Akamai

ホワイトペーパー

API 攻撃の分析

BOLA とインベントリー管理の脆弱性を理解する

はじめに

ほとんどのセキュリティチームは、特にアプリケーション・プログラミング・インターフェース（API）に関して、プロアクティブな脅威ハンティングが効果的なエンタープライズ・セキュリティ・プログラムの重要な要素であることを理解しています。API は多くの場合、データ、機能、ワークフローへの直接アクセスを提供します。また、ベースラインの境界セキュリティ対策がアプリケーションの保護に広く使用されている一方で、API の悪用やその他のタイプの攻撃が増加しています。実際、近年の注目すべきセキュリティインシデントの中には、API 関連のものもあります。そのような攻撃プロファイル（オブジェクトレベルの認可の不備（BOLA）や不適切なインベントリー管理の脆弱性など）への理解を深めるために、このホワイトペーパーでは次のことを行います。

- API の基本を確認する
- API セキュリティの重要性が高まっている理由を説明する
- いくつかの注目すべき API セキュリティインシデントを取り上げて、API セキュリティの重要領域を明らかにする
- API 脅威ハンティングを効果的に実行するために必要な機能について説明する

API とエンドポイントの基本

まず、基本的な用語を確認しましょう。API は、B2C 機能、B2B コラボレーションや統合、社内開発機能や統合機能など、さまざまな目的で使用されます。Web API は、Web ブラウザーで使用されるのと同じ HTTP プロトコルを介して通信するものであり、最も一般的な実装モデルです。この API が提供する特定の機能は、サービスまたは API 製品と呼ばれることもあります。

API セキュリティについて考える際には、エンドポイントの概念を理解することも重要です。この用語はエンドユーザー・コンピューティング・デバイスを指して使用されることもありますが、API の文脈では意味が異なります。API エンドポイントは、API の一部である単一のアクセス可能なリソース、または API エンドポイントで実行できる操作と考えることができます。

> **簡単な例を考えてみましょう。特定の会社の注文情報を返す API エンドポイントを GET /orders/{orderID} と表すとします。この場合、GET は特定の HTTP メソッドであり、orders と orderID は API を介して要求される特定のリソースを表します。**

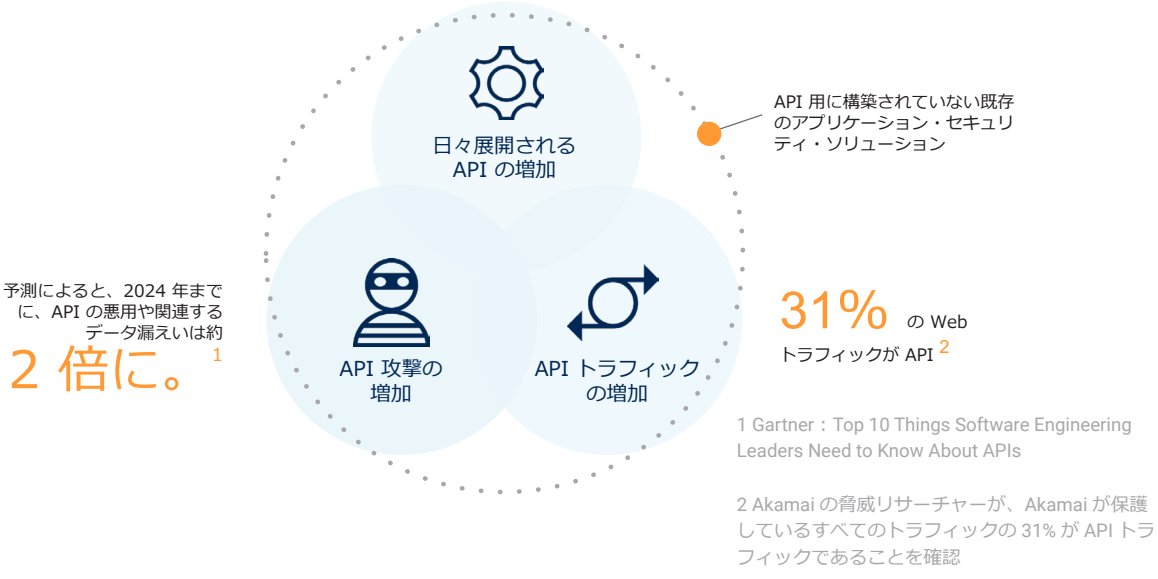
API が新しい大きなセキュリティ課題である理由

これまで、攻撃者はエンタープライズのデータセンターを侵害して、特定のサーバーにある組織のデータにアクセスし、それを窃取することを目標にしていました。または、エンタープライズのネットワークトラフィックを調べて機微な情報を取得しようとした場合もあります。このようなシナリオでは、プロアクティブな脅威ハンティングの重点は、攻撃者の潜在的な侵入ポイントを排除するための侵入テストなどのアクティビティに置かれます。

API 対応の世界では、このダイナミクスは異なります。多くの API は本質的に外部の誰でもアクセスでき、認証情報とキーが唯一の防衛線となることもあります。また、攻撃者はますます巧みにこれらの要素を侵害するようになっています。さらに、最も危険なタイプの API の悪用は、API へのアクセスが許可されているにもかかわらず承認されていない方法で API を使用することを選択した関係者が原因で、発生する場合があります。

実際の API 攻撃

Akamai が保護するすべてのトラフィックの 31% が API トラフィックです。このように API トラフィックが増加すると、攻撃や悪用の増加など、下流の影響が生じます。Gartner は、2024 年に API の悪用とデータ漏えいが倍増すると予測しています。しかし、多くのセキュリティチームは遅れを取り戻せずにいます。API は増加し続けていますが、既存のアプリケーション・セキュリティ・ツールによる API 保護は非常に限定的です。



1 Gartner：Top 10 Things Software Engineering Leaders Need to Know About APIs

2 Akamai の脅威リサーチャーが、Akamai が保護しているすべてのトラフィックの 31% が API トラフィックであることを確認

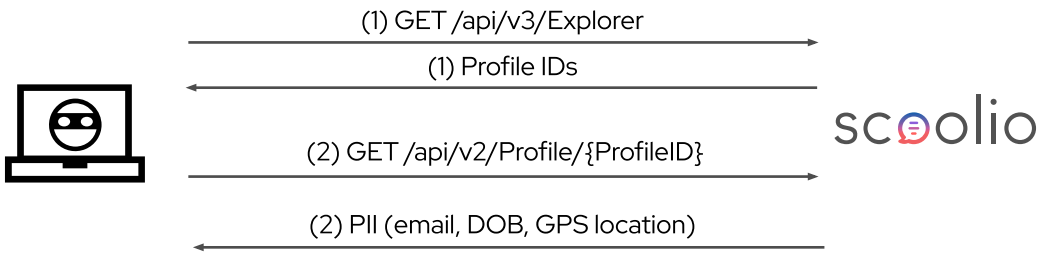
この問題を実感するために、API 攻撃が企業や顧客に与え得る実際の影響を示すケーススタディを見てみましょう。

ケーススタディ

アカウントの乗っ取り | Scoolio

注目すべき事例は、ドイツの教育アプリ Scoolio に影響を与えた 2021 年のインシデントです。このアプリは学生ユーザーから幅広い情報を収集します。たとえば、性格診断テストを実施したり、ソーシャルネットワーキング機能やチャット機能を提供したり、学習計画や個人指導などの活動を管理したりします。これらの機能により、大量の個人情報（PII）が蓄積されます。セキュリティ研究者の Lilith Wittmann 氏は、この教育アプリの API に BOLA の脆弱性を発見しました。この脆弱性により、教育アプリの他のユーザーが 2 つの API 呼び出しを利用して PII やその他のデータにアクセスできるようになっていました。

仕組みは次のとおりです。



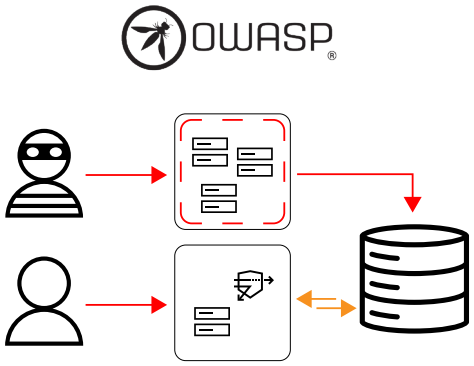
ステップ 1
GET /api/v3/Explorer API 呼び出しを送信します。
この呼び出しに対して、UUID が返されました。この実装では UUID を ProfileID と呼んでいました。

ステップ 2
GET /api/v2/Profile/{ProfileID} API 呼び出しを送信します。
このリクエストに対して、関連ユーザーの広範な PII（電子メール、生年月日、GPS 位置情報など）が返されました。

UUID を使用することの価値

どちらのシナリオも UUID の使用に重点を置いていますが、実際のところ UUID を使用するのは非常に良い方法です。予測可能な配列のユーザー識別子ではなくランダムに生成された番号を使用すると、攻撃者が一斉にユーザーの情報にアクセスするのが難しくなります。この問題は、UUID 情報が過度に公開され、BOLA の脆弱性と組み合わされた場合に発生します。

不適切なインベントリー管理



この API の脆弱性の悪用のもう 1 つの側面は不適切なインベントリー管理であり、これは OWASP API Top 10 リストの 9 番目に挙げられています。攻撃シーケンスを詳しく見てみると、1 つ目のステップはバージョン 3 の API に適用され、2 つ目のステップはバージョン 2 に適用されていることがわかります。バージョン 3 では、PII へのアクセスがより厳格に管理されるよう改善が行われました。しかし、より脆弱なバージョン 2 をすべてのユーザーが利用できるままだったため、この改善は台無しになりました。最終的に、バージョン 2 とバージョン 3 の両方が BOLA の脆弱性の影響を受けました。しかし、不要であるはずのバージョン 2 が存在したことにより、この脆弱性の影響はさらに深刻になりました。

API を保護するために組織が現在実施している対策

多くの組織は、次の 3 つの柱に重点を置いて API セキュリティにアプローチしています。

1. 集約型認可 — まず、すべての API アクセスポートに集約型の認可エンジンを実装することで、認可メカニズムの欠陥につながる開発ミスを防止し、API の脆弱性リスクを軽減できます。

2. API テスト — 2 つ目の重要な対策は API テストです。静的コード分析と動的テストの両方を使用して、すべての脆弱性（特に不備のある認可）をテストすることにより、開発プロセスの初期段階で問題が表面化します。

3. ランタイム保護 — 第 3 の基本的な柱は、本番環境に対する一連のランタイム保護です。極めてプロアクティブなチームでも、展開前にすべての脆弱性を検出することはできません。そのため、本番データへのユーザーアクセスを検査し、既知のカテゴリーの脆弱性の悪用を可能な限り防止することが不可欠です。

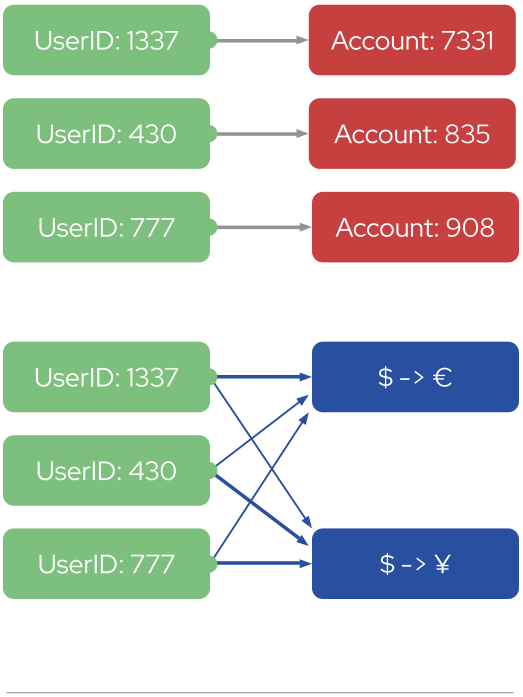
これらの 3 つの対策は、API セキュリティ戦略の優れた基盤となります。しかし、これらは完璧でも包括的でもないということも、覚えておいてください。たとえば、集約型認可を取り入れた組織であっても、開発者が常にベストプラクティスに従うという保証はありません。最後に、既存のアプリケーション保護ツールは、既知の攻撃パターンを検知するためには適していますが、BOLA のようなニュアンスの異なる脅威の検知には長けていないことが多いのです。

この基盤の上により高度な BOLA 検知手法を取り入れる方法

BOLA やその他の微妙な API の脆弱性を検知して緩和するための鍵の 1 つは、API アクティビティに関与するエンティティ間の関係をモデル化することです。これには、リソース自体だけでなく、リソースにアクセスしようとするアクター（ユーザーなど）も含まれます。API とやり取りするアクターエンティティやビジネス・プロセス・エンティティのつながりをマッピングできれば、他の点では同一である API イベントを分析する際に、正当なアクティビティと不正アクティビティを区別できるようになります。

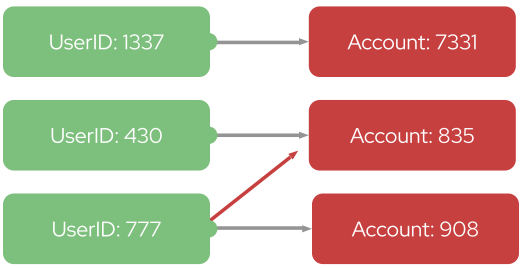
関係マッピングの例

関係マッピングをよりよく理解するために、この基本的な例について考えてみましょう。銀行業務アプリケーションは 2 つのアクションをサポートします。1 つ目のアクションは、口座データ（口座残高や最近の取引などの情報）を読み取ることです。2 つ目のアクションは、為替レートを表示することです。これらの例では、ユーザーとリソースの関係が大きく異なります。口座情報へのアクセスは 1 人のユーザーに限定する必要があります。一方、為替レート機能はすべてのユーザーが一般的に利用できる必要があります。



これは非常に基本的な例であり、BOLA を防止または検知するためにはエンティティ間の関係マッピングの高度なモデルを構築するのが現実的です。

この例では、自身が所有していないアカウントにアクセスしようとしているユーザーがいます。この API 呼び出しは同じように見えるかもしれませんが、エンティティマッピングによって提供される追加コンテキストにより、許可されていない API 呼び出しであることが明確になります。



実際の高度な BOLA 攻撃検知

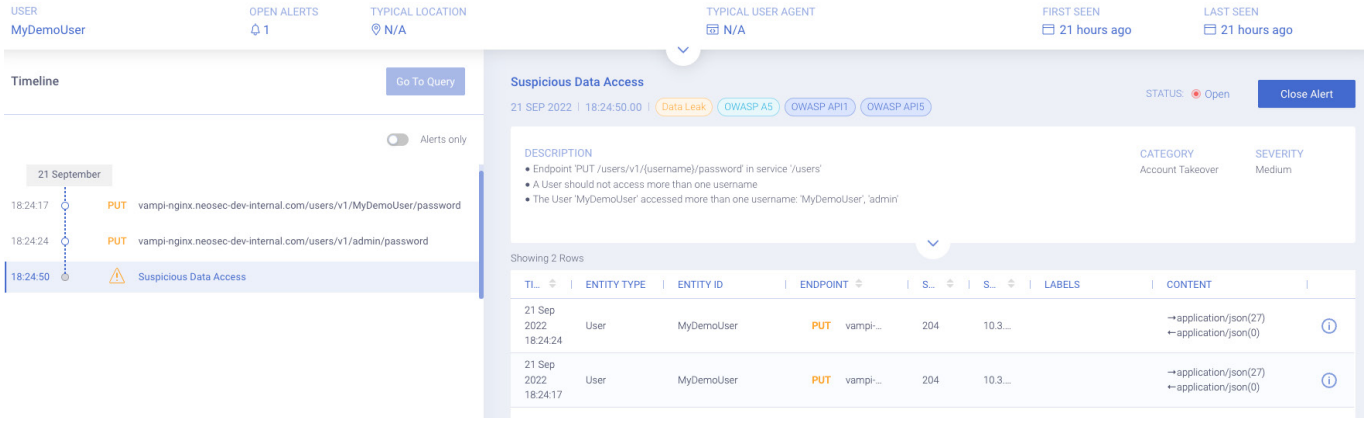
次に、前述のケーススタディの脆弱性のような複雑な例にこのコンセプトを適用してみましょう。以下は、このシナリオに関係するエンティティのスニペットです。

scoolio

GET/api/v3/Profile/{ProfileID}

Headers:
- Authorization: <MyAccessToken>

緑色でハイライトされているのがアクターエンティティ、赤色でハイライトされているのが要求されたリソース（プロファイル ID）です。これらの関係を理解すれば、アクターによるアクセスを必要に応じて 1 つのリソースに制限するなどの一般的なロジックを適用するための手順を実行できます。これは単純なことではありません。なぜなら、関係がこれより複雑で、一対多の関係が含まれる場合があるからです。しかし、機械学習やふるまい分析などの手法を利用すれば可能になります。たとえば、あるお客様に対する BOLA の脆弱性の悪用が正常に検知された場合、次のようになります。



この例では、ラボ環境で BOLA の脆弱性がシミュレーションされました。弊社のプラットフォームは、エンティティマッピングとふるまい分析によって BOLA を検知し、情報豊富なアラートを生成しました。このアラートを閲覧するセキュリティアナリストや脅威ハンターは、MyDemoUser が自分のユーザープロファイルにアクセスして、パスワードを変更したことを確認します。これは、許可されているアクションです。しかし、その後すぐに、管理者パスワードを変更するために別の API 呼び出しが実行されたことがわかります。これは明らかに、アクターとリソースの関係に基づいて許可されていない行為であるため、アラートが生成されました。

API セキュリティに取り組むための出発点

ほとんどの組織にとって、API セキュリティは継続的に行われる作業です。そのため、どこから始めればよいかわからなくなる場合があります。上記の 3 つの基本的な柱は出発点として有用ですが、導入時に次の 3 つの推奨事項に従えば、アプローチの効果は大幅に強化されます。

1. 常に最新の API インベントリーを確保する
2. 非本番 API 環境と本番 API 環境を監視する
3. エンティティ間の関係を強化する

把握できない API は保護できません。そのため、API の効果的な保護は、最新の API インベントリーとセキュリティ体制評価から始まります。同様に、API セキュリティ監視機能を開発する際には、本番 API 環境と非本番 API 環境の両方に実装することが重要です。そして最も重要なことは、API の監視と適用は、アクションに対して実行するだけではなく、API アクティビティに関係するエンティティ間の関係も考慮する必要があるということです。これにより、脆弱性と保護のギャップを見つけ、意図した API 使用モデルとのコンプライアンスを実施することができます。API 内のふるまいを理解することで、あらゆる不正行為を見抜けるようになります。

API 攻撃とその防御方法についてさらに理解を深めたい場合は、Akamai による OWASP API Top 10 の分析結果をご確認ください。