なく、重大度が重大の CVE の数はその間です。

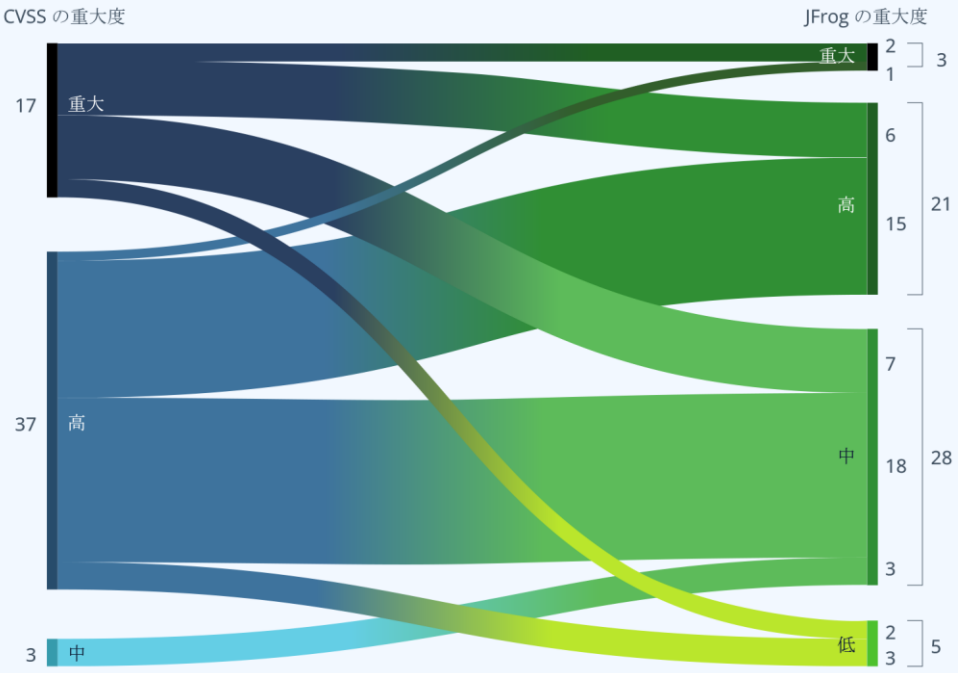


図 10.2. CVE の重大度スコア
(NVD の重大度と JFrog 独自のセキュリティ リサーチ重大度の比較)

しかし、すべての CVE の評価が見た目通りであるとは限りません。JFrog セキュリティ リサーチ チームは、CVE を定期的に評価して実際の影響を判断し、JFrog の重大度評価を割り当てています。JFrog の DevSecOps エキスパートが作成する JFrog の重大度評価は、脆弱性を悪用するための構成要件を考慮しています。CVSS の評価は、脆弱性が悪用される可能性ではなく、脆弱性が悪用された場合の重大度のみを考慮しています。構成や悪用の手法がパッケージや依存関係の標準設定ではないために、脆弱性が悪用される可能性が非常に低い場合もあります。このような過度な CVE スコアリングは、年々懸念されています。

この CVE のスコアが高くなる傾向が懸念される理由は、スコアリング手法の説明が変更されていないためです。スコアリング手法はパッケージに関連するリスクの初期認識を判断する上で中心的な役割を果たすため、過度な CVE スコアリングを行うと、誤検知の可能性が高くなります。

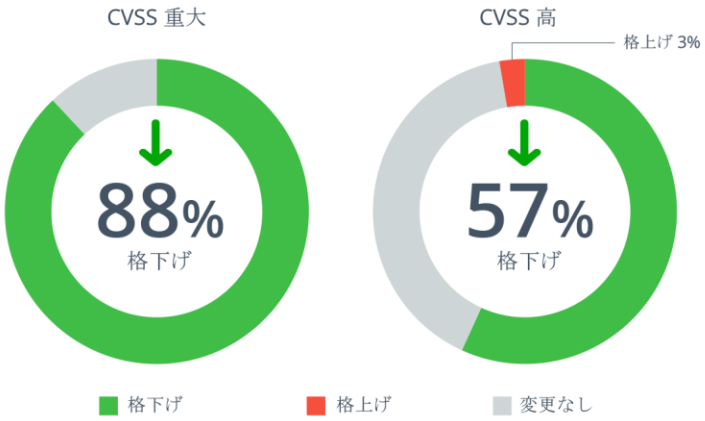


図 10.3

140 の注目度の高い CVE のサンプルに基づく JFrog のセキュリティ調査では、重大 CVE の 88%、高 CVE の 57% が、CVSS スコアリングから予想されるほど深刻ではないことが明らかになりました。

注目度の高い CVE の適用性レーティング

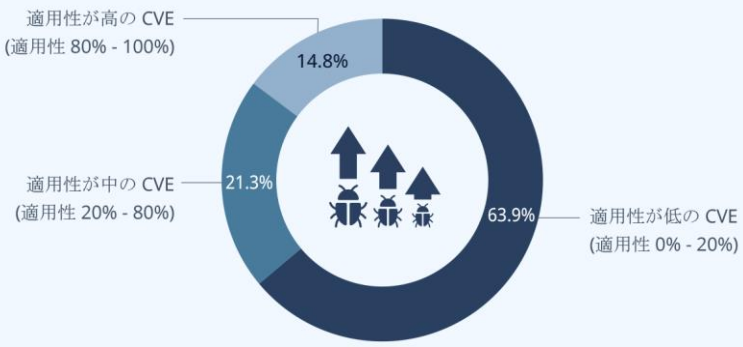


図 10.4. 注目度の高い 183 の CVE の適用性レーティング
(CVE と JFrog データベースを使用した JFrog 独自のセキュリティ調査)

CVE に重大度スコアを割り当てるだけでは、特定のソフトウェア製品に対する脆弱性の影響を評価するには不十分です。JFrog のセキュリティ リサーチ チームは、JFrog の重大度スコアを割り当てるだけでなく、これらの脆弱性が悪用される可能性に影響を与える条件も評価します。このため、JFrog では、特定のソフトウェア製品が悪用される可能性の基準を満たしているかどうか判断する「適用性」スキャナーを作成しています。

JFrog セキュリティ リサーチ チームは、JFrog のお客様の間で最も人気の高いコンポーネントとテクノロジの高 CVE と重大 CVE に重点を置いて、2024 年に公開された 183 の CVE (CVE-2024-*) の適用性スキャナーを作成しました。上記のグラフは、JFrog のお客様から適用可能 (悪意のあるアクターにより悪用される可能性がある) と判断された CVE と、適用不可 (悪用される可能性がない) と判断された CVE の比率を詳細に示したものです。2024 年に JFrog Xray でスキャンされたアーティファクトのうち、適用性が 80% を超える、悪用される可能性が非常に高いと判断された CVE は 27 (15%) のみでした。対照的に、適用性が 0% - 20% の、悪用される可能性が低いと判断された CVE は 117 (64%) でした。

CVE-2024-24792 は、適用性が非常に高い (99.6%) 注目すべき例の 1 つです。この脆弱性は、Go プログラミング言語の TIFF 解析パッケージの一般的な利用によりトリガーされる可能性があり、イメージのアップロードと処理を管理するアプリケーションで頻繁に発生します。ユーザーが操作可能な TIFF イメージを処理するためにライブラリを使用すると、この CVE がトリガーされ、アプリケーションでパニックを引き起こす可能性があることから、ほとんどのシナリオに該当します。

一方、CVE-2024-45490 (C 言語で記述された XML パーサーの Expat に関連) は、適用性が最も低い CVE の 1 つで、適用可能なケースは 10% 未満です。攻撃者がこの脆弱性を悪用するには、ライブラリの API 関数 XML_ParseBuffer() に渡される「len」パラメーターを操作する必要があります。しかし、開発者は通常、`stat` や `XML_GetBuffer` などの関数を使用して XML 文書の長さを提供するため、このシナリオは極めて可能性が低いと考えられます。

注目度の高い CVE の適用性タイプ

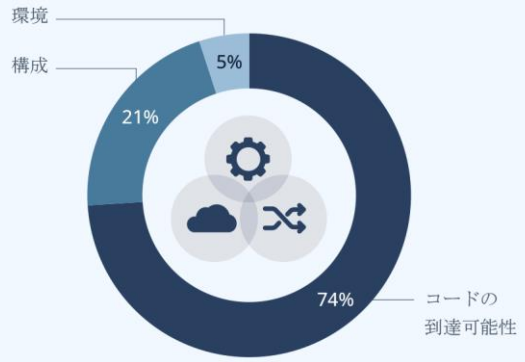


図 10.5. 2024 年の CVE の適用性タイプ
(CVE と JFrog データベースを使用した JFrog 独自のセキュリティ調査)

JFrog セキュリティ リサーチ チームは、これらの CVE がアプリケーションで到達可能か、悪用可能かについても調査しました。脆弱性が適用可能か、悪用可能かを判断するには、従来の呼び出し到達可能性分析により脆弱なコードの到達可能性を評価するだけでは不十分であり、アプリケーションやライブラリの構成設定、利用するオペレーティング システムの環境条件も調査することが不可欠です。この包括的なアプローチにより潜在的なリスクを包括的に評価できますが、到達可能なコードを特定するだけでは、脆弱性の悪用に大きな影響を与える重要な要素を見落としてしまう可能性があります。

たとえば、有名な「Sudoedit バイパス」、CVE-2023-22809 の脆弱性の適用性を判断するには、Sudo の設定ファイル - sudoers - を調べて、特定のデフォルト以外の設定を探す必要があります。脆弱なコンポーネント「sudo」はスタンドアロン ユーティリティであり、ファーストパーティ コードで呼び出すことができるコード ライブラリではないため、コードの到達可能性を調べて脆弱性が適用可能かどうか判断する方法はありません。

一部の悪意のあるパッケージは危険性がより高い

2024 年のレポートでは、npm エコシステムにおける悪意のあるパッケージの蔓延について取り上げました。2024 年末に行った主要パッケージ エコシステムのレビューでは、悪意のあるパッケージの存在という点で、npm が依然として最悪の存在でした。

今年 Hugging Face エコシステムにアップロードされた悪意のあるモデルが約 6.5 倍に増加したことは、その人気の急上昇を考えると当然のことと言えるでしょう。JFrog セキュリティ リサーチ チームが特に注目した、3 つの悪意のある攻撃を紹介します。

XZ Utils バックドア

3 月 29 日に、主な Linux ディストリビューションで広く利用されているパッケージである XZ Utils に、不正なリモート SSH アクセスを可能にする悪意のあるコードが含まれているという、重大なセキュリティ侵害がレポートされました。バージョン 5.6.0 および 5.6.1 で発見されたこの高度なバックドアは、OpenSSH サーバーのルーチンを改変し、特定の攻撃者が認証前に任意のペイロードを実行できるようにすることにより、被害者のマシンを乗っ取るものでした。

出典 >

Docker Hub

Docker Hub をターゲットとした最近のマルウェア攻撃により、コンテナー イメージの代わりに悪意のあるメタデータを含む何百万もの「イメージレス」リポジトリが作成されました。驚くべきことに、これらのパブリック リポジトリの約 20% (約 300 万) が、海賊版を宣伝するスパムからマルウェアやフィッシング サイトまで、自動アカウントによりアップロードされた有害なコンテンツをホストしていました。

出典 >

Hugging Face

AI モデルの監視により、Pickle ファイルのロード中にコードを実行して、攻撃者にコネクトバック シェルとバックドア経由で侵入したマシンの完全な制御権を与えるモデル ファミリーが明らかになりました。このサイレント侵入は、被害者が侵入に気付くことなく、重大なリスクをもたらし、重要なシステムへのアクセスを許可し、大規模なデータ侵害や企業スパイ活動につながるものです。

出典 >

コードに潜むその他のリスク要因

CISO とアプリケーション セキュリティ チームは、オープン ソース コミュニティから導入する内容を精査することが重要であることを既に認識しています。しかし、包括的なアプリケーション セキュリティで注意すべき点は他にもあります。

設定ミスなどのミス – 人的ミスの影響

2024 年には、設定ミスなどのミスによりデータが漏洩したセキュリティ インシデントが多く発生しました。

2024 年 4 月

Home Depot は、サードパーティの SaaS ベンダーが従業員データの一部を漏洩したことにより、従業員 1 万人の個人情報が漏洩するというデータ侵害を受けました。

出典 >

2024 年 9 月

Azure SharePoint サイトに保存されていた約 2,000 人の Fortinet の顧客データがハッカーにアクセスされ、インターネット上に漏洩しました。

出典 >

2024 年 8 月

数千の Oracle NetSuite の顧客が、NetSuite SuiteCommerce や NetSuite Site Builder でビルドした外部向けのストアから、認証されていないユーザーに機密データを意図せず漏洩していました。

出典 >

2024 年 9 月

ローコード SaaS プラットフォームの Microsoft Power Pages で、アクセス制御の設定ミスによる、何百万人ものユーザーに影響を与える可能性のある重大なデータ漏洩が発見されました。

出典 >

2024 年 9 月

1,000 を超える ServiceNow エンタープライズ インスタンスの設定ミスにより、機密性の高い企業情報を含むナレッジベース (KB) の記事が外部ユーザーや潜在的な脅威アクターに公開されていることが判明しました。

出典 >

2024 年 12 月

フォルクスワーゲンの自動車ソフトウェア会社、Cariad に関連するデータ漏洩は、2024 年に発生した最もインパクトの強い SaaS の設定ミスの 1 つです。このインシデントにより、約 80 万台の電気自動車から収集された、正確な車両の位置や運転者の名前に結び付けられる可能性のある情報を含むデータが漏洩しました。

出典 >

バイナリ アーティファクトで漏洩したシークレットの状況

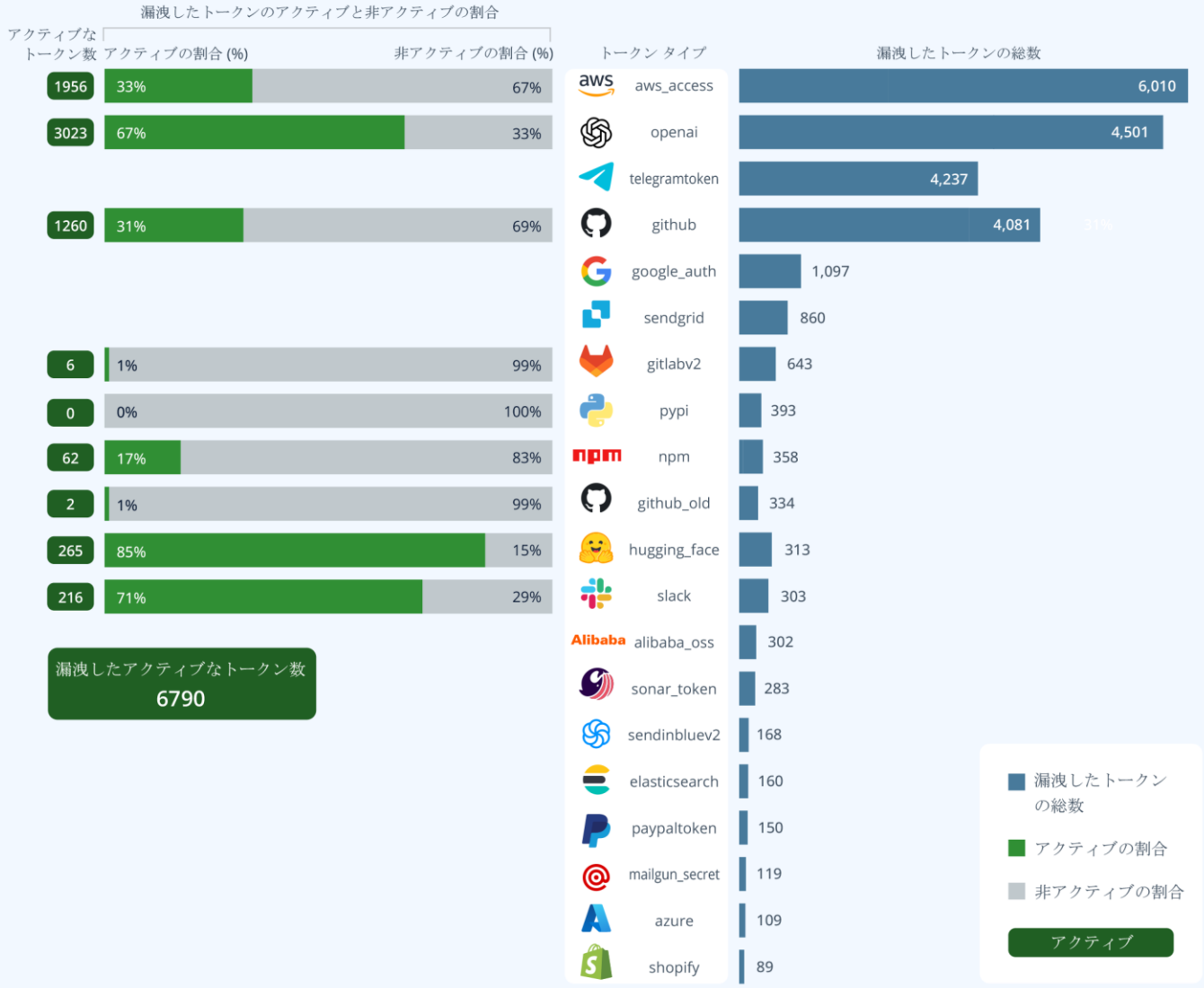


図 11.1. 2024 年に最も漏洩したトークン タイプ上位 20

JFrog セキュリティ リサーチ チームは、最も一般的なオープンソース ソフトウェア レジストリである DockerHub、npm、PyPI の何百万ものアーティファクトをスキャンしました。今年は、アクティブなトークン (データ収集時に使用可能なトークン) が見つかった場所も記録しました。

発見されるトークンのタイプは、ほぼすべて、年々増加傾向にあり、漏洩したシークレットの総数は前年比で 66% 増加しました。漏洩したトークン数の上位は前回のレポートと同じでしたが、AWS (70% 増加)、OpenAI (103% 増加)、Telegram (62% 増加)、GitHub (82% 増加) と、前年比で大幅に増加しました。GCP トークンも前年比で 86% 増と大幅に増加しました。

Hugging Face トークンは、今年 JFrog セキュリティ リサーチ チームのスキャナーに追加された新しいトークン タイプで、オープンソース モデルとデータセットの人気の高まりを示すものです。Hugging Face トークンは、リストの他のトークンと比較して、アクティブ トークンの割合が最も高くなっています (約 85% がアクティブ)。JFrog セキュリティ リサーチ チームはデータ収集時点で 6,790 のアクティブなシークレットを特定しており、悪意のあるアクターが独自システムにアクセスするための大量の潜在的なソースが存在することが明らかになりました。

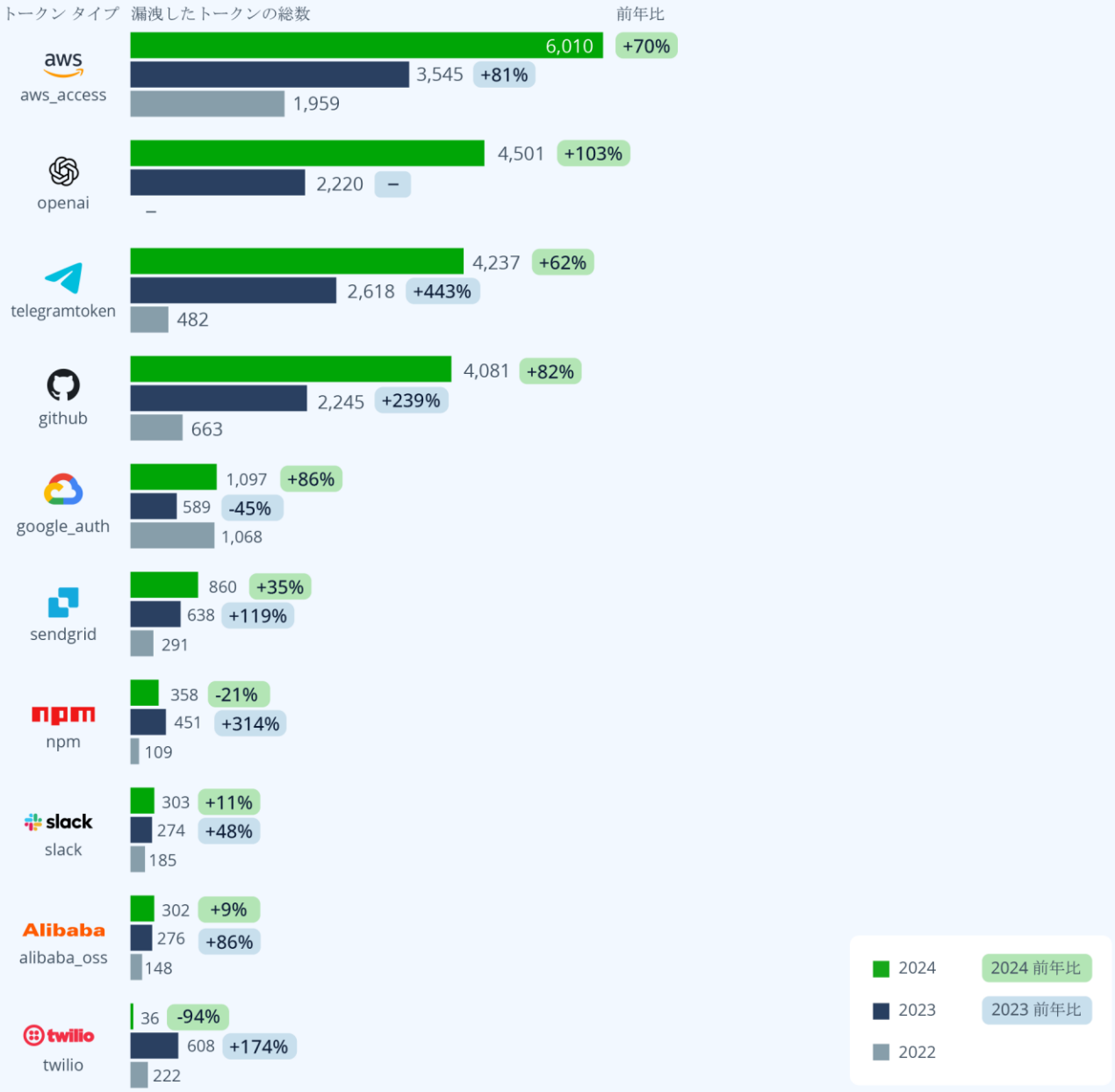
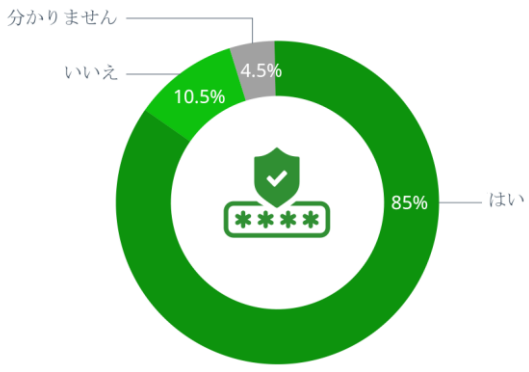


図 11.2. 最も一般的な漏洩したトークンの数と前年比

あなたの組織ではコード ベースに残されたシークレットや漏洩したトークンを検出するためのセキュリティ対策を講じていますか? (委託調査、2024 年)

組織は、シークレットが悪意のあるアクターや一般の人々の手に渡らないようにするため、具体的な投資を行っています。しかし、対策を講じていない、または講じているかどうか不明な組織が 15% もあります。漏洩したシークレットの状況と、発見されたほぼすべてのタイプのトークンで増加が見られたという事実を考えると、自らを脆弱な状態で放置している回答者の割合が非常に高いことが懸念されます。



シークレットの漏洩はどれほど深刻な事態になり得るか?

2024 年 6 月に、JFrog セキュリティ リサーチ チームは、Docker Hub でホストされているパブリック Docker コンテナーで、Python、PyPI、Python Software Foundation の GitHub リポジトリへの管理者アクセス権を持つアクセス トークンの漏洩を発見し、レポートしました。

JFrog セキュリティ リサーチ チームは、コミュニティ サービスとして、Docker Hub、npm、PyPI などのパブリック リポジトリを継続的にスキャンし、悪意のあるパッケージや漏洩したシークレットを特定しています。チームは、攻撃者がそれらを悪用する前に、発見した情報を関連するメンテナーにレポートしています。JFrog では、同様の方法で漏洩した多くのシークレットを目にしますが、今回のケースは、悪意のあるアクターに渡った場合、その影響が計り知れない、例外的なものでした。すべての PyPI のパッケージのみでなく、Python 言語自体に悪意のあるコードが注入される可能性がありました。

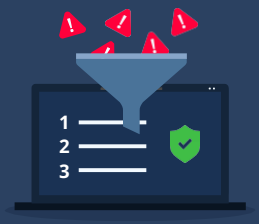
JFrog セキュリティ リサーチ チームは漏洩したシークレットを特定し、速やかに PyPI のセキュリティ チームにレポートし、わずか 17 分でトークンは無効化されました。

今回は大惨事を回避できましたが、このインシデントは、たった 1 つのシークレットの漏洩が壊滅的な状況をもたらす可能性を示しています。PyPI は世界有数のリポジトリの 1 つであることを考えると、その影響は広範囲に及び、無数のユーザーとプロジェクトに影響を与えていた可能性があります。高度にメンテナンスされ、広く利用されているインフラストラクチャである Python/PyPI でこのような侵害が発生したことは、あらゆるプラットフォームと言語に脆弱性が存在する可能性があり、その脅威がいつでも誰にでも襲いかかる可能性があることを示したと言えるでしょう。

重要なポイント

データの冗長性の必要性

NVD の脆弱性データのみに依存している組織やセキュリティ ツールは、過去 1 年間に NVD が経験した大幅なバックログの遅延により、重要な CVE 情報を見逃すリスクがあります。これらの遅延は、新たに公開された脆弱性とその潜在的な影響がデータベースに迅速に反映されず、組織が知らないうちに新たな脅威にさらされるリスクがあることを意味します。このリスクを軽減するには、ベンダーのアドバイザリや脅威インテリジェンス フィードなどの追加のソースで NVD データを補完し、重大な脆弱性をタイムリーに認識することが不可欠です。組織は、セキュリティ スキャン ツールのデータ ソースを評価し、広範なカバレッジと冗長性を確保する必要があります。

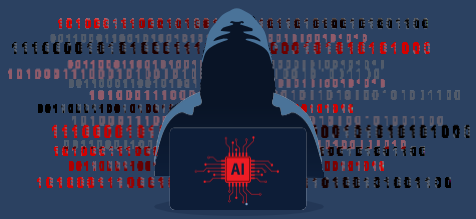


適用性、影響、優先順位付け

対処すべき CVE の数がますます増加する中、セキュリティ チームと開発チームは、すべての脆弱性をトリアージする作業に追われ、他の業務に支障が出る可能性があります。真に重要な脆弱性の対処に注力するには、アプリケーションにおける CVE の適用性、攻撃ベクトル、潜在的な影響を理解することが不可欠です。JFrog セキュリティ リサーチ チームは、CVSS スコアでリスク評価が過大評価されているケースを継続的に発見しています。CISA が最初の Authorized Data Publishers (認定データ パブリッシャー) として CVE レコードの拡充に貢献するようになってから、この傾向は加速しているようです。

シークレットの漏洩は誰にでも起こり得る

組織は、漏洩したシークレットに対する保護に常に注意を払い、個人プロジェクトやコミュニティ プロジェクトに取り組む開発者にも保護を拡大するように努める必要があります。開発者のシステムが個人プロジェクトから侵害された場合でも、その影響が企業システムにも及ぶ可能性があります。漏洩したシークレットや漏洩したトークンが誰かに発見された場合、その影響は非常に深刻になる可能性があります。JFrog が発見した Python/PyPI のシークレットのケースでは、そのトークンの所有者が Python、PyPI、Python Software Foundation のすべてのリポジトリへの管理者アクセス権を持ち、非常に大規模なソフトウェア サプライ チェーン攻撃を行う可能性がありました。Python/PyPI に起こり得ることは、誰にでも起こり得ます。



悪意のあるアクターの高度化

悪意のあるアクターは、より創造的で機知に富んだ手法を用いてソフトウェア サプライ チェーンへ侵入するようになっています。XZ Utils バックドアのケースでは、攻撃者は数年かけて OSS 開発者としての信用を築き、コード レビューによる検出を回避するため高度に難読化されたコードを使用していました。AI コード アシスタントが「幻覚的な」ライブラリを推奨する事例を特定し、悪意のあるコードを含むライブラリを迅速に作成することで AI ツールを悪用しているアクターもいます。組織は、評価の高いオープンソース プロジェクトであっても警戒を解かず、「一夜漬け」のライブラリがサプライ チェーンに誤って取り込まれるのを防ぐ運用リスク ポリシーを策定する必要があります。

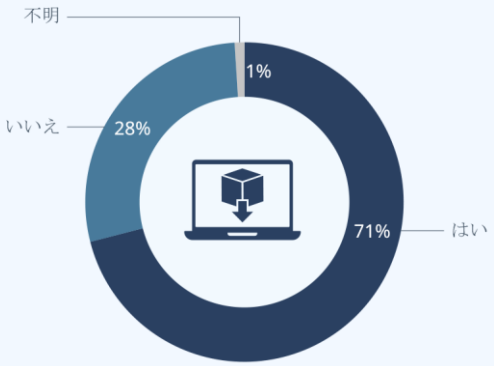
今日の組織における セキュリティ対策の 適用状況

今年は、1,402 人のセキュリティ、DevOps、エンジニアリングの専門家を対象にアンケートを実施しました。アンケートの質問を拡大し、他の JFrog がスポンサーとなった調査レポートの結果も取り入れて、ソフトウェア開発ライフサイクル (SDLC) 全体を通じて、チームがアプリケーションのリスクをどのように管理しているか、より包括的な視点を得ることができました。ほとんどのチームがセキュリティ フレームワークとツールを導入していましたが、サードパーティ製のパッケージやライブラリをインターネットから直接ダウンロードするなど、リスクの高いアクティビティが蔓延していることに驚きました。

調達制限

組織がソフトウェア サプライ チェーンにおけるリスクに対処するためにできる最善策の 1 つは、リスクの侵入を完全に防ぐことです。このためには、開発フェーズでセキュリティ対策を浸透させることを意味する「シフトレフト」をさらに進め、リスクがそもそもソフトウェア サプライ チェーンに入る前にブロックする「レフトオブレフト」のアプローチが必要です。

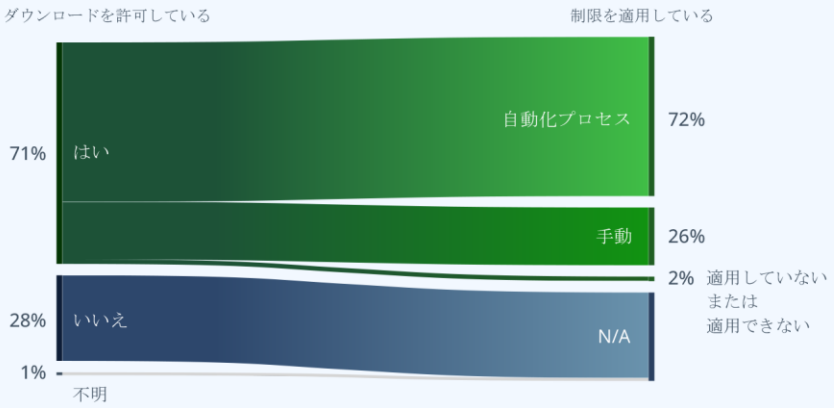
Q **あなたの組織では、開発者がパブリック レジストリやその他のインターネットのソースからパッケージやその他のソフトウェア コンポーネントを直接ダウンロードすることを許可していますか?** (委託調査、2024 年)



驚くべきことに、71% の組織が開発者にインターネットからソフトウェア コンポーネントをダウンロードすることを許可していました。開発者がパッケージやライブラリをインターネットから直接ダウンロードすることはリスクがあまりにも大きい (1 人の開発者のマシンから組織全体が攻撃にさらされる可能性がある) ため、ベスト プラクティスは、これらのダウンロードを制限することです。開発者がインターネットから直接ダウンロードすることを許可すると、何をダウンロードしているか把握できないため、トレーサビリティも損なわれます。

しかし、これほど多くの組織がダウンロードを許可しているという事実は、そういったニーズがあることを示しています。アップストリームのパブリック レジストリをプロキシできるアーティファクト管理ソリューションは、この点で役に立ちます。プロキシ機能を備えたアーティファクト リポジトリは、ソフトウェア サプライ チェーンに入るすべてのコンポーネントの中央制御ポイントとして機能し、チームがそれらのコンポーネントを追跡および保護できるようにします。このようなソリューションを使用することで、組織はこの種のアクティビティに関連するリスクを安全に軽減して、計り知れない規模の潜在的な損害を防ぐことができます。

Q **あなたの組織では、パブリック レジストリやその他のインターネットのソースからパッケージやその他のソフトウェア コンポーネントを直接入手することに対する調達制限をどのように追跡および適用していますか?** (委託調査、2024 年)



制限を適用している回答者の 4 分の 1 以上 (26%) が、パブリック レジストリやその他のインターネットのソースからパッケージやその他のソフトウェア コンポーネントを直接入手することに対する調達制限を手動で追跡および適用していると回答しています。

制限を適用している回答者の 72% が自動化プロセスを導入しているという数字は意外に高いのですが、この数字は回答者が SDLC のどの段階を指しているかによって左右される可能性があります。たとえば、開発者がコード ベースにチェックインする前にパッケージと依存関係を手動で調査していて、自動化プロセスが CI/CD サイクル中、またはリリース ビルドの監査中に後から実行される可能性があります。このアプローチの課題は、開発者の作業のやり直しが発生し、このレポートで前述したように、SDLC の早い段階でリスクが発生する可能性があることです。

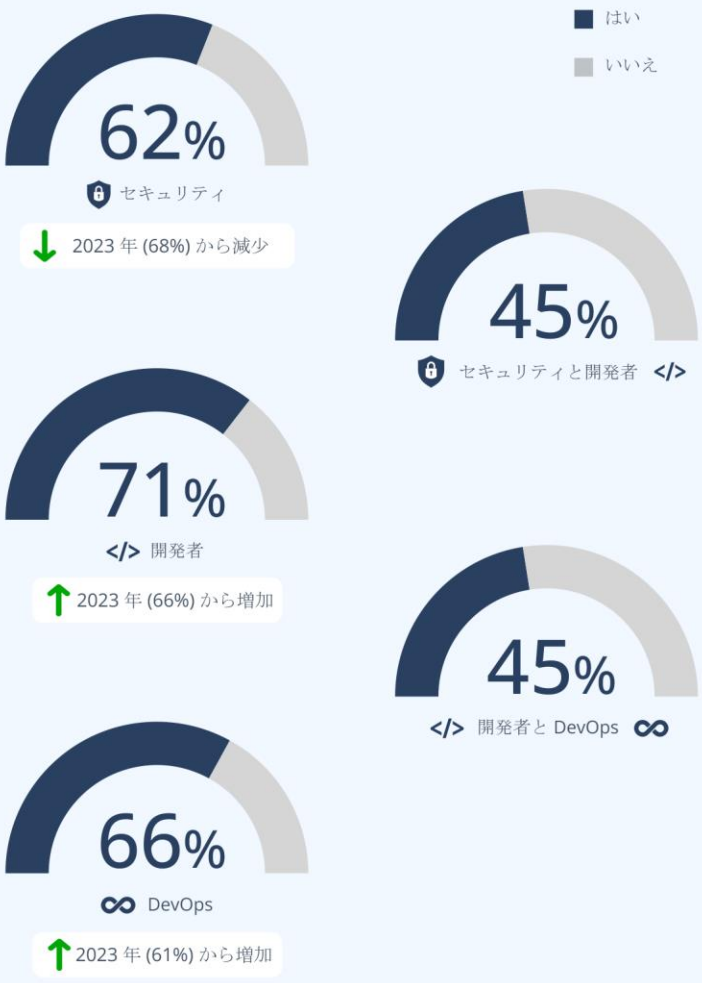
開発者がインターネットから直接パッケージをダウンロードすることを許可し、調達制限を実施するために**手動の**アプローチを活用していると回答した回答者は、制限を適用している回答者の 26% (全体の 19%) でした。これには膨大な量の難しい手動の作業が必要になるため、リスクをブロックするための効果的なアプローチとは言えません。

開発者がインターネットから直接コンポーネントをダウンロードすることを許可し、調達制限を追跡および実施するために**自動化**プロセスを活用していると回答した回答者は、制限を適用している回答者の 72% (全体の 52%) でした。

Q **ソフトウェア パッケージ、ライブラリ、フレームワークの最新バージョンを取得するプロセスを管理しているのは、セキュリティ (担当者) ですか、それとも開発者ですか? 該当するものをすべて選択してください。**(委託調査、2024 年)

以前と比較すると、開発者が最新パッケージの管理でより積極的な役割を担うようになりましたが、セキュリティ担当者は依然として (特に使用するパッケージのレビューと承認に対する) 責任を負っています。取得プロセスを担当するチームを組み合わせた、「セキュリティ (担当者) + 開発者」や「開発者 + DevOps」を活用している組織もあります。

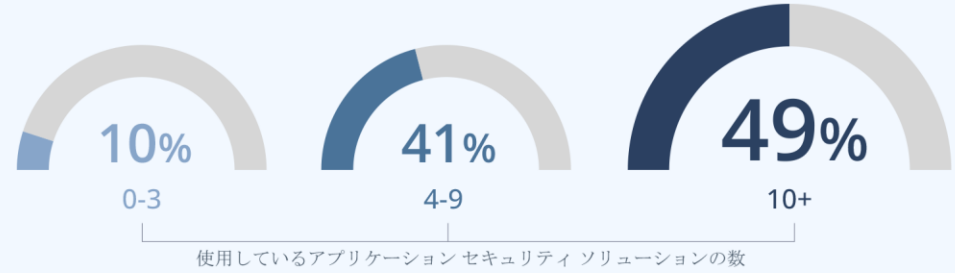
速度を向上するには、開発者が新しい最新バージョンのライブラリをセルフサービスで導入し、セキュリティ ポリシーに準拠しているパッケージの承認を自動化できるアプローチとソリューションを採用する必要があります。アプリケーション セキュリティ担当者であれセキュリティ担当者であれ、担当チームが全体的なリスク管理戦略に統合できる免除管理プログラムもサポートする必要があります。

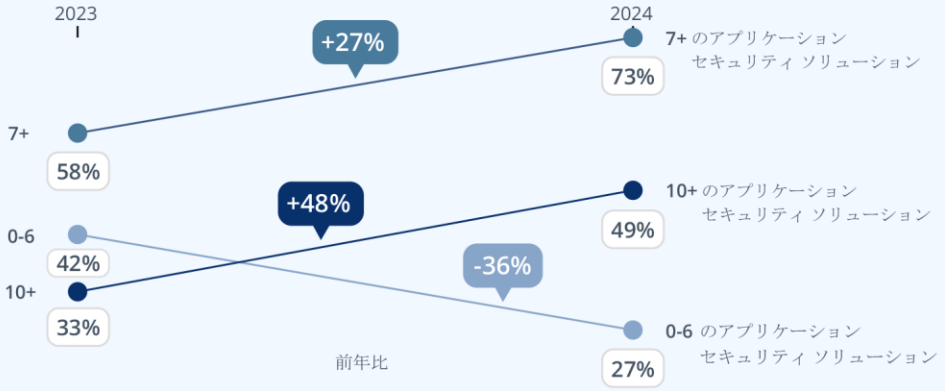


スキャン、スキャン、スキャン

組織はこれまで以上に多くのセキュリティ ツールを使用していますが、依然としてカバレッジのギャップが存在します。コードとバイナリ両方のスキャンの不足、そして SDLC と本番環境におけるスキャンの一貫性の欠如は、よくある盲点です。

使用しているアプリケーション セキュリティ ソリューションの数はいくつですか? (委託調査、2023 年および 2024 年)





回答者は、2023 年と比較して、2024 年は使用しているアプリケーション セキュリティ ソリューションの数が増加していると回答しています。2024 年末までに、7 つ以上のアプリケーション セキュリティ ソリューションを使用していると回答した回答者は 73% と、前年の 58% を大きく上回っています。

これは、市場におけるツール統合への注目度の高さや、安全なソフトウェア開発プロセスの合理化を希望する JFrog のお客様の声から考えた予想とは矛盾する結果です。データでは、組織が過剰なカバレッジを維持しつつ見逃しのリスクを防ぐために、複数のスキャン ソリューションを維持したまま重複する結果を除外できる新しいツール カテゴリの ASPM (Application Security Posture Management、アプリケーション セキュリティ態勢管理) を使用していることが示されています。しかし、ASPM はセキュリティ ツールのスプロールを抑える「応急処置」であり、解決策ではありません。組織が統合への取り組みに再び焦点を当てていることから、使用するセキュリティ ツールの数の増加が来年も続くとは予想していません。

Q あなたの組織では、コード レベル、バイナリ レベル (または両方) でセキュリティ スキャンを適用していますか?(委託調査、2024 年)

コード スキャンとバイナリ スキャン

43%

↓ 2023 年 (56%) から減少

コード スキャンのみ

29%

↑ 2023 年 (27%) から増加

バイナリ スキャンのみ

25%

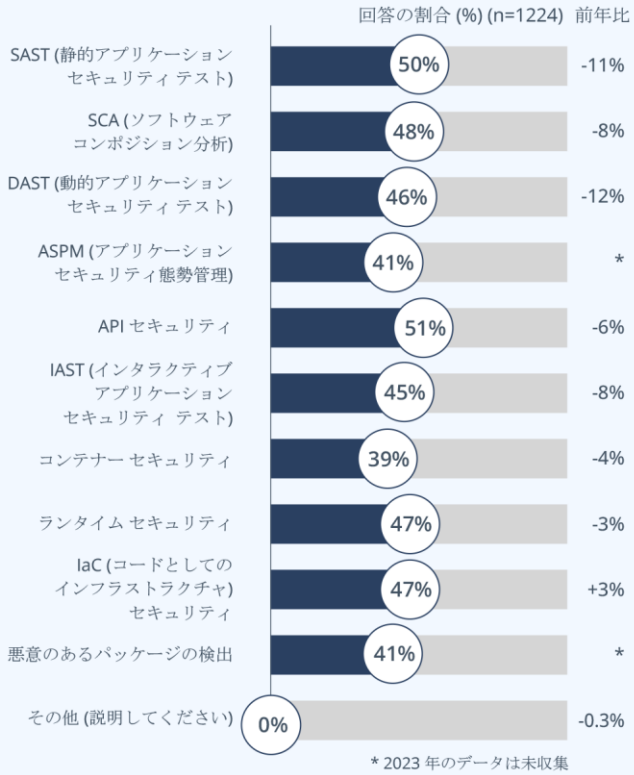
↑ 2023 年 (12%) から増加

今年は、バイナリ スキャンのみでセキュリティ スキャンを適用している回答者が倍増しました。このレベルでセキュリティを適用していると回答した回答者は 25% で、2023 年はわずか 12% でした。

コード レベルとバイナリ レベルの両方でセキュリティ スキャンを適用していると回答した回答者は 43% で、2023 年の 56% から若干減少しました。リスクを防止して可能な限り早期に発見するには、コード レベルとバイナリ レベルの両方でスキャンを行うのが理想的であるため、これはやや憂慮すべき傾向です。このアプローチを採用する理由には、バイナリ レベルでのみ現れる特定のタイプの脆弱性が存在することが含まれます。

たとえば、バイナリに注入されたシークレットやコンパイラにより挿入されたメモリ破損は、ソースコードに存在しないセキュリティ問題や、最終的に本番環境で使用するビルドに誤って残されるセキュリティ問題を引き起こす可能性があります。

Q 使用しているアプリケーション セキュリティ ソリューションのタイプは何ですか? (委託調査、2024 年)

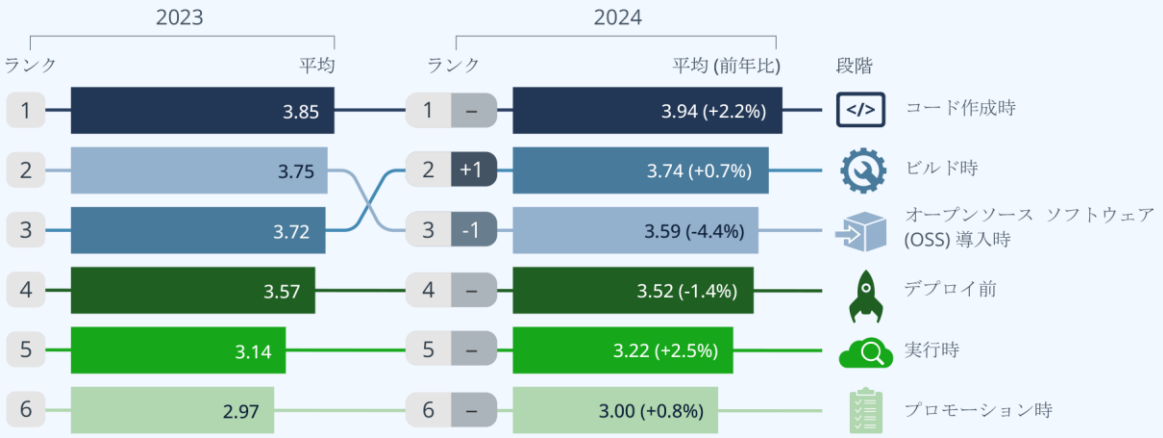


使用率が圧倒的に高いツールは 1 つもありません。使用率が 50% 以上なのは、API セキュリティと SAST の 2 つだけです。

シフトレフトの取り組みが継続的に重視されていることから、SAST の使用率が高いのは当然のことです。また、悪意のあるアクターによる悪用に対して API が潜在的な弱点となる可能性がある最新のマイクロサービス アプリケーションの普及を考えると、組織が API セキュリティ ツールに投資しているのも当然のことと言えます。

ツールのタイプごとの使用率は前年と変わっていませんが、特定のツールを使用していると回答した回答者の割合は全体的に減少しています。このレポートで前述したように、セキュリティ ツールの総数が増加していることを考えると、これは特に興味深い点です。使用しているツールのタイプが重複しているか、異なるチームがそれぞれ独自に好みのセキュリティ ツールを使用していて、他のチームの好みのツールと同じ機能を提供していることを示している可能性が考えられます。セキュリティ ツールの重複やギャップを特定するため、組織はセキュリティ ツールの監査を検討する必要があるでしょう。

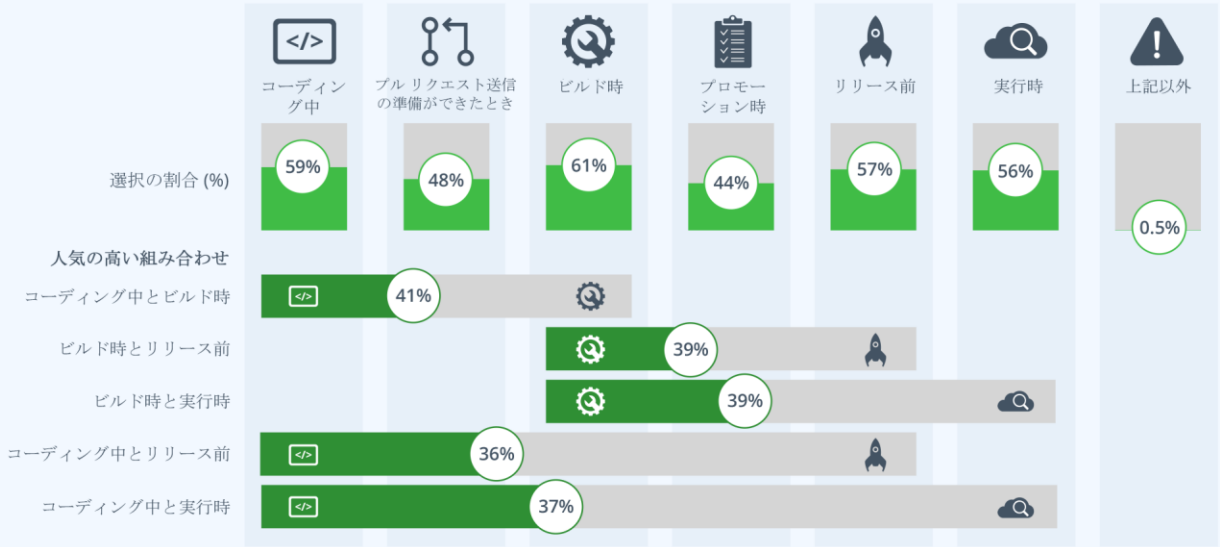
SDLC のどの段階でセキュリティを適用するのが最適だと思いますか? (委託調査、2024 年)



ソフトウェア開発ライフサイクルのどの段階でセキュリティを適用するのが最適かという質問に対する上位 3 つの回答は前年と同じでした。

コード作成時	ビルド時	オープンソース ソフトウェア導入時
M=3.94	M=3.74	M=3.59

あなたの組織では通常、開発のどの段階でセキュリティ スキャンを適用していますか? 該当するものをすべて選択してください。(委託調査、2024 年)



「コーディング中」は、SDLC で組織がセキュリティ スキャンを実行する段階として最も多くレポートされています。回答者の約 5 人に 3 人が、組織が通常この段階でセキュリティ スキャンを実行していると回答しています。

41%	コーディング中	+	ビルド時
39%	ビルド時	+	リリース前
30%	ビルド時	+	実行時
36%	コーディング中	+	リリース前
37%	コーディング中	+	実行時

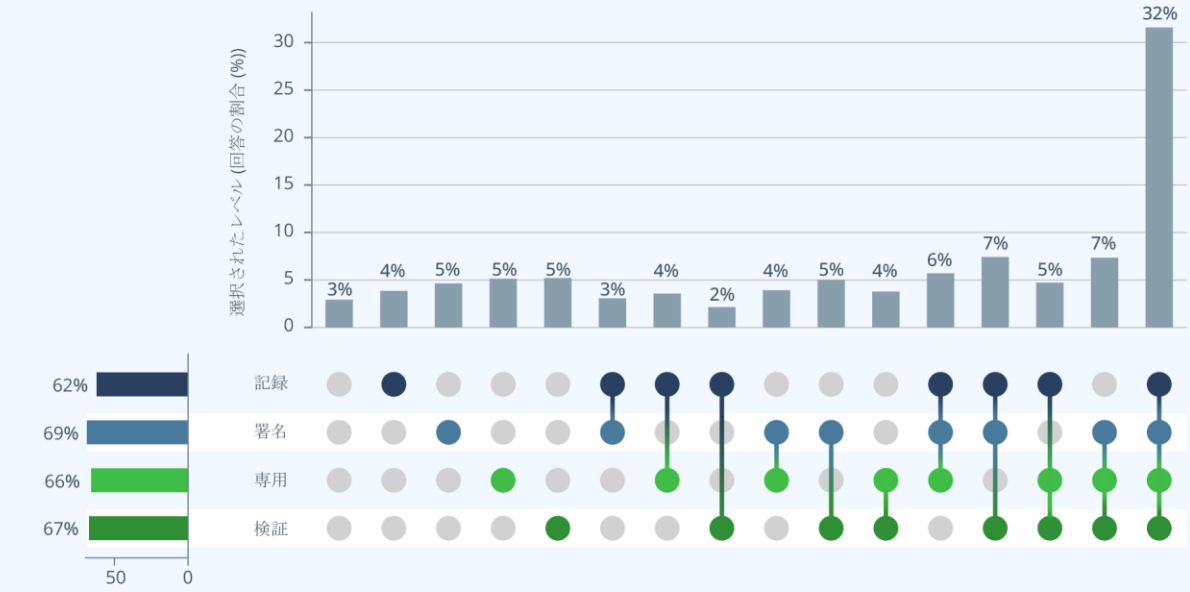
アプリケーション パイプライン全体にわたる可視性と制御の確立

アプリケーションをビルドする際に、アプリケーション レベルで包括的にリスクを管理する必要があることは明らかです。組織は既に SDLC 全体にわたりアプリケーションを定義して追跡していますが、制御とトレーサビリティのレベルは大きく異なります。リスクを確実に管理し、リリースするソフトウェアの信頼性を確保するには、これら 2 つの要素を強化することが不可欠です。

制御は、調達段階、つまり「アプリケーション」を作成する前から始まります。開発プロセスに統合するコンポーネントとライブラリは、最終的な製品のセキュリティ態勢を根本的に形作ります。

サードパーティのリソースを慎重に評価および選択することにより、組織はリスクがアプリケーションで表面化する前に、リスクを軽減することができます。

Q **あなたの組織では、以下のどのレベルのセキュリティ フレームワークを実装していますか?** (委託調査、2024 年)

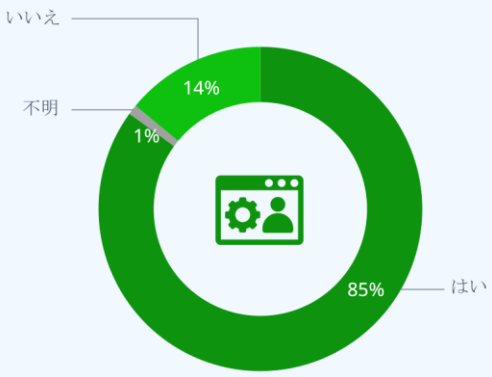


- 記録 - パッケージのビルド メタデータを記録
- 署名 - パッケージのビルド データとメタデータに署名
- 専用 - パッケージを専用のホストでビルド
- 検証 - パッケージの署名を公開前に検証

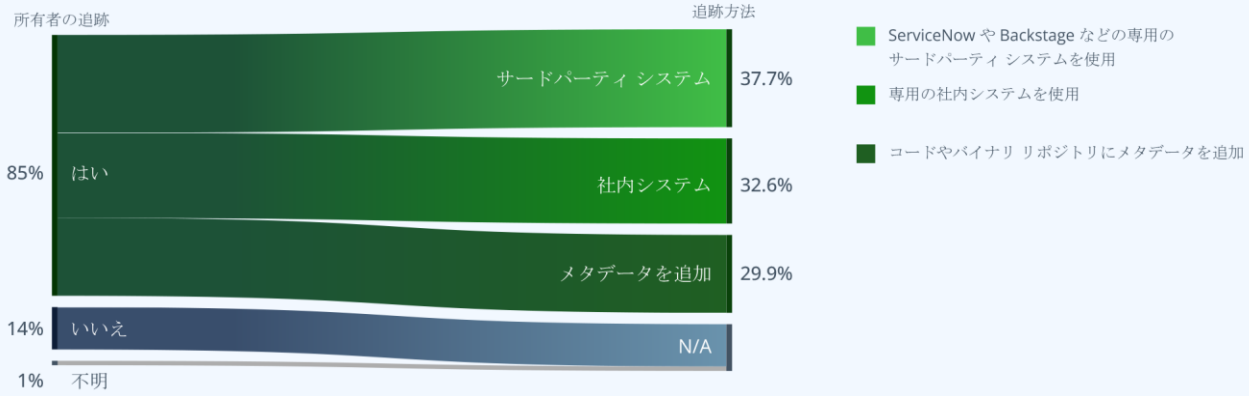
調査対象の組織の大多数は、ソフトウェア サプライ チェーンのセキュリティと整合性を向上するため、SLSA (サルサ、Supply-chain Levels for Software Artifacts) などのフレームワークを活用しています。少なくとも 1 つの SLSA レベルが広く採用されており、回答者の約 3 分の 1 はすべての SLSA レベルを採用しています。

各アプリケーションの所有者の追跡

Q **組織でビルドする各アプリケーションについて、アプリケーションの所有者 (チーム/個人) を追跡していますか?** (委託調査、2024 年)



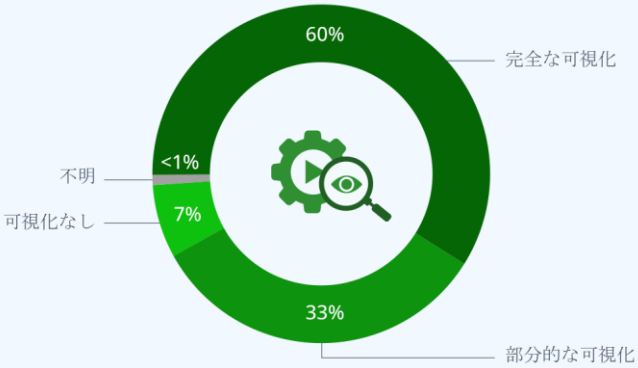
Q **組織でビルドする各アプリケーションについて、アプリケーションの所有者をどのように追跡していますか?**(委託調査、2023 年および 2024 年)



アプリケーションと、アプリケーションを構成するさまざまなマイクロサービスの所有者の追跡は、問題の迅速な解決、アプリケーション間の相互依存関係の把握、適切なガバナンスと事業継続計画の確立を含む、多くの理由から不可欠です。ほとんどの組織はビルド中にアプリケーションの所有者を追跡していますが、その方法は大きく異なります。回答は、専用のサードパーティ システム、専用の社内システム、コードやバイナリ リポジトリにメタデータを追加、にほぼ均等に分かれています。これらの 3 つ以外の方法を使用しているとレポートした回答者は 1 人もいません。

本番環境で実行しているソフトウェアの出所 (サービスのコードを誰がコミットしたか、どのようなテストと検証を経たか、依存関係の取得元など) を可視化できていますか? (委託調査、2023 年および 2024 年)

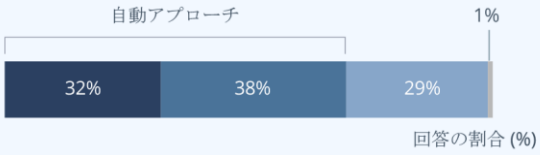
本番環境で実行しているソフトウェアの出所を完全に可視化できていると回答した組織は 60% にすぎません。 部分的に可視化できていると回答した組織は約 3 分の 1 (33%) で、可視性がない、または出所が不明と回答した組織は 8% 弱でした。



ソフトウェアの出所を理解することは、リリースするソフトウェアの品質とセキュリティを確保するために不可欠であり、さまざまな政府規制で必須の要件になりつつあります。可視性なし、または不明と回答した約 8% の組織は、最低限、コードベースと外部パッケージのインベントリを作成して、ビルドのバージョンを追跡および割り当てる、自動化された CI/CD を実装していることを確認する必要があります。多くの組織はソース管理を当然のことと考えていますが、この約 8% の一部は、開発中にコードの変更を追跡する強力なソース管理ソリューションをまだ実装していない可能性があります。

コンプライアンスとガバナンスを目的として、ソフトウェアの作成とリリース プロセスで、ソフトウェアのテストと品質の基準を確保していることをどのように確認していますか? (委託調査、2023 年および 2024 年)

テストと品質の基準を確保するために使用している方法も、組織によって異なります。大多数 (70%) は自動化されたアプローチを活用していますが、約 3 分の 1 (29%) は依然として SDLC の各段階を手動で承認して進めています。



- SDLC 全体にわたり認証の証拠を自動的に収集しています。
- 継続的インテグレーション (CI) プロセスに自動化ゲートを組み込んでいます。
- SDLC の次の段階に進むソフトウェアを手動で承認しています。
- コンプライアンスとガバナンスの正式なプロセスはありません。

組織がセキュリティ対策にかけている時間と費用

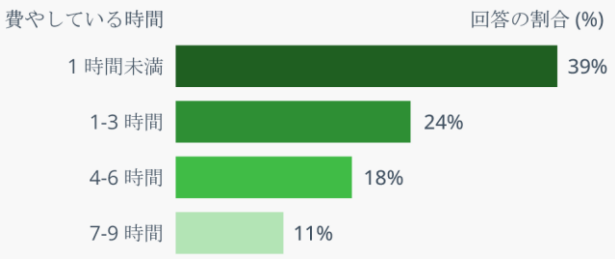
JFrog の委託により IDC が実施した調査によると、専門家の 60% が、開発者チームまたはセキュリティ チームがアプリケーションの脆弱性の対応に毎月 4 日以上費やしていると回答しています。セキュリティ関連のタスクに関わる開発者 1 人あたりの費用は年間平均約 28,000 ドルに上ります。これは財務的な影響だけでなく、開発者エクスペリエンス (DevEx) にも悪影響を及ぼします。

IDC: The Hidden Cost of DevSecOps (DevSecOps の隠れたコスト)

2024 年 9 月発行 | IDC #US52537524

開発者がセキュリティ問題に対応するために勤務時間外に費やされる時間

開発者は、セキュリティ問題に対応するため、勤務時間外に週約 3.6 時間を費やしています。これは、燃え尽き症候群に直結する環境を作り出しています。



~3.6

時間/週を開発者はセキュリティ問題に対応するため勤務時間外に費している

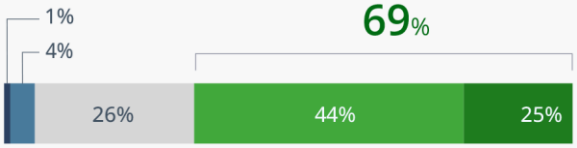
セキュリティ関連のアクティビティに費やされる費用

平均的な組織は、開発者 1 人あたり年間 28,000 ドルをセキュリティ関連のタスクに費やしています。DevSecOps はビジネスの必須事項であり、セキュアなアプリケーションの開発に不可欠ですが、非効率または適切に実装されていないツールやプロセスは、開発者の時間を浪費し、追加のビジネス コストを発生させます。

$28K

が開発者 1 人あたりの年間の費用としてセキュリティ関連のタスクに費やされている

コンテキストの切り替えが多すぎる



- 強く同意する - ツールや環境を非常に頻繁に切り替えています
- 同意する - ツールや環境を頻繁に切り替えています
- 未定
- 同意しない - ツールや環境をたまに切り替えています
- 強く同意しない - ツールや環境をほとんど、または全く切り替えていません

69% の開発者は、セキュリティ関連の責任を果たすために、コンテキストを頻繁に切り替える必要があることに同意しています。組織が DevEx の改善を目指す場合、ツールを頻繁に切り替えると、改善に支障をきたし、開発者がセキュリティ アクティビティに取り組む可能性が低くなることを考慮する必要があります。

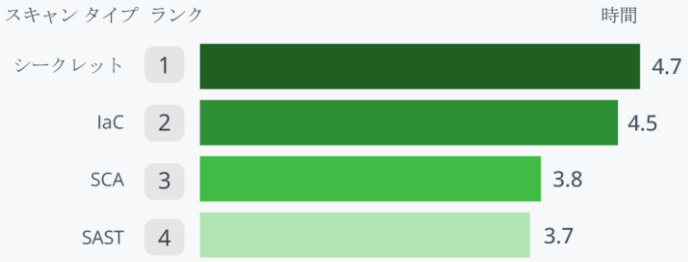
69%

の開発者は、セキュリティ関連の責任を果たすために、コンテキストを頻繁に切り替える必要があることに同意している

開発者が各スキャン タイプに費やしている時間

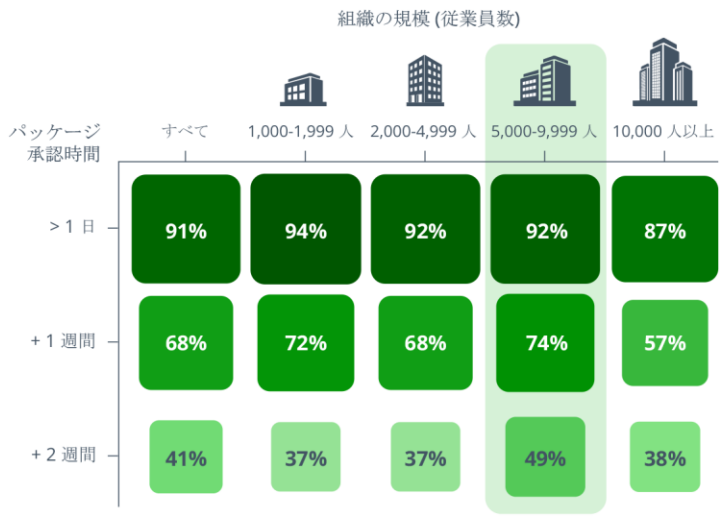
開発者はシークレットのスキャンに最も多くの時間を費やしています。これは、トークンとシークレットを処理するためのコーディング プラクティスにさらなるトレーニングが必要なこと、またはより効率的なシークレット処理ツールが必要なことを意味しています。コードにシークレットが残っていないか (分かりやすく例えると、住所が書かれたキー ホルダーに家の鍵を付けたまま落としていないか) 確認することも重要です。コードにシークレットが残っていると、攻撃者に重要なシステムやデータへの自由なアクセスを許してしまう可能性があります。

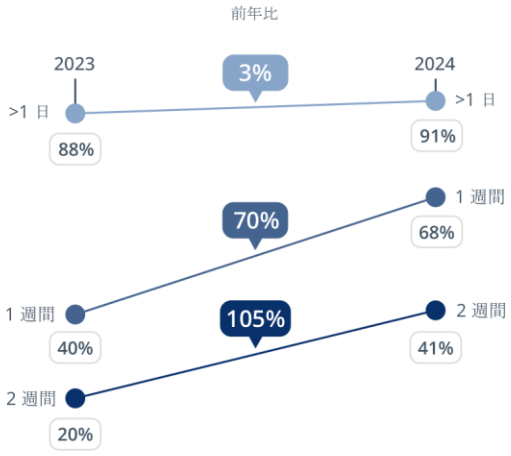
4.7
時間を開発者はシークレット スキャンに費やしている



IDC の調査は 2024 年 6 月にオンラインで実施され、DevSecOps を使用している米国および欧州の開発者、開発チーム リーダーとマネージャー、プロダクト オーナー 210 人から回答を得ました。この調査では、DevSecOps における開発者の時間のビジネスへの影響、開発者の時間を消費する DevSecOps ツールとタスク、DevSecOps における開発者の時間の価値、セキュリティ タスクが開発者のフローと満足度に与える影響に関する情報を求めました。

Q **新しいパッケージ/ライブラリの使用が承認されるまで、通常どのくらいの時間がかかりますか?** (委託調査、2023 年および 2024 年)





開発者が新しいパッケージの承認を待つ時間は、これまで以上に長くなっています。中規模の組織 (従業員数 5,000-10,000 人) は、規模に関係なく待機時間が長くなる傾向があり、92% が 1 日以上、74% が 1 週間以上、49% が 2 週間以上待っています。開発者がプロセスに積極的に関与するようになったのは心強いことですが、レビューやその他の手作業のために、依然として非効率であることは明らかです。新しいコンポーネントを導入するプロセスを真のセルフサービスにするには、さらなる取り組みが必要です。

重要なポイント

71

ソフトウェア サプライ チェーン セキュリティにおける基本的なプラクティスの欠如

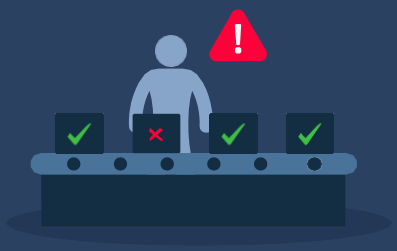
組織は、開発者やアプリケーションで参照される依存関係によりソフトウェア サプライ チェーンに流入する情報を管理するか、少なくとも強力な可視性を確保する必要があります。71% 以上の組織が開発者にインターネットから直接ダウンロードすることを許可している状況は懸念すべき事態であり、ソフトウェア サプライ チェーン セキュリティのベスト プラクティスに対する重大な違反行為です。パブリック レジストリをプロキシするアーティファクト管理ソリューションを、すべての組織に導入する必要があります。

スキャナーが増えると問題も増える?

組織はこれまで以上に多くのセキュリティ ツールを使用しているように見えますが、これはセキュリティ態勢にプラスの影響を与えているのでしょうか、それともマイナスの影響を与えているのでしょうか? いずれにせよ、カバレッジには依然としてギャップがあり、多くの組織がコード レベルとバイナリ レベルの両方でスキャンを行っていないという問題があります。

DevEx を損なわない DevSecOps

セキュリティ対策は、開発者の時間を毎週何時間も費やしています。組織は、アプリケーションのセキュリティ ステータスを損なうことなく、セキュリティ対策が開発者に与える影響を軽減する方法を模索すべきです。それには、スマートな優先順位付け、コンテキストに基づく結果、自動化が鍵となります。



アプリケーション管理のレベルアップ

組織の 85% は、社内でビルドしたアプリケーションの所有者を追跡していますが、アプリケーション標準を確保する方法は大きく異なっており、約 3 分の 1 の組織が、ソフトウェアをある段階から次の段階にプロモートするために手作業を活用しています。手作業は、意図的であれ偶発的であれ、潜在的なリスクを伴うため、組織にとって改善の余地があると言えます。

リスクの新たなフロンティア: AI と機械学習の開発

ほぼすべてのセキュリティ ツールと多くの開発ツールが、開発を高速化し、脆弱性の検出と修復を向上する AI 機能を謳い文句にしています。しかし、このセクションでは、AI ツールの利用ではなく、その**ビルド**に焦点を当てます。

人工知能/機械学習 (AI/ML) ソフトウェア サプライ チェーンは、組織にとって新たなリスクのフロンティアであり、成熟度曲線で従来のソフトウェア開発よりもはるかに左に位置しています。実際、JFrog が InformationWeek に委託した調査によると、79% の企業が、セキュリティ上の懸念により AI/ML 機能のソフトウェアへの利用や統合が遅れていると回答しています。

AI の導入と DevSecOps の傾向

InformationWeek の調査では、ソフトウェア開発者とサイバーセキュリティ チームが、アプリケーション セキュリティをソフトウェア開発ライフサイクルに統合することの重要性をどの程度理解しているか調査しました。また、チームがどのように悪意のあるコードから組織を保護し、AI テクノロジの不適切な利用を回避しているかについても調査しました。この調査から得られた主な洞察を以下に示します。

AI Adoption And DevSecOps: Staying Ahead While Staying Secure

2024 年 9 月発行 | InformationWeek & JFrog

企業全体の AI セキュリティへの信頼の欠如

79% の企業が、セキュリティ上の懸念により AI/ML 機能のソフトウェアへの使用や統合が遅れていると回答しています。

企業の AI セキュリティに関する上位 3 つの懸念事項は、LLM の使用によるデータ漏洩、AI モデルに含まれる悪意のあるコード、AI バイアスです。

64% の組織が、ソフトウェアでの AI の使用に関する新たな規制への準拠について、全く自信がない、または若干自信がある程度と回答しています。

AI サプライ チェーンの可視性は不明瞭

49% の企業は、アプリケーションでの ML モデルの使用を管理するための信頼できる方法を持っていません。

AI モデルを含むすべてのソフトウェア コンポーネントに対して、信頼できる 1 つの情報源を持っている組織は 4 分の 1 未満です。

3 分の 2 以上の組織は、ML モデルへの推移的な依存関係があるソフトウェア内のオープンソース パッケージを追跡するための信頼できる方法を持っていません。

AI ポリシーは依然として不足している

58% の企業は、開発者がオープンソースの AI モデルやコンポーネントを使用する方法に関するルールを定めるポリシーを全く導入していないか、導入しているかどうかを把握していません。

60% の企業は、開発者がトレーニング データをどのように調達またはライセンスするかに関するポリシーを持っていません。

施行に一貫性がない

68% の回答者が、AI コンポーネントの使用を強制するための方法がないか、手動レビューに依存していると回答しています。

59% の回答者が、トレーニング データに関するポリシーを強制するための方法がないか、手動レビューに依存していると回答しています。

InformationWeek に委託したこの調査は、2024 年 5 月にオンラインで実施され、主に北米に拠点を置く 210 人の IT およびサイバー セキュリティの専門家から回答を得ました。回答者は、あらゆる規模の企業に所属している幹部から一般社員までで、コンサルティング、銀行と金融サービス、教育、政府機関、テクノロジ、ヘルスケア、製造業など、21 以上の垂直産業が対象となっています。

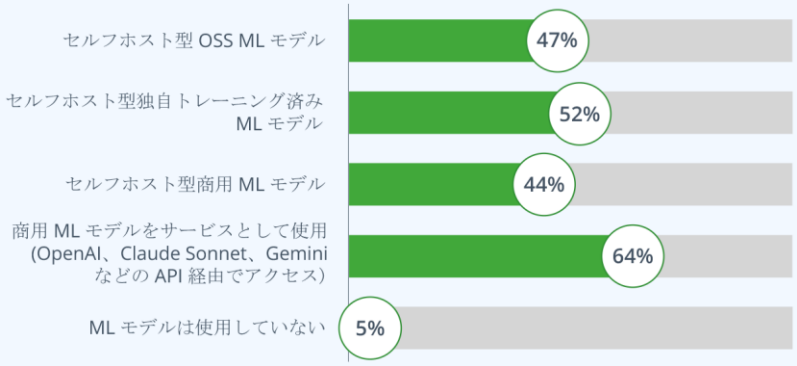
InformationWeek の調査では興味深い傾向が明らかになりました。このセクションの残りの部分では、組織が実際にどのように AI サービスをアプリケーションに導入し、その利用を管理しているかについて、より深く掘り下げます。

ML モデル アーティファクトの使用、ガバナンス、スキャン

開発しているアプリケーションの一部として ML モデルを利用する主な方法は?
(委託調査、2024 年)

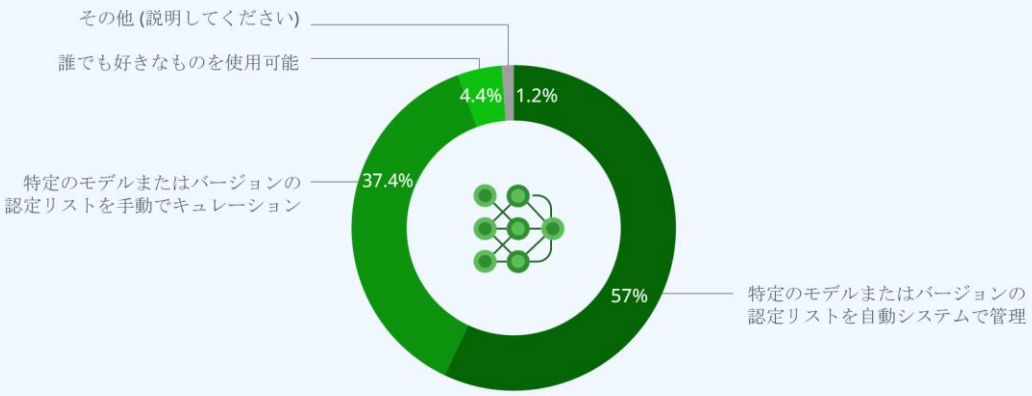
組織が AI サービスやアプリケーションを導入する方法はさまざまであり、調査によると、組織は複数の方法を同時に使用していることが示されています。

最も人気の高いアプローチは、API 経由でアクセスする商用モデルの使用です (64%)。



この方法では、組織は初期開発コストやインフラストラクチャ コストをかけることなく、強力で汎用的な AI 機能を迅速に利用できます。しかし、セルフホスティング モデルへの投資も見受けられます。半分以上は、特定のビジネス ニーズに合わせて作成された独自モデルをセルフホスティングしています。回答者の約 2 人に 1 人が、機械学習モデルを利用する主な方法としてセルフホスト型 OSS モデルを挙げています。

開発組織内で ML モデル アーティファクトの使用をどのように管理していますか? (委託調査、2024 年)



専門家の 3 人に 1 人以上 (37%) が、手動でキュレーションした特定のモデルまたはバージョンのリストを使用して ML モデル アーティファクトの使用を管理していると回答しています。

InformationWeek の調査によると、49% の企業は、アプリケーションでの ML モデルの使用を管理するための信頼できる方法を持っていません。これが、4% の回答者が開発者による ML モデル アーティファクトの使用の管理 (手動を含む) を意図的に行っていない理由と言えるでしょう。

調査対象者全体の

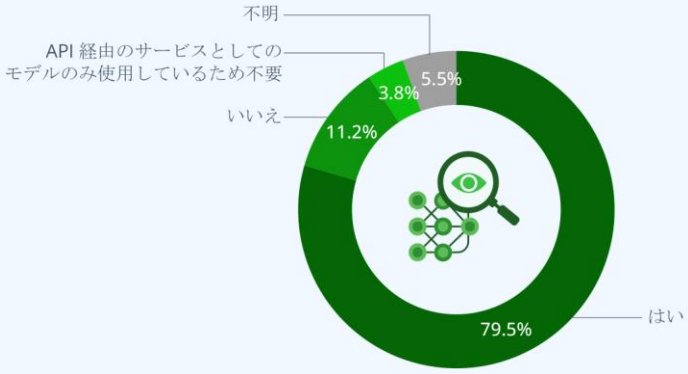
16%

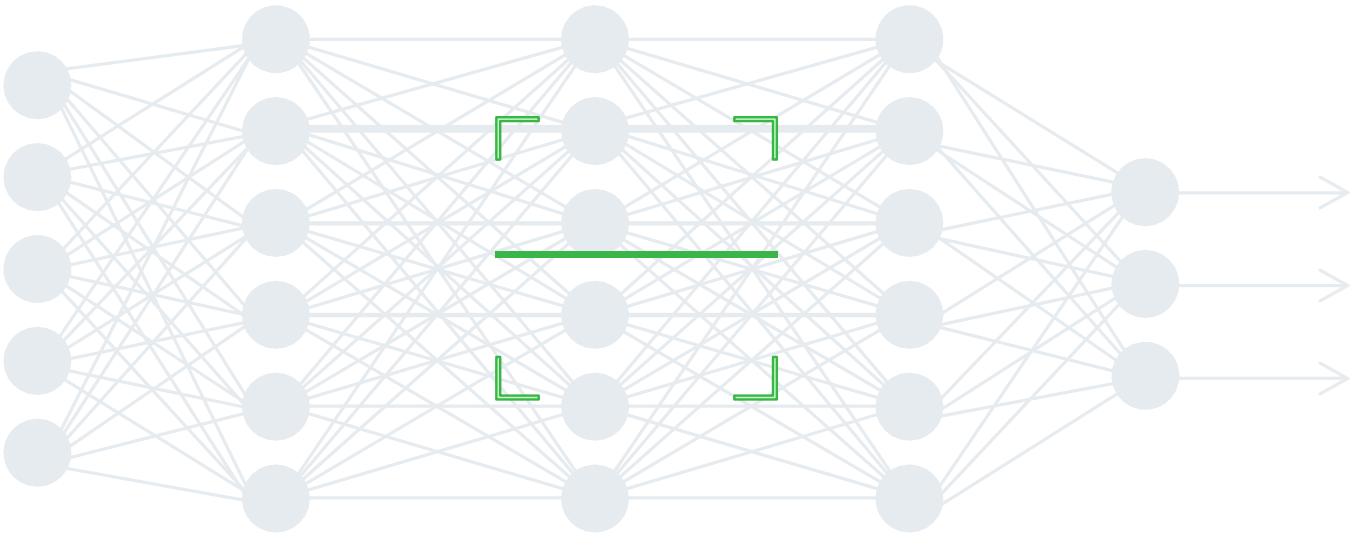
がセルフホスト型 OSS モデルを利用し、モデル アーティファクトの使用を手動で管理しています。

あなたの組織では、ML モデル アーティファクトにセキュリティの脆弱性や悪意のあるモデルが含まれていないかスキャンする方法を導入していますか? (委託調査、2024 年)

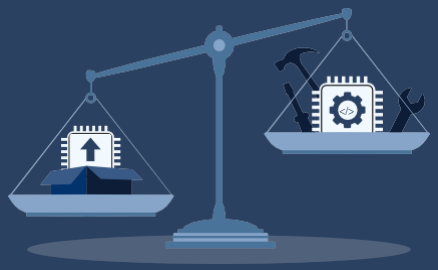
大多数の組織 (79%) が、ML モデル アーティファクトにセキュリティの脆弱性や悪意のあるコードが含まれていないかスキャンする方法を導入していると回答しています。11% は導入していません。

幸いなことに、セルフホスト型の OSS モデルを利用していて、脆弱性や悪意のあるモデルを防ぐためにスキャンする方法を導入していないと回答したのは調査対象者全体のわずか 3% でした。しかし、AI/ML 開発を安全に行うための適切なツールとポリシーを導入するには、組織とセキュリティ業界レベルの両方で、さらなる取り組みが必要です。





重要なポイント

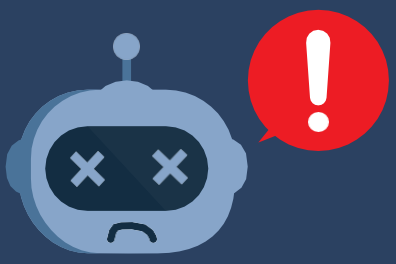


商用モデルによる AI の導入

商用モデルの利用は、AI サービスをビジネス アプリケーションに導入するための一般的な手段です。API 経由で商用モデルにアクセスすることにより、組織は自社モデルのビルドと管理に必要なツール、リソース、専門知識の調達にかかる時間と費用を節約できます。あまり AI/ML の経験がない組織は、この分野でより専門知識を備えたプロバイダーにモデルのセキュリティを委託するほうが賢明でしょう。

AI サプライ チェーンの可視性は不明瞭

多くの組織は、アプリケーションでの機械学習モデルの使用を管理するための信頼できる方法を確立するのに苦労しており、ML モデルを含むすべてのソフトウェア コンポーネントに関して信頼できる 1 つの情報源を持っていないことがよくあります。さらに、ML モデルに関連する推移的な依存関係を含むオープンソース パッケージを効果的に追跡する能力には、大きな盲点が存在します。ML ソフトウェア開発プロセスにこれらの重大なギャップが存在すると、組織が AI/ML サプライ チェーンを効率良く管理することが困難になるだけでなく、セキュリティの脆弱性のリスクも高くなります。



AI セキュリティへの過信

79% の組織が何らかのレベルのモデル スキャンを適用しているとレポートしている一方で、AI/ML モデル セキュリティの現状のソリューションは、単純な検出手法を用いた初期段階にあります。たとえば、現在のアプローチでは Hugging Face での悪意のあるモデルの特定で 96% の誤検知率が発生すると同時に、単純なスキャン手法のために脅威を見逃していました。多くのセキュリティ ツールがモデル アーティファクトを利用しようとしていますが、組織はセキュリティ プロバイダーのモデル セキュリティ ソリューションの効果を真剣に評価する必要があります。

調査方法

このレポートは、JFrog 使用状況データ、JFrog セキュリティ リサーチ チームによる CVE 分析の結果、委託した第三者機関の調査データから得られた洞察を組み合わせて作成したものです。各情報源の詳細を以下に示します。

JFrog Platform 使用状況データ

このレポートで取り上げているテクノロジ利用傾向は、JFrog Platform for Cloud の匿名化された使用状況データの年末のスナップショットに基づいており、数千のお客様、数十万のリポジトリ、ペタバイト規模のデータを表しています。

パッケージの人気度は、特定のパッケージ タイプにおけるアクション数 (アップロード/ダウンロード)、アーティファクトの総数、リポジトリの総数、アーティファクトの総サイズで表されます。アクション数は、開発者がさまざまなパッケージ タイプを呼び出して生成する頻度を適切に表しており、ソフトウェア開発における実際の使用状況を示す指標となります。

一部の企業により、これらのランキングが歪曲されている可能性はありますが、アーティファクトのアクションも調査しているため、開発プロセスで積極的に使用されているパッケージ タイプを安全に判断できます。

JFrog セキュリティ リサーチ チームによる分析

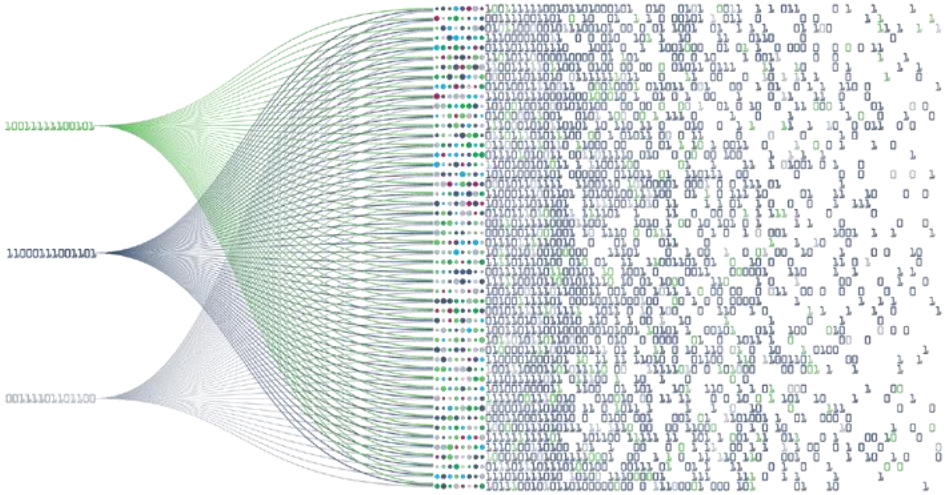
指定された CNA として、JFrog セキュリティ リサーチ チームは、新たな脆弱性を定期的に監視および調査し、その真の重大度を理解して、コミュニティとすべての JFrog のお客様に情報を公開しています。

このレポートには、JFrog Catalog サービスによりパブリック ソースから取得したデータ、NVD から取得した CVE 情報、これらのデータ ソースに基づいて JFrog セキュリティ リサーチ チームが行った独自の分析が含まれています。

委託調査の結果

JFrog は Atomik Research に委託して、米国 (n=375)、英国 (n=205)、インド (n=206)、ドイツ (n=205)、フランス (n=205)、イスラエル (n=205) の特定の業界[1] に勤務する 1,402 人を対象に、国際的なオンライン調査を実施しました。サンプルは、組織の情報テクノロジ、情報システム、テクノロジ部門で特定の職務[2] を担当する正社員で構成されています。さらに、回答者全員が総従業員数 1,000 人以上の組織に所属しており、組織内にチーム メンバーが少なくとも 50 人いるソフトウェア開発チームが存在することを確認しています。オンライン アンケートのすべての回答者は、英語、フランス語、ドイツ語、ヘブライ語、ヒンディー語の翻訳版にアクセスできました。

サンプル全体の誤差は +/- 3 パーセント ポイント、信頼度は 95 パーセントです。フィールドワークは 2024 年 11 月 22 日から 12 月 9 日まで実施されました。Atomik Research はクリエイティブな市場調査会社です。



1. 参加資格を得るには、すべての回答者は、次の業界にサービスを提供している組織に雇用されていることを示す必要があります。(a.) 航空宇宙 (b.) 建築およびエンジニアリング (c.) 自動車 (d.) 銀行、金融サービス、保険、フィンテック (e.) エネルギー、石油、ガス (f.) 政府または公共部門 (g.) ヘルスケアおよびライフサイエンス (h.) ホスピタリティ (i.) 製造 (j.) 小売 (k.) テクノロジ (l.) 運輸および物流 (m.) 公益事業、通信、電力。

2. 参加資格を得るには、すべての回答者は、次の職務または類似する職務に就いていることを示す必要があります。(a.) AI スペシャリストまたは AI エンジニア (b.) アプリケーション セキュリティ エンジニア (d.) サイバー セキュリティ エンジニア (e.) データ サイエンティスト (f.) 開発者 (g.) DevOps アーキテクト (h.) DevOps エンジニア (i.) エンジニアリング マネージャー (j.) 機械学習スペシャリストまたは ML エンジニア (k.) プラットフォーム エンジニア (l.) セキュリティ アーキテクト (m.) セキュリティ研究者 (n.) サイト信頼性エンジニア(o.) ソフトウェア アーキテクト (p.) ソフトウェア開発者 (q.) ソフトウェア エンジニア (r.) ソリューション アーキテクト。これに加えて、組織の情報テクノロジ、情報システム、テクノロジ部門、IT 製品開発部門での雇用を示すこともできます。

JFrog Platform について

JFrog Platform は、ソフトウェア サプライ チェーンのパッケージ テクノロジやツールと統合する、拡張性に優れたオープンなクラウドネイティブのソリューションです。開発者から、機械学習モデル、エッジ デバイス、本番環境データ センターを含む、あらゆるデプロイ環境へのソフトウェア コンポーネント フローとして、組織に完全な制御とトレーサビリティを提供します。

このデータ レポートには、米国連邦証券法の定義による「将来の見通し」に関する記述が含まれています。JFrog 使用状況データやソフトウェア サプライ チェーンに関する記述を含みますが、これらに限定されません。

これらの将来の見通しに関する記述は、JFrog の現在の仮定、期待、見解に基づくものであり、JFrog の実際の結果、業績、成果が将来の見通しに関する記述で明示または暗示されている内容と大きく異なる可能性のある、重大なリスク、不確実性、仮定、状況の変化の影響を受けます。

実際の結果、業績、成果がこのプレス リリースに記載された記述と大きく異なる可能性のある要因は多数存在します。2024 年 12 月 31 日を期末とする Form 10-K による年次レポート、Form 10-Q による四半期レポート、JFrog が証券取引委員会に随時提出するその他の提出書類やレポートを含む、証券取引委員会への提出書類に記載されているリスクを含みますが、これらに限定されません。将来の見通しに関する記述は、このプレス リリースの日付時点の JFrog の見解と仮定を表したものであり、JFrog は将来の見通しに関する記述を更新する義務を負いません。