
F5 LABS
**2018**

# アプリケーション プロテクション レポート

## 著者について

**レイ・ポンポン(Ray Pompon)氏** は、F5 Labsの脅威リサーチ主任エバンジェリストです。レイは、インターネット セキュリティにおける 20 年以上の経験があり、連邦法執行機関のサイバー犯罪調査に密接に協力しています。これまで、FBI の Flyhook おとり捜査や NW Hospital ボットネットの起訴など、いくつかの大規模な侵入事例に直接協力しています。また、『*IT Security Risk Control Management: An Audit Preparation Plan*』(Apress出版)の著者でもあります。

## 貢献者

**デビー・ウォルコフスキー (Debbie Walkowski)**
*F5、脅威リサーチ エバンジェリスト*

**デイビッド・ホームズ (David Holmes)**
*F5、脅威リサーチ主任エバンジェリスト*

**サラ・ボディ (Sara Boddy)**
*F5 Labs、ディレクター*

**ジャスティン・シャタック (Justin Shattuck)**
*F5、脅威リサーチ主任エバンジェリスト*

## ビジネスおよびデータ パートナ

**LORYKA** は、新しい攻撃、持続的標的型攻撃、および関連する組織や個人を監視および調査する専門研究者によるチームで、進行中の攻撃および新しい脅威の特定、調査、追跡を行うリサーチ ツールの開発も行っています。私たちは、LORYKA と協力して、2017 年に 21,010 のネットワークで発生した WEB 攻撃から収集した世界中の侵入およびハニーポット データを分析しました。

**PONEMON INSTITUTE** は、プライバシー、データ保護および情報セキュリティ ポリシーを独自に研究しています。この報告書では、F5 の委託で Ponemon により実施されたセキュリティ専門家に関する 2 つの調査から得られた結果を参照しています。

**WHATCOM COMMUNITY COLLEGE CYBERSECURITY CENTER :** WCC コンピュータ情報システム学部教員の SEAVER MILNOR 氏と CHRISTY SAUNDERS 氏に特別な感謝を送ります。両氏には、カリフォルニア州、ワシントン州、アイダホ州およびオレゴン州の法務長官事務所に提出された情報漏洩通知記録を包括的に検証、分析および分類していただきました。

**WHITEHAT SECURITY** は、アプリケーション プロテクションにより安全なデジタル体験を保証できるように企業を支えています。この報告書では、私たちは、WhiteHat 脆弱性データに関する上記の資料から収集されたデータのほとんどを相互参照しています。

**UNIVERSITY OF WASHINGTON TACOMA** 大学院生の方々には、Masters of Cybersecurity and Leadership Capstone 研究プロジェクトの一環として背景調査およびデータソース分析にご協力いただきました。

F5 LABS
**2018**

# アプリケーション プロテクション レポート

# 目次

# 目次


01

# エクゼクティブサマリ

Webアプリケーションを構成する独立した多数のコンポーネントは、運用要件およびサポート インフラストラクチャ（クラウドとオンプレミスの両方）が異なる別々の環境で稼働し、それぞれがネットワークでつながっています。アプリケーション サービス、アプリケーション アクセス、Transport Layer Security（TLS）、Domain Name Service（DNS）およびネットワークが相互作用するこのひと続きの層は、各層が潜在的な攻撃対象となるため、この報告書ではこれらの層について調べます。

アプリケーションがどのように攻撃を受けているか客観的に捉えるため、F5 Labs は、独自の内部データ、WhiteHat Security の脆弱性データ、Loryka の攻撃データおよび F5 の委託により Ponemon が実施した IT 専門家に関するセキュリティ調査など、さまざまなソースのデータを検証しました。

また、Whatcom Community College のサイバーセキュリティ センタと協力して、カリフォルニア州、ワシントン州、アイダホ州およびオレゴン州における情報漏洩通知記録を包括的に検証しました（米国では、各州の司法長官が、その州の情報漏洩事例に関する情報公開法および通知を監視および施行します。この役割において、一部の州では、情報漏洩通知書を公開しています）。

これら 4 つの州に関して、2017 年および 2018 年 Q1 に発生した 301 件の情報漏洩を分析した結果、報告された全情報漏洩の原因として、Web アプリケーション攻撃が 30% と最も多いことが分かりました。これより前に F5 Labs が実施した過去 12 年、26 か国における 433 件の大規模な情報漏洩事例の研究では、その 53% でアプリケーションが初期標的にされていたことが分かっています。

**アプリケーションの保護は、これまでもそしてこれからも重要です。しかし、CISO は今、何を知っておくべきでしょうか。**

大多数の組織は、組織内のすべてのアプリケーションを把握できていることにあまり自信がありません。

## どんなアプリケーションをどこで使用しているか

F5 Ponemon 調査「Web Application Security in the Changing Risk Landscape: Global Study」では、大多数の組織は、組織内のすべてのアプリケーションを把握できていることにあまり自信がないことが分かりました。回答者の 38% は、組織内のすべてのアプリケーションが使用されている場所を把握していることに「自信がない」と答えています。その一方で、Web アプリケーションの 34% はミッションクリティカルだと考えています。最も一般的に使用されている Web アプリケーションは、バックアップとストレージ（83%）、電子メールなどの通信アプリケーション（71%）、文書管理と連携（66%）および Microsoft Office スイートのアプリケーション（65%）でした。

## アプリケーション攻撃が組織に与える影響

アプリケーションが攻撃されると、多種多様な影響を受けます。サービス拒否は組織に最も被害を与える攻撃に挙げられ、回答者の 81% がアクセスの損失を 1 ～ 10 の 10 点満点で 7 点に評価しました。機密または秘密情報（知的財産権や企業秘密など）の情報漏洩がこれに続き、調査回答者の 77% が 7 ～ 10 点に評価しました。同様に、回答者の 73% は、アプリケーションの改竄を 7 ～ 10 点に評価しました。さらに、回答者の 64% は、顧客、消費者または従業員に関する個人を特定できる情報（PII）の損失を 7 ～ 10 に評価しました。

## 重大なリスク

州の司法長官が公開した 2017 年および 2018 年 Q1 の情報漏洩通知書の範囲内で、Web 攻撃について詳しく調べました。具体的なアプリケーション情報漏洩には、Web インジェクションによるペイメント カードの盗難（70%）、Web サイトのハッキング（26%）およびアプリケーション データベースのハッキング（4%）がありました。このデータと、関連する WhiteHat Security の脆弱性データ、Loryka の攻撃監視および Exploit-DB が公開する既知のエクスプロイト（CVE 準拠の公開エクスプロイト[1] および対応する脆弱性ソフトウェア[2] のアーカイブ）を相互参照して、最も重大な新しいリスクについて調べました。

**70%**

2018 年 Q1 で最も多かった情報漏洩報告は、ペイメント カード情報を盗む WEB インジェクションでした。

### アプリケーション サービスに対するインジェクション攻撃

2018 年 Q1 で最も多かった（70%）情報漏洩報告は、顧客のペイメント カード情報を盗む Web インジェクションでした。インジェクション攻撃では、攻撃者は、実行中のアプリケーションにコマンドまたは新しいコードを直接挿入して（またはアプリケーションを改竄して）、悪意のある計画を成功できます。過去 10 年の間、情報漏洩記録の 23% には、インジェクション攻撃で最も悪名が高い SQL インジェクション攻撃が関与していました。インジェクション脆弱性（まだ悪用されていない弱点）も同様によく見られます。WhiteHat Security の報告によると、インジェクション脆弱性は、2017 年に発見されたすべての脆弱性の 17% を占めていました。この問題は非常に重大で、インジェクション脆弱性は OWASP Top 10 2017 リストでアプリケーションの最大のリスクに評価されています。このため、インジェクション脆弱性を検出、パッチ処理および阻止することを優先して行ってください。

### アカウント アクセスのハイジャック

情報漏洩記録分析によると、2017 年および 2018 年 Q1 に発生したすべての Web アプリケーション情報漏洩の 13% はアクセス関連でした。これらの内訳は、不正電子メールを介した認証情報詐取（34.29%）、不適切なアクセス制御設定（22.86%）、ブルートフォース（総当たり）攻撃によるパスワードの解読（5.71%）、盗んだパスワードによるクレデンシャル スタッフィング（8.57%）およびソーシャル エンジニアリングによる詐取（2.76%）です。Web アプリケーション Exploit-DB スクリプトの約 25% もアクセス関連でした。F5 Ponemon セキュリティ調査によると、回答者の 75% は、重要な Web アプリケーションのアプリケーション認証にユーザ名とパスワードのみを使用しています。重要なアプリケーションでは、フェデレーション アイデンティティや多要素認証などのより強力な認証ソリューションを検討してください。完全には制御できない外部アプリケーションについては、Cloud Access Security Broker（CASB）を使用して認証を統合および強化できます。

### アプリケーション サービスに対するデシリアライゼーション攻撃

2017 年の時点では、デシリアライゼーション攻撃は、多くはありませんが、莫大な影響を与えています。Apache Struts のデシリアライゼーションのインジェクション脆弱性は、攻撃者が Equifax の情報漏洩に利用したセキュリティホールで、これにより米国で 14,800 万人、英国で 1,520 万人の市民の ID が盗まれました。[3] シリアライゼーションとは、アプリケーションがそのデータをトランスポート用の形式に変換することで、デシリアライゼーションは、そのデータの形式を元に戻すプロセスです。アプリケーションは現在、データシリアライズされた通信ストリームを必要とするサブシステムで構成される、ネットワーク化されたクラスタであるため、デシリアライゼーション攻撃はますます一般的になっています。攻撃者は、シリアライズされたデータ ストリームにコマンドを埋め込み、これらをフィルタ処理せずにアプリケーション エンジンの中心に直接送ります。デシリアライゼーションに関連する Exploit-DB スクリプトは 30 あります。アプリケーションで、シリアライゼーション データ ストリームなどすべてのユーザ入力をスキャンおよびフィルタ処理する必要があります。

### トランスポート層に対する攻撃への保護

調査回答者の 63% は、Web アプリケーションに必ず SSL/TLS を使用していると答えていますが、これらのアプリケーションの大多数（76 ～ 100%）で SSL/TLS の暗号化を使用していると答えた回答者はわずか 46% でした。トランスポート層の暗号化標準（SSL や TLS 1.0 など）は「安全でない」ので廃れていますが、今でも数多く使用されています。そのため、攻撃者からの傍受または中間者ハイジャックのリスクは継続的に存在します。さらに、組織の 47% は、アプリケーションの信頼性が下がる自己署名証明書を使用していると答えています。組織は、すべてのアプリケーションが許容レベルの暗号化を実行し、適切なサードパーティにより署名された証明書を使用する必要があります。

### アプリケーションの任意のコンポーネントに対するサービス拒否攻撃（DoS 攻撃）

サービス拒否攻撃（DoS 攻撃）はさまざまな手段で仕掛けられます。たとえば、ソフトウェアで見つかった欠陥に直接仕掛けられることもあります。Exploit-DB のデータベースには 5,665 のサービス拒否エクスプロイトがあります。一般的には、攻撃者が制御するデバイスの軍隊または thingbot から大量の分散型サービス拒否（DDoS）攻撃が仕掛けられ、直接的なトラフィックまたは増幅 / リフレクションしたトラフィックでアプリケーションに多大な負荷を与えます。さらに危険な攻撃は、トラフィック フラッド攻撃とアプリケーション サービスの脆弱性を狙った攻撃を組み合わせたハイブリッド攻撃です。これらの攻撃は、Web アプリケーション インフラストラクチャを操作して、サイトの限界まで負荷をかけるように調整およびカスタム設定されます。F5 では、このような攻撃により、1 分間に 2,000 ページ以上のリクエストが数十万の個別の IP アドレスから送信されたことを確認しています。DDoS 攻撃は、アプリケーション層のすべてのレベルが対象となる攻撃なので、どの組織も DDoS 対策戦略を実施することが重要です。

### アクセスをハイジャックするためのクライアントに対するスクリプティング攻撃

アプリケーション クライアントに対する攻撃は、情報漏洩報告では公開される可能性が低い個人を標的とし、アプリケーション情報漏洩の場合とは異なり規制上の報告義務がないため、過少報告されることがよくあります。クライアントがハイジャックされる一般的な方法は、最も一般的な脆弱性の 1 つであるクロスサイト スクリプティング（XSS）です（2017 年の WhiteHat Security の脆弱性データの 30%、Exploit-DB スクリプトの 9.24%）。XSS 攻撃では、ユーザ認証情報が盗まれる、またはアクセスがハイジャックされることがよくあります。クロスサイト リクエスト フォージェリ（CSRF）は、クライアントがハイジャックされ知らないうちに Web サイトで不正コマンドを実行させられる別の攻撃です。どちらの攻撃にも、Web サイトのどこかで攻撃者により仕掛けられた悪意のあるスクリプティング コードに遭遇したクライアント アプリケーションが巻き込まれます。サイトで、セッション cookie を HTTP のみに設定してドメインを制限する、あるいは X フレーム オプションを DENY に設定するなど、Web サーバ オプションを使用することでスクリプティング攻撃を軽減できます。

**5,665**

EXPLOIT-DB のデータベースには 5,665 のサービス拒否エクスプロイトがあります。

### アプリケーション クライアントに対するマルウェア攻撃

クライアントは、ブラウザをハイジャックしてアプリケーション認証情報を盗聴または傍受するマルウェアの攻撃も直接受けます。金融ログインを狙ったマルウェアは、ブラウザおよびモバイル クライアントの両方において非常に一般的です。クライアント デバイスの保護は制御が難しいのでこれまでほとんど無視されていましたが、EU 一般データ保護規則（GDPR）などの厳しいデータ プライバシー法により、適切に作成されていないアプリケーション クライアントがその規制の対象となる可能性があります。一部の Web アプリケーション ファイアウォール ソリューションは、不正クライアントを検知して、そのアクセスをフィルタ処理することで、疑わしい接続を監視できます。

## 十分なアプリケーション プロテクションと他者との比較

F5 Ponemon セキュリティ調査は、他の組織がアプリケーション セキュリティにどのように向き合っているかに対するいくつかの洞察を提供します。最初の質問である責任者についての質問では、回答者の 28% は、アプリケーション セキュリティ リスク管理プロセスの責任を担うのは CIO または CTO だと答えました。CISO については 10% 程度ですが、今後は情報漏洩発生時の責任を担うことになりそうです。

アプリケーション セキュリティ対策を強化する上での主な 3 つの障害は、「アプリケーション層が可視化されていない」、「熟練の担当者または専門的な担当者の不足」および「クラウド環境への移行」でした。1 つめと 3 つめの回答はどちらも、アプリケーション層の分析と装備についてですが、これは、スキャン、監視および開発チームとの連携により解決できる問題です。

## アプリケーション セキュリティに使われる最も多い方法は、26% で WEB アプリケーション ファイアウォールでした。

アプリケーション セキュリティに使われる最も多い方法は、26% で Web アプリケーション ファイアウォールでした。他には、アプリケーション スキャン（20%）および侵入検査（19%）が挙げられました。驚くことに、組織の 26% は、アプリケーションのセキュリティ向上に役立つアプリケーション強化を行っていません。

## アプリケーションを保護するための 4 つの方法

全体的な調査結果では先行きが不安に思えますが、以下の 4 つの方法を実施することで、アプリケーション セキュリティの向上に大きな影響を与えることができます。ほとんどの場合、これらの実施は難しくありません。

**1. 現状を把握する。**

使用するアプリケーション、およびそれらがアクセスするデータ リポジトリを理解します。難しいかもしれませんが、必要なことです。組織に必要なアプリケーション、および顧客のために構築するアプリケーションに焦点を置きます。組織に必要なアプリケーションについては、定期的なスキャンおよびインベントリを行います。ユーザが利用する外部アプリケーションについては、アプリケーションの数と使用状況の管理に Cloud Access Security Broker（CASB）が非常に役に立ちます。内部アプリケーションについては、開発者チームとうまく連携して、アプリケーション、今後のアプリケーション計画および開発環境を特定します。

**2. 攻撃面を減らす。**

攻撃者は、インターネット上で直接または間接的に確認できるアプリケーション サービスのどこかに悪用できる可能性がないか探します。アプリケーションの層が複数あり、サード パーティとデータを共有するアプリケーション プログラミング インターフェイス（API）が普及していることから、攻撃面は広大です。

公開されるすべてにおいて、アクセス制御、パッチ処理および攻撃への対策強化を実施する必要があります。最も多くの調査回答者が選んだ、優れた Web アプリケーション ファイアウォール（WAF）は、このための時間を提供できます。一部の WAF は、アプリケーション トラフィックをスキャンして、既知のエクスプロイト攻撃をブロックすることで、「仮想パッチ処理」を実行できます。これらは、脅威インテリジェンス フィードおよび環境の脆弱性スキャンからの自動シグネチャ アップデートから何をブロックすべきかを認識します。これにより、新しいエクスプロイトが公開されたらすぐにパッチ処理しなければならないという時間的圧力が緩和され、運用チームは余裕を持って、修正を適切に検査してから提供できます。セキュリティ チームが WAF で必要なセキュリティ ブロック機能を有効にしていないために発生している回避可能なセキュリティ インシデントは数多くあります。優先順位の低いアプリケーションで発生した情報漏洩が、優先

順位の高いシステムまで影響しないように、アプリケーションを分離および分割しておく必要もあります。これは、コード上や、サーバ隔離、サンドボックス、ユーザの最小権限の規則およびファイアウォールでも実行できます。

**3. リスクに基づき防御の優先順位を決める。**

重要なアプリケーションを理解して、攻撃面を最小限に押さえたら、追加リソースが必要なアプリケーションを特定します。リスク分析は、適当な当て推量や偏った意思決定より優れていれば十分で、完璧である必要はありません。そのため、リスク戦略にデータを利用して、攻撃者が何を狙っているかを解明します。アプリケーションに関する重要なデータは、セキュリティ検査およびスキャンから得ることができます。内部開発コードの検査は、内部スキャナやコード レビューを利用して、または豊富な知識に基づいた独自の見解を示すサード パーティと協力して行います。この情報に基づき、内部開発アプリケーションのリスクを適切に評価できます。

リスク分析は、適当な当て推量や偏った意思決定より優れていれば十分で、完璧である必要はありません。

**4. 柔軟な統合型の防御ツールを選ぶ。**

既存および新しい脅威を阻止、検知し、被害から修復できる、柔軟で強力なソリューションを適切かつ管理可能な方法で選択する必要があります。前述の Web アプリケーション ファイアウォール、脆弱性スキャン ソリューションおよび CASB といった技術制御のほかに、アプリケーションが依存するすべての層まで保護を拡張する必要があります。DNS サーバは、DNS を重視したファイアウォールで適切に保護して、高可用性を提供する必要があります。トランスポート層通信は、現在の許容できる暗号標準で暗号化する必要があります。Web サーバは、HTTP Strict Transport Security（HSTS）を使用して、重要なデータ フロー全体を確実に暗号化する必要があります。セキュリティ ソリューションは、一貫していて、セキュリティ エンジニアに適切に理解されている必要があります。これらのソリューションを使用していても、多くの情報漏洩が発生していますが、それは単に製品を適切に理解および設定していないためです。

**DDoS 攻撃の潜在的な影響に基づき、オンプレミス スクラビング設備またはホステッド ソリューションを使用して、ネットワーク、アプリケーションおよびインフラストラクチャ レベルでアプリケーションを保護する必要があります。重要なことは、アプリケーションへのリスクおよび可能性のある脅威に基づきソリューションを調整することです。**

アプリケーション クライアントの保護については、アプリケーションにアクセスする顧客と、インターネット上のアプリケーションにアクセスする内部ユーザの 2 つのクラスを考慮してください。内部ユーザを保護するには、フェデレーション アイデンティティまたは多要素認証（MFA）を使用して、強力で確実なアクセス制御を実施してください。ユーザ名とパスワードの認証をサポートするだけのアプリケーションでは（この場合ユーザがパスワード推測や認証情報詐取などの攻撃にさらされます）、外部アプリケーションの認証を統合および強化する CASB が役立ちます。価値のあるアプリケーションで、顧客のアプリケーション クライアント セッションを保護することも重要です。より強力で柔軟な WAF システムでは、ボット攻撃、ブルートフォース（総当たり）攻撃および疑わしい場所からのログインを検知することで、顧客のアプリケーション クライアント セッションを保護できます。この簡単な検証は、アプリケーションにアクセスする顧客の保護を強化する優れた方法です。

## アプリケーション プロテクションの将来の展望

### サーバレス アプリケーション

「サーバレス」は、開発者が Web アプリケーションを構築する新しい方法で、サーバおよびアプリケーションのサポート インフラストラクチャを意識する必要がありません。コードがサーバ内で実行して他のサーバを呼び出す従来の方法とは異なり、開発者が作成するコードおよびスクリプトは、機能およびサービスを直接トリガする API に接続されます。サーバレス コンピューティングにより、開発者は、より迅速で合理化されたアプリケーション開発に集中できます。長期的には、サーバレス アプリケーションは、より優れた柔軟性と拡張性を提供します。注意すべきことは、サーバレス アプリケーションは、すべての層において従来のアプリケーションと同じ種類の攻撃に対して、特に主要なユーザ ログイン ページおよび API に対して脆弱であることです。

**765**

一般的な組織では 765 種類の WEB アプリケーションを使用しているので、アプリケーション セキュリティのアウトソーシングは増えると考えられます。

### アプリケーション セキュリティのアウトソース

F5 Ponemon セキュリティ調査で主な障害として挙げられた、セキュリティにおける熟練の担当者または専門的な担当者が不足している状況は、アプリケーション セキュリティ担当者にとってさらに悪化しています。また、一般的な組織が 765 種類の Web アプリケーションを使用していることから、アンチ DDoS または Web アプリケーション セキュリティ監視などのセキュリティ機能のアウトソースであっても、製品の一部としてセキュリティ サービスを提供するホステッド プラットフォームへの移行であっても、アプリケーション セキュリティのアウトソースは増えると考えられます。アウトソースでは、具体的なセキュリティ要件と、アウトソース企業のアプリケーション セキュリティにおける能力、取り組みおよび専門的知識が一致しなければなりません。

### Transport Layer Security の改善

Transport Layer Security（TLS）1.3 は、従来のプロトコルとは大きく異なるため、セキュリティ コミュニティはその採用に苦労しています。長期的には量子コンピュータへの不安も TLS を脅かします。私たちの見解では、Transport Layer Security は、量子コンピュータ以外からさらに激しい衝撃を受けると予想していますが、これらの衝撃が何かはまだ分かりません。組織は、TLS 1.3 のブラウザ サポートと、量子コンピューティングのネットワーク暗号化およびアップデートに関する主要なコンプライアンス標準に注意する必要があります。

最近のアプリケーションは、スクリプト、ライブラリ、サービスおよびデバイスの集合体なので、セキュリティ ツールがこのパラダイムに対応できることを期待しています。

### これらの層に続くアプリケーション プロテクション

最近のアプリケーションは、スクリプト、ライブラリ、サービスおよびデバイスの集合体なので、セキュリティ ツールがこのパラダイムに対応できることを期待しています。将来的には、開発者は、現在の壊れやすく、過度に依存するエコシステムとは大きく異なり、より優れた方法で選択できるセキュア コンポーネントおよびフレームワークを利用できるようになります。開発者には、セキュリティ ステータスおよびイベントを標準形式で報告する機能を備えた「デフォルトでセキュア」なアプリケーション フレームワークが必要です。セキュリティ担当者には、安全かつ便利な方法で、生産中のアプリケーション コンポーネントを継続的に検査できるセキュリティ スキャナが必要です。

# 02

# 概要

この報告書でアプリケーションを中心とする理由は、単純に、私たちがインターネットを使用するからです。アプリケーションは、私たちに代わり通信、計算、処理、保存、検索、調整および予想を行います。アプリケーションはビジネスための筋肉です。そのため、アプリケーションは、必要なときに、予測通りに機能する必要があります。


## インターネットを使用する理由はアプリケーション

アプリケーションは、必要な情報を形成および保持する、データのコンテナでもあります。私たちは、アプリケーションにデータを提供し、アプリケーションからデータを抽出します。アプリケーションは、データの門番であり通訳者でもあります。データは貴重なので、アプリケーションは、最も価値のある資産の倉庫です。

F5とPonemonが2018年に行った「Web Application Security in the Changing Risk Landscape: Global Study」では、一般的な組織は、765 種類の Web アプリケーションを使用していて、そのうち平均 34% をミッションクリティカルなアプリケーションと考えていることが分かりました。また、調査回答者は、重大な Web アプリケーション セキュリティ インシデントに関連する平均損失額を約 800 万ドルと予想しました。

## 調査対象の組織は、アプリケーション 34% がミッションクリティカルだと考えています。

このような大きな数字（およびビジネスにおけるアプリケーションの重要な役割）のために、私たちは 1 年かけて F5 Labs 内外のアプリケーションに関するデータを調査し、まとめました。この目的は以下の質問の答えを出すことです。

- **アプリケーションは何で構成されているか**
- **アプリケーションへの脅威とは何であり、どのように攻撃されるか**
- **アプリケーションを保護するために何をすべきか**

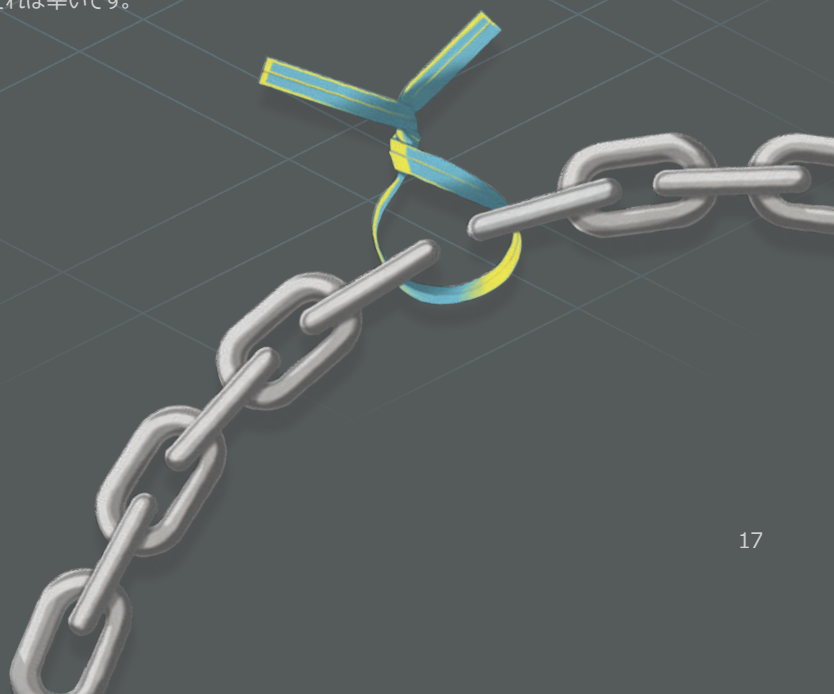

図 1：アプリケーションはビジネス

図 2: アプリケーションは
データの門番


これらは難しい問題で、完全な答えは出ていません。これは当然なことです。アプリケーション セキュリティは複雑であり、プログラミング、インフラストラクチャ、サポート ユーティリティ、サードパーティへの依存、システム運用、アプリケーション ユーザおよびセキュリティ制御など多くの異なる分野が関連します。この鎖の輪が 1 つでも綻びると、セキュリティの侵害につながることがあります。さらに、攻撃者およびその進化し続ける戦術と破壊行為への欲求も考慮しなければなりません。

## 回答者は、 重大な WEB アプリケーション セキュリティ インシデントに関連する平均損失額を約 800 万ドルと予想しました。

セキュリティ自体は意味がありません。 意味を持たせるためにリスクを考慮する必要があります。これは、 古き良き方法である、 徹底したリスク分析により行います。もちろん、 リスク分析にはある程度の不確実性が伴います。しかし、推測だけで最善の結果を期待するよりも何かを評価する方がはるかに価値はあります。 脅威が脆弱性を悪用して、 迷惑な影響を及ぼす可能性がどのくらいあるかを調査します。その後、 これらのリスクを軽減できるセキュリティ制御の種類を検証します。この報告書がアプリケーション防御の強化に役立てれば幸いです。



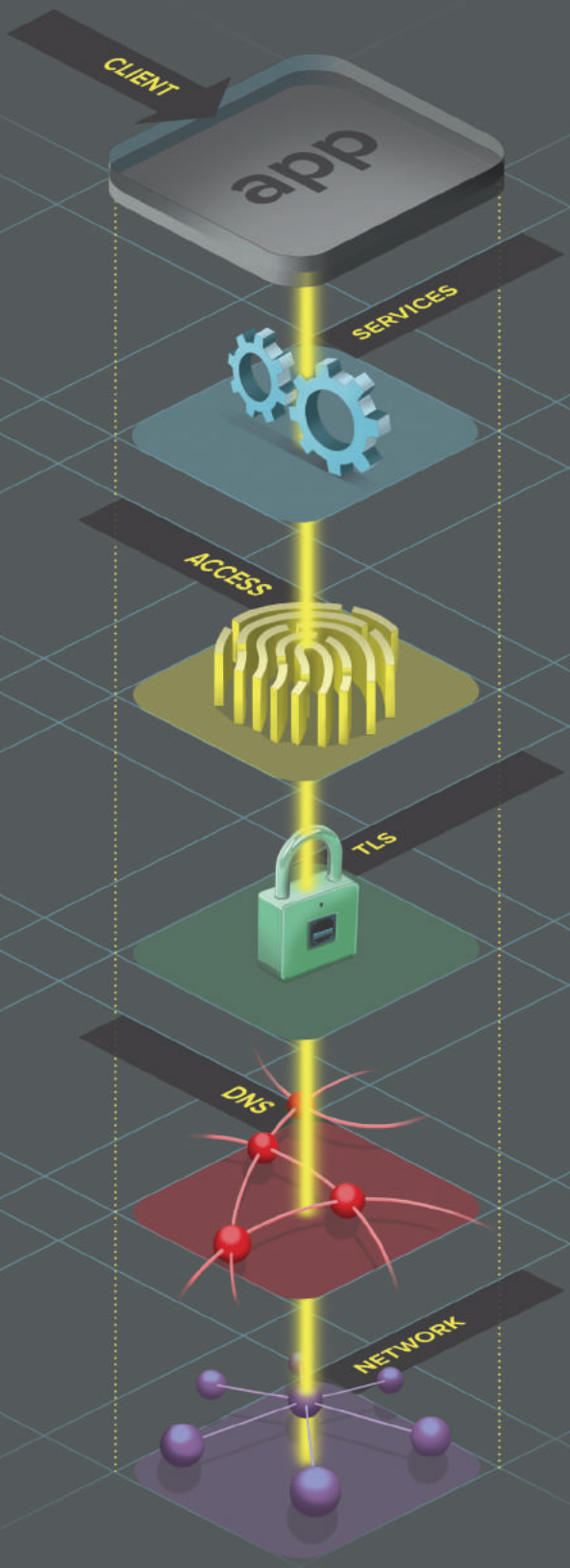



図 3: アプリケーション層

# アプリケーションとは

私たちが日常使用する Web アプリケーションのほとんどは「群体生物」です。Web アプリケーションを構成する別々の独立した多数のコンポーネントは、運用要件およびサポート インフラストラクチャ（クラウドとオンプレミスの両方）が異なる別々の環境で稼働し、それぞれがネットワークでつながっています。ここでは、アプリケーションの各層およびその下層は潜在的な標的であるため、アプリケーションをこれらが相互作用するひとつなぎのものと表現します。適切な防御を評価できるように、個々の攻撃面を理解する必要があります。

図 3 は、一般的な Web アプリケーションの主要な層、アプリケーション サービス、TLS、DNS およびネットワークの概要です。

これらの層内には、その下層およびコンポーネントがあります（次のページの図 4 参照）。


図 4：アプリケーションの下層およびコンポーネント

**以下に、 アプリケーション層とその下層の概要を示します。**

## アプリケーション サービス層

インターネットが誕生したとき、Web サイトは、Web サーバ上に公開されたハイパーリンクを含む静的な HTML ドキュメントでした。その後、ユーザ入力に基づく動的なページを使用する Common Gateway Interface（CGI）標準が誕生しました。動的な Web アプリケーションは突然生まれました。同時に、匿名の信頼できないユーザが、悪意のあるコンテンツを Web ページにインジェクトするようになりました。ここから、Web アプリケーションのハッキングが急増し始めました。

F5 Labs の『2016 TLS Telemetry Report』で示されているように、最も普及している 3 つの Web サーバは、Apache（1 位）、NGINX および Microsoft Internet Information Server（IIS）です。これらのサーバは、Web アプリケーションの基礎ですが、モジュール、プラグイン、ライブラリ、フレームワークおよび機能を追加する拡張などのアドオンも可能にします。一般的な Web アプリケーションは、通常、1 つ以上の Web サーバ アドオンをそのアーキテクチャに使用します。これにより、複雑性が高くなり、アプリケーションの攻撃面が広がります。

一般的な WEB アプリケーションは、 通常、WEB サーバ アドオンをそのアーキテクチャに使用します。これにより、複雑性が高くなり、 アプリケーションの攻撃面が広がります。

サービス層には、サーバ側インフラストラクチャ、サーバ側フレームワークおよびアプリケーション ソース コード（これは以下のようにさらに細かく分類されます）の 3 つの下層が含まれます。

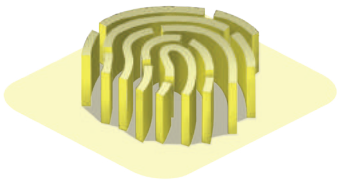
### 内部コード

内部コードは、Web アプリケーション固有の機能に一意な Web アプリケーションの一部です。これは、コア アプリケーション（Microsoft SharePoint または Salesforce など）、あるいは組織がそのニーズのために特別に開発または変更した内部コードです。

### 外部コード

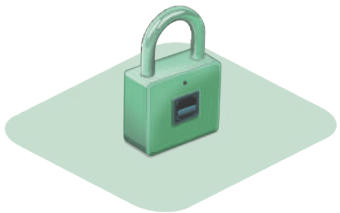
外部コードは、コア アプリケーションにリンクされる再利用可能またはサード パーティのコードを参照するアプリケーションの一部です。たとえば、リンクされたライブラリ、プラグイン、フレームワーク、サーバ側スクリプトおよび外部リンク コードなどです。外部コードは、通常、ある程度の検査を受けていますが、多くの攻撃者にもセキュリティ ホールがないか調べられています。Apache Struts は、このカテゴリに分類され、最新のパッチを適用していない場合に大きなセキュリティ ホールが生まれる典型的な例です。

サーバ側インフラストラクチャの下層は、アプリケーションをサポートするスタンドアロン サーバで構成されます。これには、Web サーバやコンテンツ デリバリ ネットワーク（CDN）など、およびアプリケーション、データベース、ファイル サーバが含まれます。サーバ側インフラストラクチャは、外部リンク コードよりもモノリシックですが、交換が可能なので、理論上はロードバランスとパッチ処理が簡単です。しかし、アプリケーション層（レイヤ 7）を狙った DDoS 攻撃では、そのサポートするアプリケーションをダウンさせるために、ネットワークではなく、サーバ側インフラストラクチャが直接的な標的となることがよくなります。



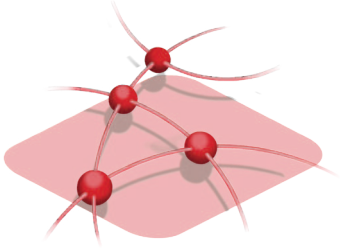
## アクセス制御層

アプリケーションのアクセス制御層は、ユーザが認証および許可のために通過するゲートウェイです。アプリケーションは、さまざまな方法でアクセス制御を導入できます。クライアント認証情報は、通常、データベースに保存されます。または、アプリケーションが、たとえば、Lightweight Directory Access Protocol（LDAP）サーバを使用して、共有オンプレミス ソリューションを利用することもできます。また、フェデレーション サービスの場合のように、内部または外部のいずれかで、シングル サインオン（SSO）ゲートウェイに接続することもできます。



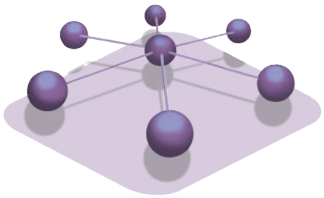
## TRANSPORT LAYER SECURITY 層

トランスポート層は、インターネットや便利な Wi-Fi サービスなどの信頼されていないネットワークをネットワーク パケットが通過するときの暗号化を提供します。パケットの暗号化されたカプセル化は、通常、Web アプリケーション サーバの近くから始まり、クライアントに到達するまで続きます。Transport Layer Security（TLS）は、攻撃者が送信中のデータを改竄していないことを確認して、信頼できる認証機関からの適切なドメイン証明書によりアプリケーションを検証します。この層には、一般的な HTTPS プロトコルと TLS、および時代遅れとなった SSL プロトコルも含まれます。



## DOMAIN NAME SYSTEM サービス層

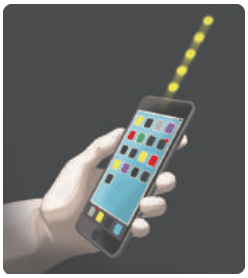
インターネットの「アドレス帳」である DNS は、同意された運用標準に基づき稼働する、世界的に分散したサービスです。アプリケーションに接続するクライアントは、機能的で信頼できる DNS に大きく依存します。DNS が妨害、不適切に設定または改竄された場合、アプリケーションのセキュリティに重大な影響を及ぼします。アプリケーション自体も、アプリケーションの直接制御外の他のサービスへの接続が必要な場合があるので、正確で機能的な DNS に依存します。この層には、クライアントおよびアプリケーションに必要なすべての DNS サーバだけでなく、これらのドメインの関連レジストラも含まれます。



## ネットワーク層

クライアントは、アプリケーション サーバに接続する必要があります。この接続は、ほぼ必ずインターネットを介して発生します。一般向け Web サイトでも、マシン間の API でも、Web トラフィックの最も一般的なプロトコルの 1 つは HTTP です。よりセキュリティを意識したアプリケーションでは、これらの接続は HTTPS を使用して暗号化されます。

ネットワーク層には、インターネット サービス プロバイダ (ISP)、ISP から顧客のオンプレミスへの最後の接続、インターネット ルーティング プロトコルなど、すべてのアプリケーション ネットワーク サービスも含まれます。



## アプリケーション クライアント

ほとんどのアプリケーションは、物理または仮想のいずれかの環境のサーバとして稼働できますが、ユーザにデータを送る、またはユーザからデータを受け取るクライアント インターフェイスが必要です。現在、インターネットから独立して接続されずに使用できる便利な (Windows Notepad などの) アプリケーションはほとんどありません。

最も一般的な Web アプリケーション クライアントは Web ブラウザです。Web ブラウザは、1993 年に National Center for Supercomputing Applications によりリリースされた Mosaic よりもはるかに進化しています。ほぼすべての Web アプリケーションは、Web クライアントが JavaScript または Flash などのアクティブ スクリプトを実行することを想定しています。

アプリケーション自体もブラウザ自体でアクティブ スクリプトを実行することが増えています。このような場合、クライアントとブラウザは、HTTP を介して通信を行い、データおよびコマンドをやり取りして処理します。これにより、セキュリティの新たな影響が生まれるので、検査および防御のための新しい要件が必要になります。コード パーサは、結果の優先順位決定が難しいので新しいセキュリティ問題が生じることがよくあり、コードとユーザ入力の区別に問題が生じることもあります。

アプリケーション クライアントには、モバイル アプリケーションも含まれます。ほとんどの場合、これらは、既存の Web アプリケーションで事前に構成されている Web ブラウザ インターフェイスです。アプリケーションおよび IoT デバイスは、他のアプリケーションを呼び出して、データを送信、受信または処理できます。通常、これは、API およびアプリケーション サービス接続を介して行われます。


図 5：アプリケーション脅威

## 各層における脅威

アプリケーション スタックの各層には、攻撃者の標的にされる一意な脆弱性があります。アプリケーションを保護するには、図 5 に示すように、各アプリケーション層の内部の複雑さを理解することが重要です。各層に示す各種脅威は、この報告書の各項で詳しく分析しています。

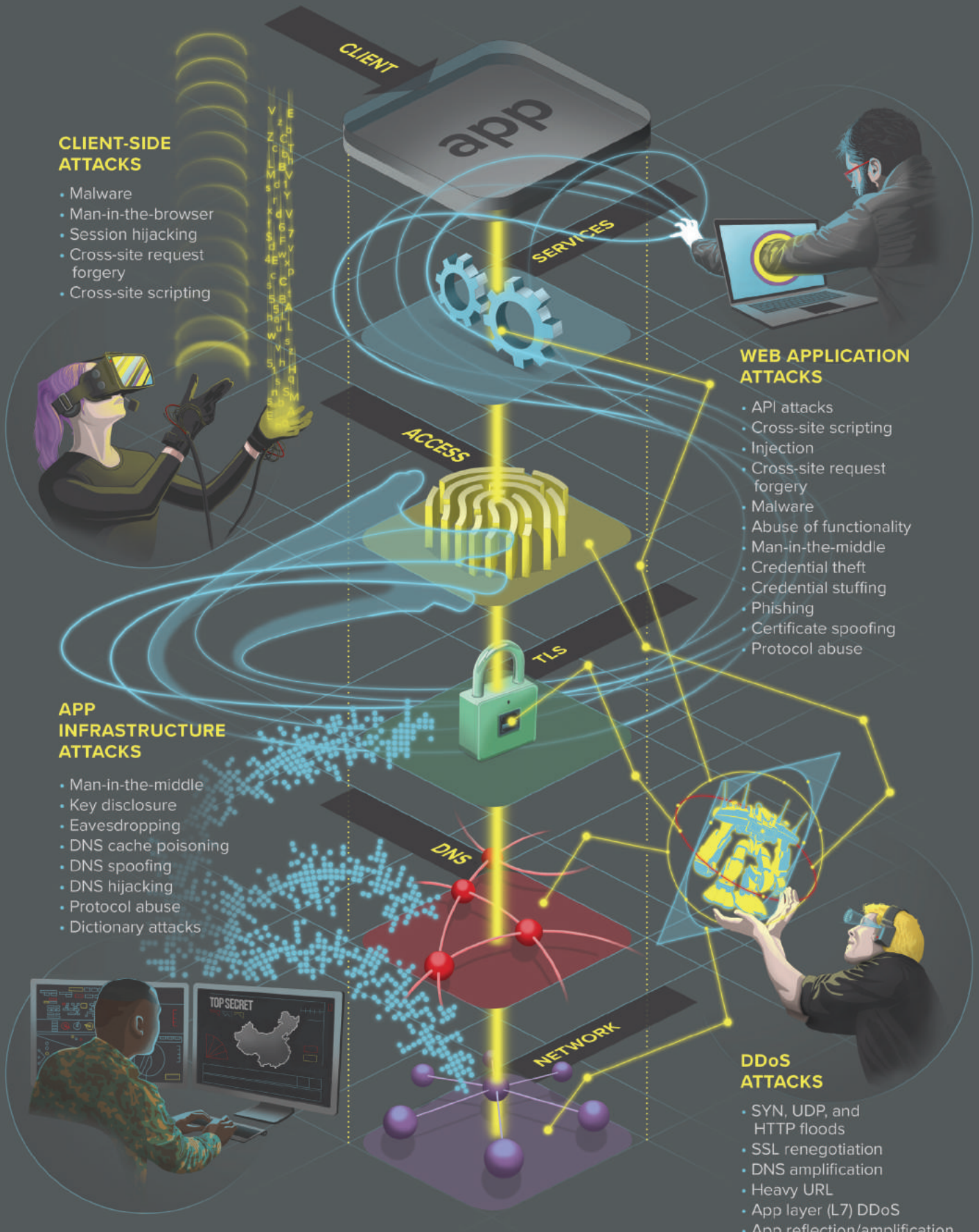
## アプリケーションへの攻撃方法

当たり前の事実ですが、インターネットに接続するものはすぐにでも攻撃を受けます。

当たり前の事実ですが、インターネットに接続するものはすぐにでも攻撃を受けます。攻撃者は、ほとんど費用をかけずに、アプリケーション サービスに探りを入れ、何を得られるか確認できます。さらに攻撃者はこれを 24 時間体制で実行できます。新しいエクスプロイト技術が発見されれば、攻撃者はすぐに脆弱なサーバを探してインターネット全体を循環します。

アプリケーションに仕掛けられる毎日数千の（その多くは新しい）攻撃を分類できれば、防御能力を大幅に改善できます。攻撃は、アプリケーションのすべての層で、場合によっては複数の層で同時に起こる可能性があり、実際に起こっています。アプリケーション攻撃は、Web アプリケーション攻撃、アプリケーション インフラストラクチャ攻撃、サービス拒否攻撃（DoS 攻撃）およびクライアントへの攻撃に分類して検証します（図 6）。

図 6：アプリケーション攻撃の各部門に割り当てられる脅威



## WEB アプリケーション攻撃

アプリケーション サービスは複雑で、大量にあることから、ここでのリスクが大量に見つかることは明らかです。
F5 Labs が 12 年に及ぶ 433 件の情報漏洩事例を分析した結果、情報漏洩の 86% は、アプリケーション自体またはアプリケーションの認証情報を持つユーザを狙った攻撃から始まっていました。

アプリケーション スタックでは、2 つの主要領域が攻撃を受けます。1 つは、サービス層です。これには、内部コード、外部コードおよびサーバ側インフラストラクチャといった下層に対する攻撃が関連します。ここでの攻撃には、Web サーバのバッファ オーバーフロー、Apache Struts などの Web サービスのエクスプロイト、またはビジネス ロジックの悪用などのカスタム コードで見つかった一意な脆弱性などが含まれます。
F5 Labs の分析では、サービス層は情報漏洩の 53% で初期の攻撃標的となり、主な攻撃ベクトルとなっていました。

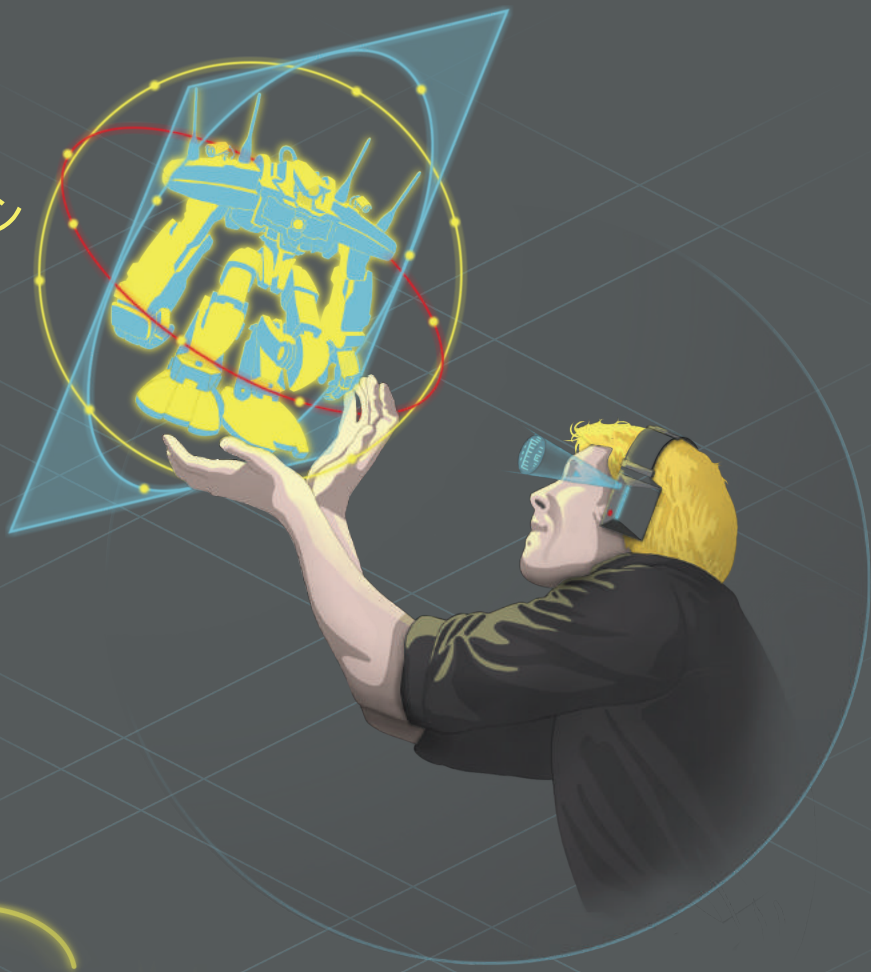
Web アプリケーション攻撃には、クレデンシャル スタッフィング、ボットネットによるブルートフォース（総当たり）攻撃、中間者攻撃およびフィッシングによる認証情報の盗難などのアクセス制御攻撃も含まれます。F5 Labs による情報漏洩分析によると、アプリケーション アクセスに対する攻撃は 2 番目に多い初期ベクトルで、アプリケーション アクセスは情報漏洩事例の 33% で標的とされていました。


## アプリケーション インフラストラクチャ攻撃

アプリケーション インフラストラクチャとは、アプリケーションが依存するが、アプリケーション自体の外部にあるシステムのことです。アプリケーション インフラストラクチャへの攻撃には、Transport Layer Security（TLS）層、Domain Name System（DNS）層およびネットワーク層を標的とした攻撃が含まれます。

情報漏洩の 86% はアプリケーションまたはアイデンティティ攻撃から始まっていました。

# アプリケーション アクセスに対する攻撃は 2 番目に多い初期ベクトルで、アプリケーション アクセスは情報漏洩事例の 33% で標的とされていました。



## サービス拒否攻撃（DOS 攻撃）

攻撃者は、アプリケーションのすべての層および確認できるコンポーネントを攻撃できます。つまり、サービス拒否攻撃(DoS 攻撃) は絶えず存在する脅威です。アプリケーションのすべての層は適切に機能する必要があるので、DoS 攻撃からの保護をアプリケーション セキュリティ戦略の核と見なす必要があります。ほとんどのサービス拒否攻撃（DoS 攻撃）は、分散型（DDoS）です。つまり、攻撃は、ハッカーが制御するボットの軍隊から仕掛けられます。しかし、BIND DNS の欠陥を利用する TKEY クエリなど、単一パケットで仕掛けることができるサービス拒否攻撃（DoS 攻撃）がいくつかあります。[4]

## クライアント攻撃

アプリケーション クライアントも、一般的にはマルウェアまたは物理攻撃により攻撃を受けます。この報告書のクライアント脅威に関する記述では、アプリケーション クライアントとアプリケーションの関係に焦点を置いています。たとえば、マルウェアはアプリケーション クライアントに感染する場合がありますが、この報告書では、そのマルウェアがアプリケーション セキュリティに与える影響のみに注目し、アプリケーションの使用範囲外でユーザのセキュリティに与えるその他の損害には触れない場合があります。


03

# アプリケーションが攻撃を受けたときの組織への影響

アプリケーションのリスクについて話すとき、何を考えるのでしょうか。リスクは、脅威が発生し、その結果として組織の大事なものに影響する可能性がある場合のみ意味があります。この意味では、影響は、起きて欲しくない悪いことを評価する基準です。

# CISO にとって組織の第一の使命はアプリケーション ダウンタイムを防ぐことです。

通常、影響は、ドルでの推定損失額で計算され、組織が許容できる損失かどうかのしきい値となります。大手電子商取引サイトでは、1 か月あたり 1 時間程度のダウンタイムであれば許容できますが、2 時間だと許容できないかもしれません。

F5 と Ponemon の報告書『*The Evolving Role of CISOs and their Importance ～ the Business*』[5] では、CISO は、ダウンタイムを防ぐこと（肯定的に言えば優れた可用性を保証すること）が組織で最も重要な使命だと答えています。可用性はセキュリティ CIA の三本柱として知られる機密性、完全性、可用性の 1 つなので、これは当然のことです。

アプリケーションにより保存および処理されるデータについて、アプリケーションおよびデータの完全性を保護することは、特に世界規模での金融システムにおいて、将来さらに重要になる可能性があります。最近、カーネギー国際平和基金は、「平常時および戦争時におけるデータおよびアルゴリズムの完全性を損ねる」恐れのあるサイバー攻撃の実施を控えるように各国に要請しました。[6]

図 7：影響およびリスクの計算

## アプリケーションのリスクの分析

リスク評価を開始する適切な方法は、使用している重要なアプリケーションの総数を測定し把握することです。[7] アプリケーションの数を測定、分析および追跡することで、何をどこで保護すべきかが分かります。

### アプリケーションの数の測定と追跡

アプリケーションは非常に重要なので、これらを把握することに努力し続けています。最近では、ユーザが企業所有および個人所有のデバイスにアプリケーションをダウンロードできることで、これがより一層難しくなっています。さらに厄介なことは、多くの従業員が IT 部門に知らせず、または許可を得ずに、Web ベースおよびモバイル アプリケーションを使用する「シャドー IT」の存在です。これにより、組織には重大なセキュリティ リスクが生じています。その理由は、これらのアプリケーションはビジネス活動に使用されることが多く、企業の機密情報を含む、または共有するために使用することがよくあるためです。従業員が職務に関連する活動に個人所有の（監視されず、セキュリティが侵害されていることがよくある）デバイスを使用する問題もあります。

2018 年の F5 Ponemon セキュリティ調査のデータによると、大多数の組織は、すべてのアプリケーションを把握できていることにあまり自信がありません。ある程度の自信があると答えたのは回答者の 24% のみで、残念なことに、まったく自信がないと答えた回答者は 38% もいました。

調査対象の組織の 38% は、組織内のすべてのアプリケーションを把握できていることにまったく自信がありません。

### アプリケーションはどこにあり、どのくらい重要か

アプリケーションは、クラウド上でホストされたり、モバイル アプリケーションや SaaS ソリューションとしてホストされたり、組織外の広範囲に分散しています。ほとんどの組織で一般的に使用されるアプリケーションのうち、オンプレミスでホストされているのは 52% 以下です。

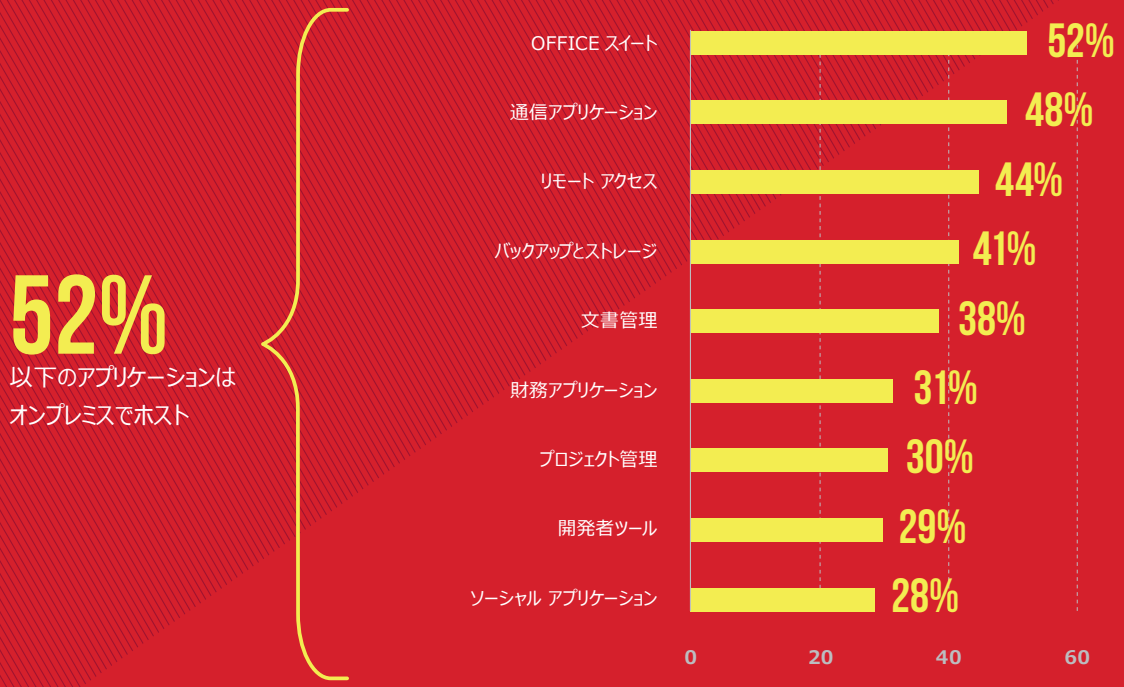
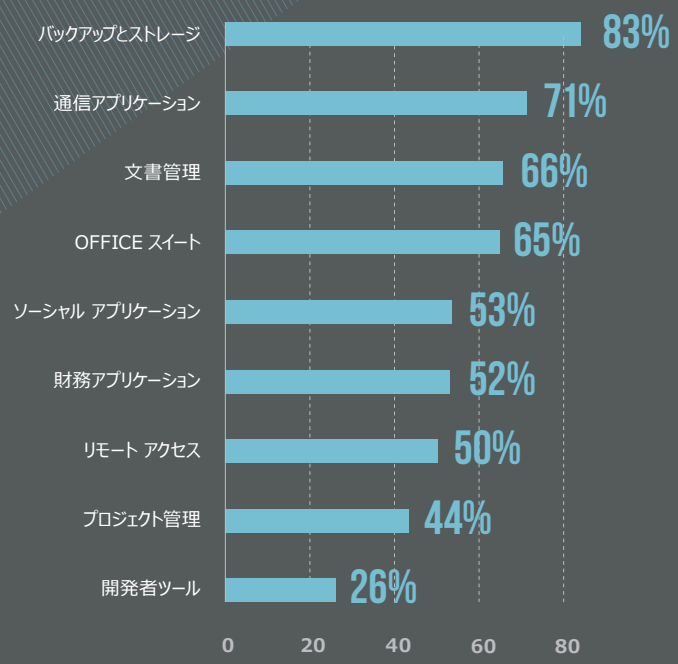



図 8：オンプレミスでホストされているアプリケーションの割合と種類

図 9：組織で最も一般的に使用されているアプリケーションの種類




# 83%

## 最近の多くのデバイスではローカル ストレージが限定的なので、バックアップとストレージが使用リストの 1 位になることは当然です。

どのようなアプリケーションがあるのでしょうか。バックアップとストレージは、最も一般的に使用されている Web アプリケーションの 1 つです。最近の多くのデバイスではローカル ストレージが限定的で、ベンダは消費者にクラウドベースのストレージ ソリューションを使用することを勧めているので、バックアップとストレージが使用リストの 1 位（83%）になることは当然です。

その他の一般的に使用されている Web アプリケーションは、電子メールなどの通信アプリケーション（71%）、文書管理と連携（66%）および Microsoft Office スイートのアプリケーション（65%）です。これほど多くはありませんが、50% を超える組織が、Web ベースのソーシャル、財務およびリモート アクセス アプリケーションを使用していると答えています（図 9 参照）。

図 10：最も重要な WEB アプリケーション

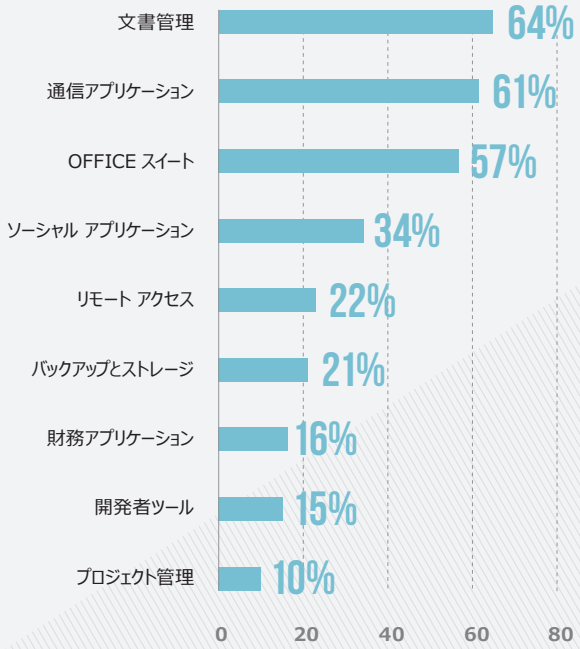
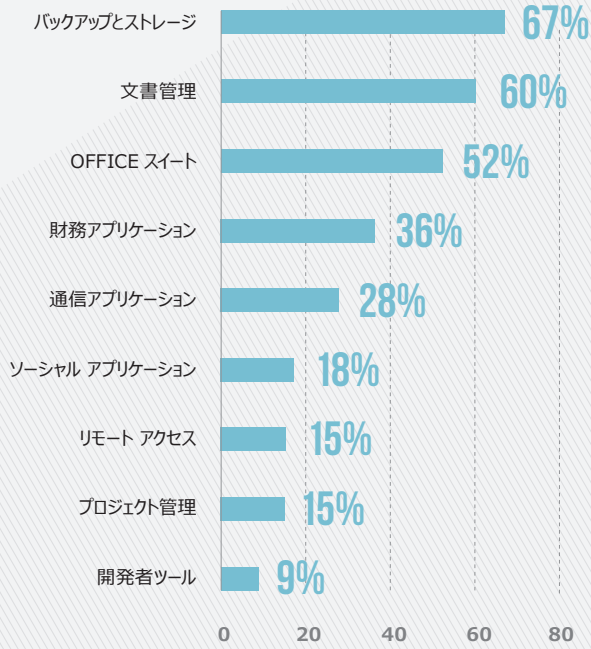



図 11：最も重要なデータが保存される WEB アプリケーション




# 28%

## 調査回答者の 28% は、アプリケーションの 1/4 ～ 1/2 がミッションクリティカルだと答えています。

最も使用されているアプリケーションと、組織がその業務で最も重要だと考えるアプリケーションを比較すると、たとえば、Web ベースの文書管理と連携、通信アプリケーション、Office スイートなど、一部が重なります。これらは、職務遂行のために日常的に使われるツールです。

興味深いことに、Web ベースのバックアップとストレージ アプリケーションは、組織の業務における重要性においては 21% まで下がりますが（図 10 参照）、組織の最も重要なデータが保存される Web ベース アプリケーションとしては 67% で最も多くなります（図 11 参照）。文書管理と連携アプリケーションおよび Office アプリケーションはここではそれぞれ 2 位と 3 位です。割合にすると、アプリケーションの 11 ～ 25% がミッションクリティカルだと考える回答者は 4 分の 1 以上（29%）で、アプリケーションの 25 ～ 50% がミッションクリティカルだと考える回答者も同程度（28%）でした。

図12： 機密または秘密情報の漏洩につながる攻撃の被害レベル

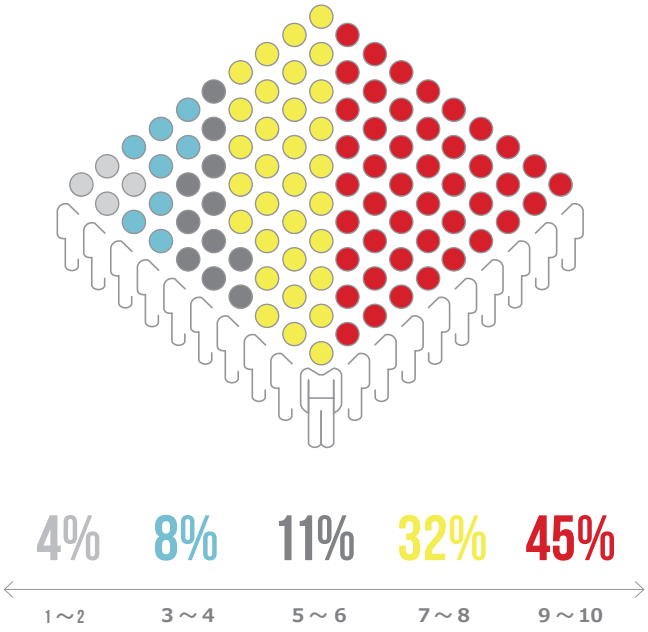
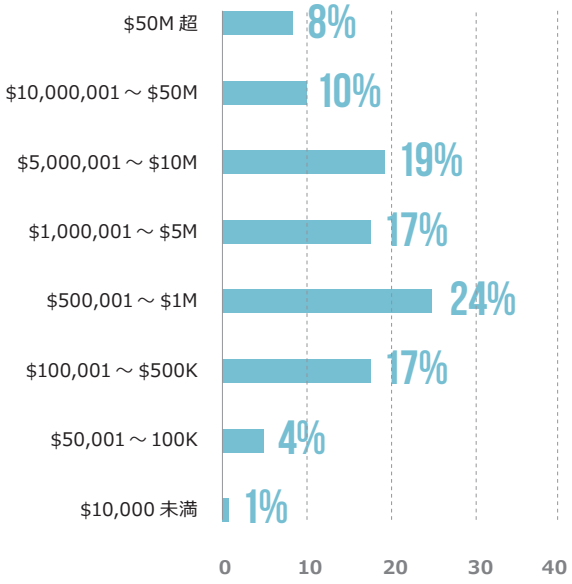



図13： 機密または秘密情報の情報漏洩による損失額




## 攻撃による損失額および影響 IMPACTS?

どのような攻撃でも組織に重大な影響を与えるのは明らかですが、機密情報漏洩、Web アプリケーションの改竄または重要なアプリケーションの可用性の損失のような特異な事象は単独で重大な影響を与えることができます。各組織は、ビジネス、業界、ビジネス モデルの種類に応じたさまざまな影響を評価し、以下のことを考慮する必要があります。

- 収益、コンプライアンスまたは契約責任に影響するシステムのダウン状態の長さ
- 攻撃がサービス提供能力に与える影響
- 従業員の生産性に影響する停電の長さ
- 回復不能なデータ損失による影響
- システム損失による他のシステムへの影響

**秘密または機密データの損失**

組織に与える影響に関して、10 満点の 10 段階評価で、調査回答者の 77% は、機密または秘密情報（知的財産や企業秘密など）の情報漏洩を 7 ～ 10 に評価しました（図 12 参照）この影響がほとんど、またはまったくないとした回答者はわずか 4% でした。

損失額（範囲は $10,000 ～ $5,000 万以上）で影響を検証すると、回答者の 4 分の 3 以上（78%）は、このような情報漏洩が発生した場合、企業の損失額は $500,000 を超えると答えています（図 13 参照）。回答者の 22% は、企業の損失額は $500,000 以下だと答えていますが、8% は $5,000 万を超えると答えています。

図 14：個人を特定できる情報（PII）が漏洩する攻撃の被害レベル

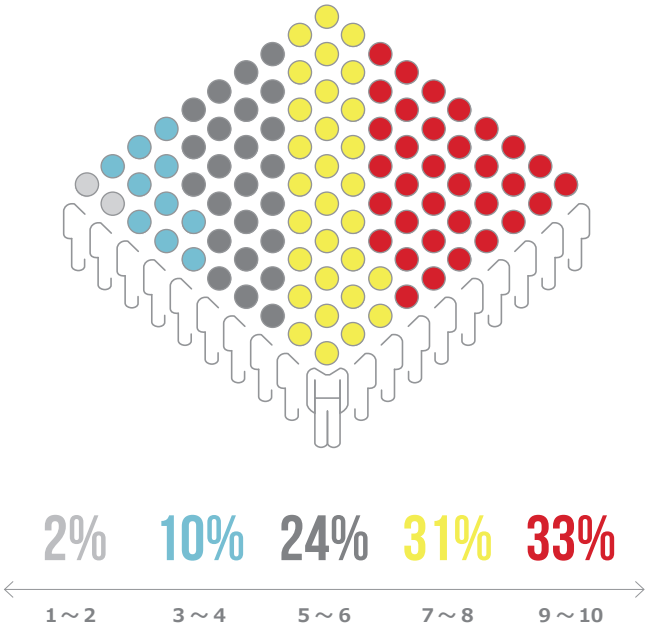
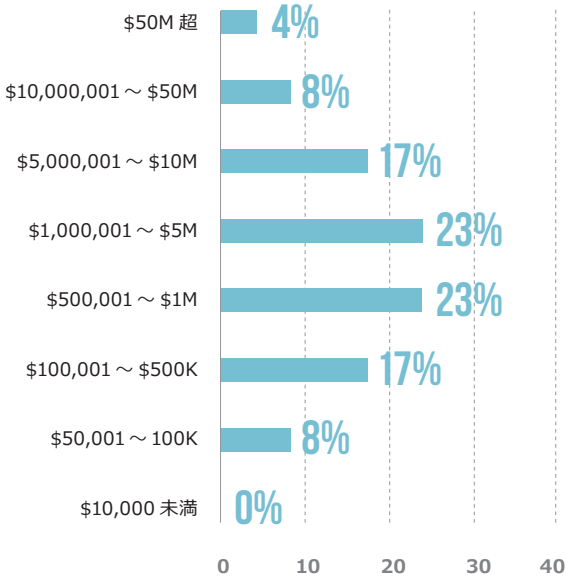



図 15：個人を特定できる情報（PII）が漏洩する攻撃の推定損失額




**個人を特定できる情報（PII）の損失**

PII の損失は、顧客、消費者および従業員のプライバシーを守ることができないと判断されるので、企業の評判に特別に壊滅的な打撃を与えます。ここでも、調査回答者が PII の情報漏洩の影響を 10 段階（10 点満点）で評価したところ、64% は 7 ～ 10 と非常に高く評価し、24% は 5 ～ 6 の中程度に評価し、影響が低いまたはないと答えた回答者はわずか 12% でした（図 14 参照）。

また、PII データの損失につながる情報漏洩の損失額の評価については、調査回答者は、秘密または機密情報漏洩の損失額の評価と非常に良く似た結果を示しました。$100 万～ $500 万の範囲に評価した回答者は 17% より 6% 多い 23% でした（図 15 参照）。

**アプリケーション自体の改竄**

データ損失を及ぼす情報漏洩は、疑いを持たない数百万の消費者に直接影響することがよくあるので、マスコミに大きく注目されます。アプリケーション自体を（攻撃者独自の不正な目的のために）改竄する攻撃は、世間からそれほど大きく注目されることはありませんが、このような攻撃の被害者となった企業は、Sarbanes-Oxley Act（SOX 法）によりその事実の報告が義務付けられています。アプリケーション改竄には、アプリケーションの外観または表示されるコンテンツを変える Web サイトの書き換えなども含まれます。また、パフォーマンスの低下、誤った、改竄された、または予期せぬページの表示、あるいはまったく別の Web サイトへのユーザの転送など、アプリケーションの機能を変える変更も含まれます。

アプリケーションのパフォーマンスが低下していると感じた組織は、暗号通貨のマイニングに使用されている可能性を憂慮してください。組織化されたサイバー犯罪者は、最も儲かるハッキング形態に素早くシフトし、大金を稼げるマイニングを始めています。このような変更はすべて、アプリケーションの完全性を損ねます。

図 16：アプリケーションの完全性を損ねる攻撃の被害レベル（アプリケーション改竄）

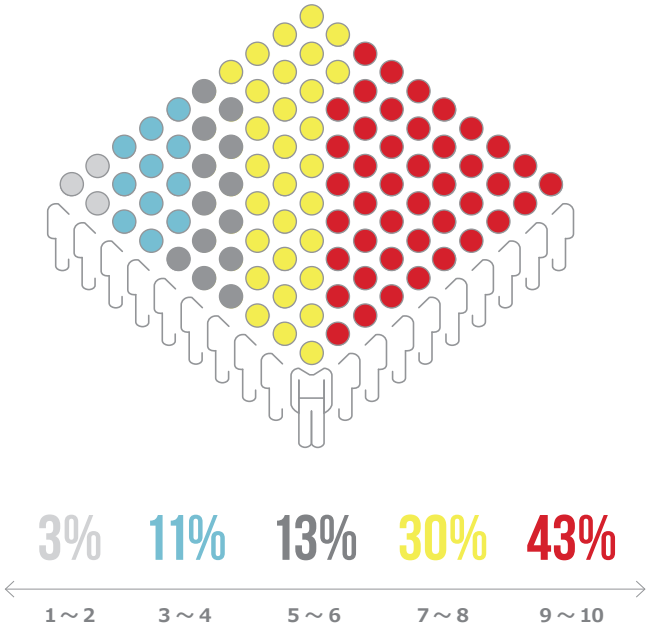
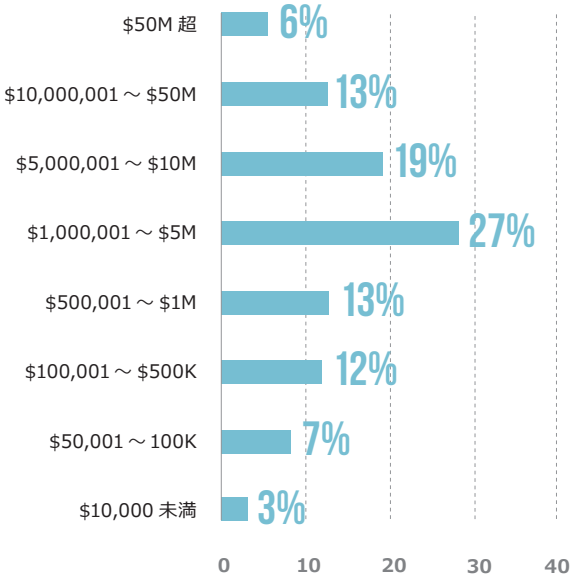



図 17：アプリケーションの完全性を損ねる攻撃の推定損失額（アプリケーション改竄）




たとえば、アプリケーションが改竄されたことで、ユーザがある製品を購入しようとしても、別の企業の製品が注文されてしまう場合の電子商取引サイトへの影響を想像してください。または、オンライン バンキング アプリケーションの送金機能が改竄され、ユーザの口座から資金が想定通りに引き出されるが、これらの資金が意図する受領者の口座ではなく、攻撃者の銀行口座に預金されることを想像してください。

このような攻撃は、事業活動を破壊し、重大な間違いやビジネスの損失を引き起こすだけでなく、さらに面倒なことに、ある程度の期間気付かれない恐れがあるため、ほとんどの組織では、情報漏洩とほぼ同様に壊滅的な被害になると考えられています。
このような攻撃による組織への影響について、調査回答者の 73% は、10 段階評価で 7 以上に評価しています。この影響がほとんど、またはまったくないとした回答者はわずか 3% でした（図 16 参照）。

アプリケーションの改竄は、情報漏洩と同様の損失を組織に与えます。実際、回答者の 4 分の 3 以上（78%）は、総損失額が $500,000 を超えると推定しました。より具体的には、損失額について、40% が $500,001 ～ $500 万、32% が $500 万～ $5,000 万と推定し、5 分の 1 弱（22%）が $500,000 以下、6% が $5,000 万を超えると推定しました（図 17 参照）

**サービス拒否攻撃（DoS 攻撃）に関連する影響**

CIA の三本柱、機密性、完全性および可用性については先ほど少し触れました。多くのセキュリティ実務者は、データの機密性と完全性は重視していますが、適切なデータを適切な時に適切な人が利用できるようにすることはあまり重視していません。サービス拒否（DoS）攻撃は、アプリケーションまたは Web サイトの可用性に直接影響するので、組織が顧客と取引する、または内部的も事業活動する能力に大きく影響します。

図18： ユーザがアプリケーションまたはデータに
アクセスできないようにするDOS攻撃の被害レベル

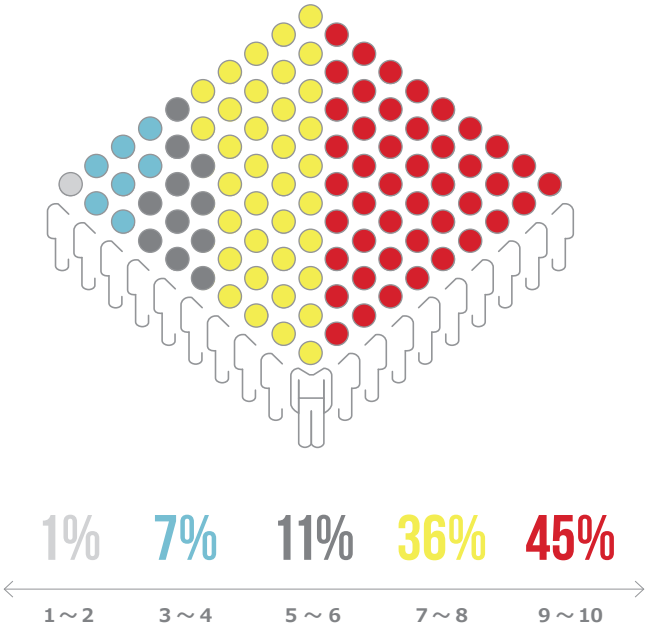
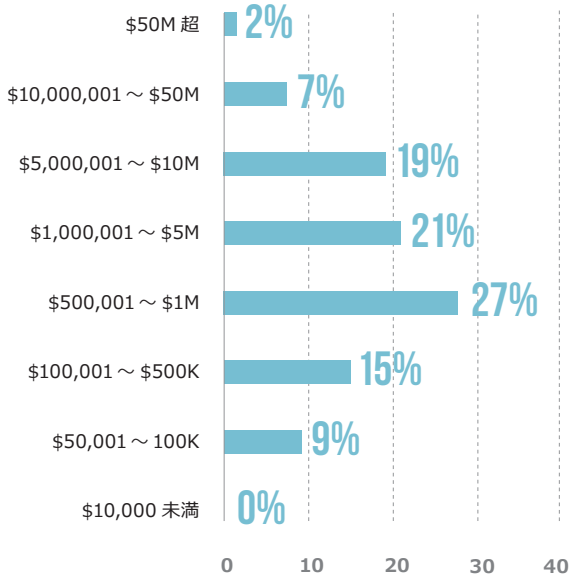



図19： ユーザがアプリケーションまたはデータに
アクセスできないようにするDOS攻撃の推定損失額




このような攻撃は、調査対象者により影響レベルのかなり上位に評価されています。可用性の損失を 10 段階評価で 7～10 に評価した回答者は 81% もいました（図 18 参照）このような攻撃の影響がほとんど、またはまったくないとした回答者はわずか 8% でした。これらの違いは、アプリケーションまたはビジネスの種類が要因となった可能性があります。大部分が静的なコンテンツのアプリケーションまたは Web サイトは、たとえば可用性がビジネスに重要である電子商取引、バンキングまたは株取引サイトよりも DoS 攻撃に対する耐性はかなり優れています。

## 可用性の損失を 10 段階評価で 7～10 に評価した回答者は 81% いました。

このような攻撃の損失額について、$50 万～ $5,000 万と推定した回答者は約 4 分の 3（74%）で、その中で最も高い割合（27%）は $500,000 ～ $100 万でした（図 19 参照）。また、$500,000 以下と推定した回答者は 24%、$5,000 万を超えると推定した回答者はわずか 2% でした。

この調査結果で明らかになりましたが、組織の大多数（概して全回答者の約 4 分の 3）は、機密情報や PII など種類を問わずデータの損失、および重要なアプリケーションの可用性の損失が、評判と損失額の両方で組織に大きな影響を与えると強く感じています。

ここで再び、考えられる最大の課題は、従業員が使用するすべてのアプリケーションを組織が把握できるかどうかです。
アプリケーションおよびそこで使用されるデータを把握できるかどうかについて、自信がない、またはある程度は自信があると答えた回答者が 62% いることから、ここでの改善が必要なことは明らかです。使用されるアプリケーションを把握すると、次の項でさらに検証する、さまざまな種類の攻撃から適切に保護する方法を決めやすくなります。


Your email has been sent!
1111 2222 3333 4444
5555
SSN
Password
Username
Username
Password
1111 2222 3333 4444
5555

# Web アプリケーション攻撃

Web アプリケーションは、「アプリケーションとは」の項で説明した構成層と下層との相互通信から形成されます。各層の規模、場所および複雑さはさまざまです。この複雑さは、これらの下層が厳しいインターネット環境と接触すると、マイナスの影響が増加することを意味します。これを示す完全な例は、2017 年 3 月[8] に発見された Apache Struts の Jakarta Multipart パーサの脆弱性です。この外部コードの下層における小さなソフトウェア バグは、突然出現して、私たちの安全に関する概念に穴を空けました。

**30%** 分析した情報漏洩 304 件のうち、Web 攻撃は 30% で断然の最上位でした。

**最上位の情報漏洩にはアプリケーション サービスが関与**

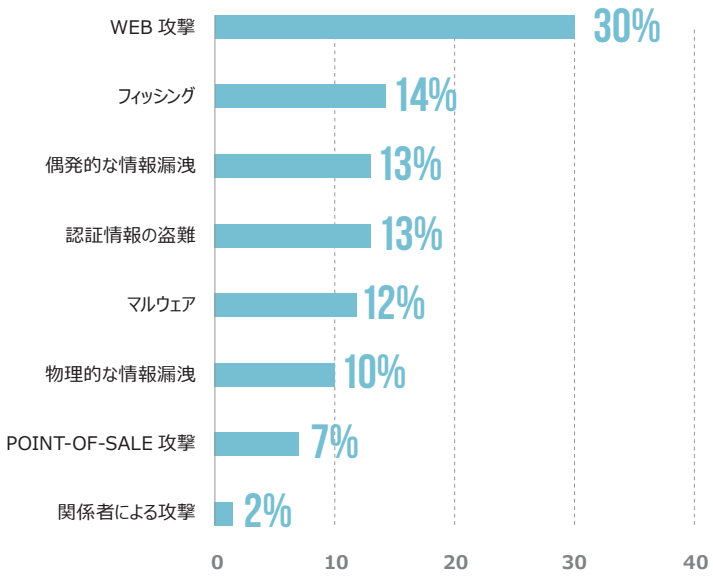
攻撃を受けるアプリケーション数は、公的な情報漏洩記録から導くことができます。本報告書では「情報漏洩」という用語を使う場合、データが盗まれたかどうかに関係なく、ネットワーク、システムまたはアプリケーションへの何らかの侵入を示します。すべての攻撃が情報漏洩というわけではありません（DDoS などは情報漏洩ではありません）。

米国のほとんどの州では、被害者への情報漏洩の通知が義務付けられていますが、Web サイトで情報漏洩に関する文書を収集および共有している州の司法長官はごく一部です。しかし、カリフォルニア州、ワシントン州、アイダホ州およびオレゴン州の人口は、米国の全人口の 16% 以上を占めているので、問題に関する洞察を提供できる十分な規模のサンプルです。

F5 Labs との研究プロジェクトの一環として、Whatcom Community College の Cybersecurity Center 学部は、2017 年および 2018 年 Q1 でのこれらの州における情報漏洩に関する文書の 1 つ 1 つを調べて、検証しました。2017 年および 2018 年 Q1 に報告された 384 件の情報漏洩のうち、304（79%）件では、分析できるだけの十分な情報漏洩の原因に対する説明がありました。残りの 21% では、情報漏洩の原因は報告されていませんでした（これらの情報漏洩の一部は、2017 年より前に発生した可能性がありますが、報告されたのは最近です）。

これらの 304 件の情報漏洩を分析して、Web 攻撃、フィッシング、認証情報攻撃、Point-of-Sale 攻撃、物理的および偶発的な情報漏洩、マルウェア（情報漏洩に関する文書での分類）、関係者による情報漏洩に分類しました。この中で圧倒的 1 位は、30% の Web 攻撃で、その後に、フィッシング（14%）、認証情報ハッキング（13%）、偶発的な情報漏洩（13%）が続きます（図 20 参照）。

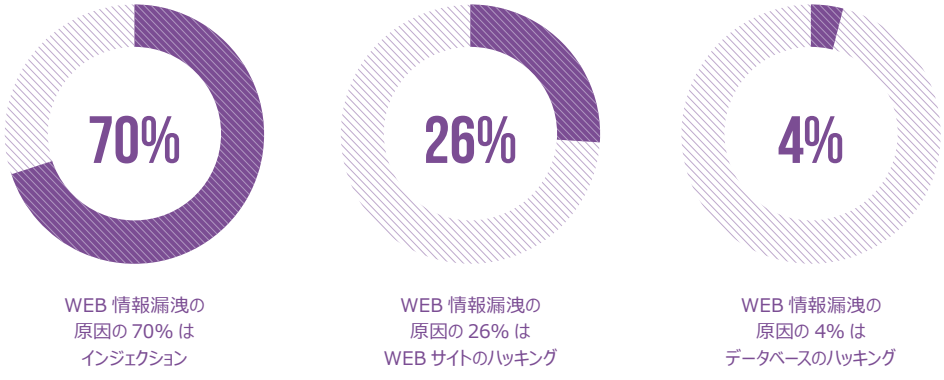
図 20：初期攻撃の種類別に示すアプリケーション情報漏洩（2017 年 WA、OR、ID、CA）




## アプリケーション攻撃

アプリケーション攻撃は、57 件（30%）の情報漏洩において主な報告済みの情報漏洩の原因でした。アプリケーション情報漏洩は最大の脅威なので、このデータは興味深いものです。具体的なアプリケーション情報漏洩には、Web インジェクションによるクレジット カードの盗難（70%）、Web サイトのハッキング（26%）およびデータベースのハッキング（4%）がありました。

図 21：根本原因別の情報漏洩（2017 年 WA、OR、ID、CA）



## WEB アプリケーション サービスに関連するエクスプロイト

アプリケーション セキュリティに取り組む前に、まず最小の脅威に対処する必要があります。インターネット上の誰にとっても、この最小の脅威は「スクリプトキディ」です。スクリプトキディは、独自のスクリプトを作成するだけの技術的な知識はないものの、インターネット上で公開されているクリックするだけの簡単なエクスプロイトを利用してアプリケーションに侵入できます。その能力を理解するには、スクリプトによる操作が可能で簡単に使えるエクスプロイト キットにどのようなものがあるか監視する必要があります。このために、F5 は、Exploit-DB で公開されているすべての Web アプリケーション エクスプロイトを分析しました。[9] この Web サイトで公開されているスクリプトは、誰でも自由に利用できるので、これがこの必要最小限の脅威能力であり、最も一般的な脅威の基準です。

Web 攻撃に関する Exploit-DB の記録のすべてを調べた結果、エクスプロイトの 69% は、幅広く使用されている Web 開発スクリプティング言語である PHP を対象としていることが分かりました。ハードウェアベースのデバイス（IoT）のエクスプロイトはその次に多く 10% でした。図 22 に示すように、私たちは、PHP エクスプロイトが何をするか詳しく検証し、分類しました。

PHP を介した SQL インジェクションは、最も普及しているエクスプロイト スクリプトにおいて 46% で圧倒的な 1 位でした。これは、PHP ベースの Web アプリケーションに対する SQL インジェクション攻撃のリスクに大きく寄与しています。

その他の PHP 以外のエクスプロイトもすべて調べ、図 23 に示すように分類しました。PHP 以外のエクスプロイトでは、13.3% のクロスサイト スクリプティング（XSS）と 9.6% のクロスサイト リクエスト フォージェリ（CSRF）が上位で、その後にそれほど差がなく 9.1% で認証バイパスが続きます。知識のある攻撃者がこれらの 3 つのすべてのエクスプロイトを利用することで、ユーザになりすました敵が Web アプリケーションに不正アクセスできます。そのため、PHP 以外の Web サイトでは、アクセスを保護することを最優先すべきです。

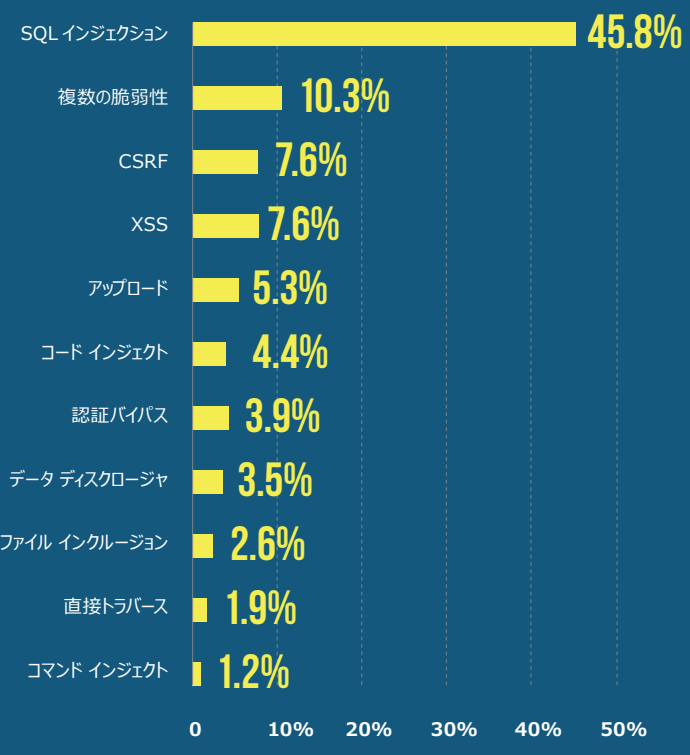



図 22：EXPLOIT-DB PHP エクスプロイトの分類

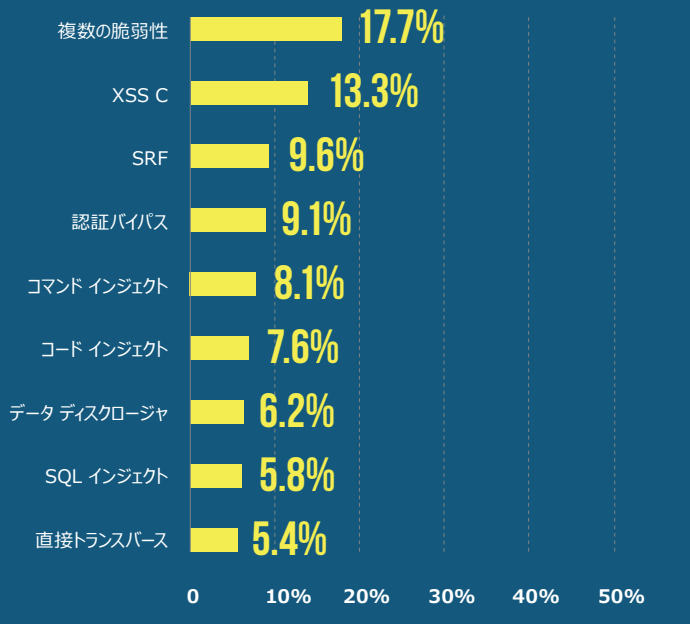
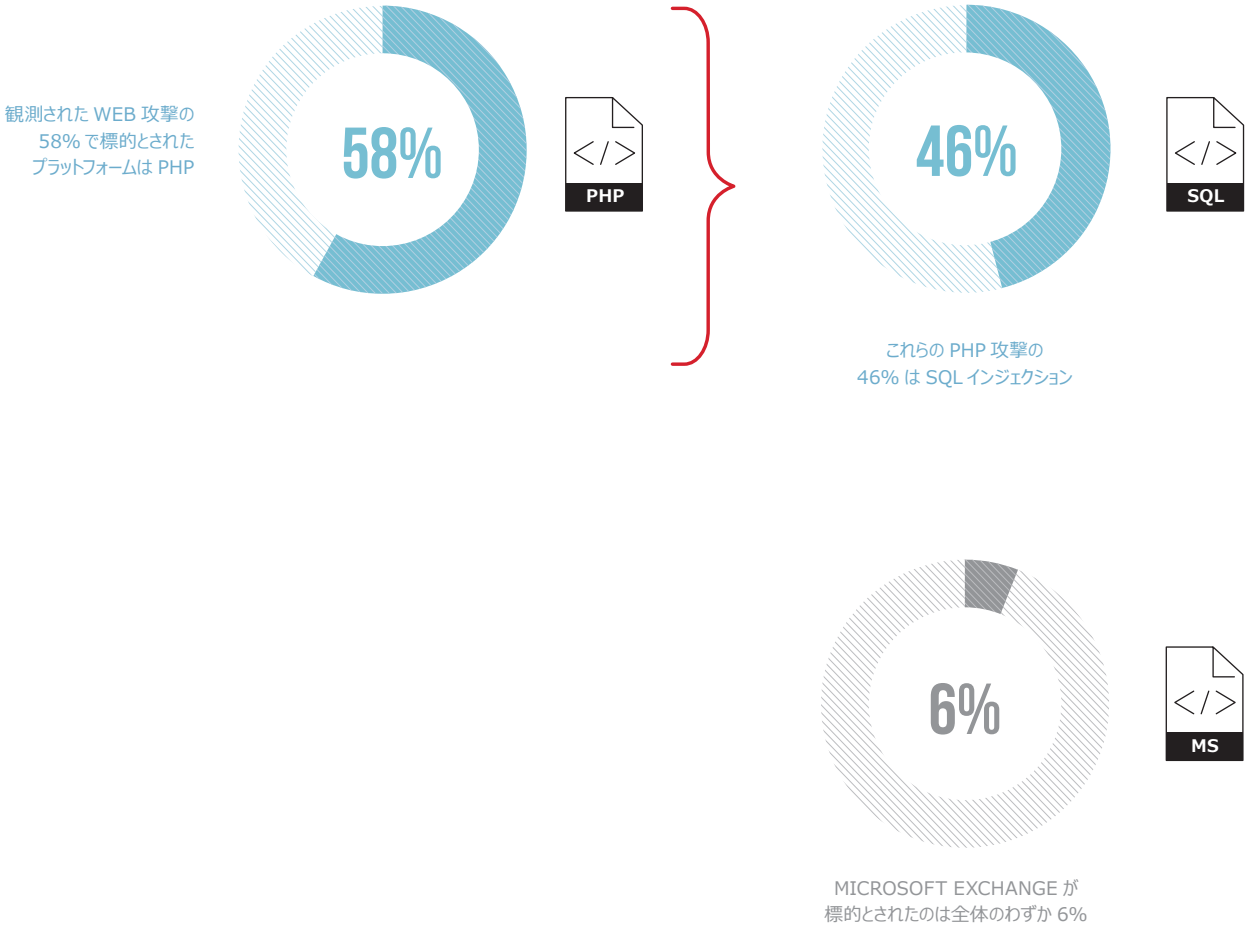



図 23：EXPLOIT-DB PHP エクスプロイト以外の分類

図 24: 上位 3 つの侵入攻撃の標的

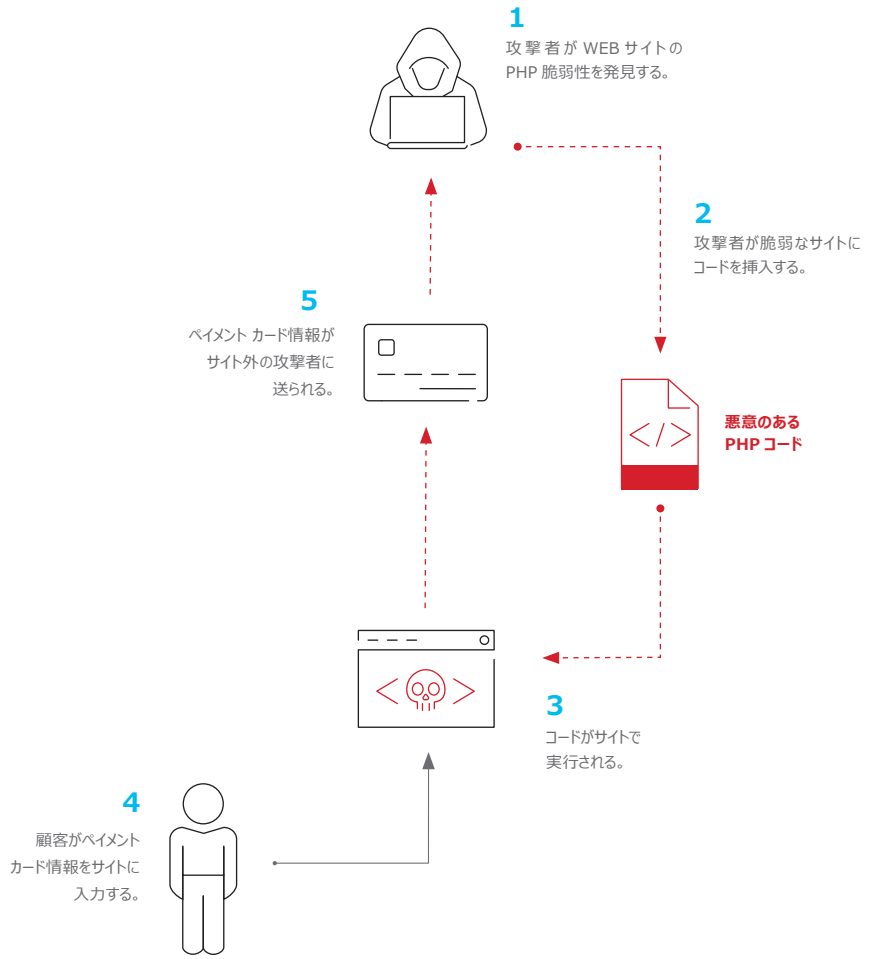



## 上位の攻撃にはアプリケーション サービスが関与

私たちは、データ パートナである Lorykaと協力して、2017 年に 21,010 のネットワークで発生した Web 攻撃から収集した世界中の侵入およびハニーポット データを検証しました。

このうち 58% では、PHP が標的とされ、これらの攻撃の 46% は SQL インジェクションでした。Microsoft Exchange が標的とされたのは全体のわずか 6% でした。標的とされたすべてのプラットフォームで、攻撃の 34% は SQL インジェクションでした。

図 25：一般的なインジェクション攻撃の経路



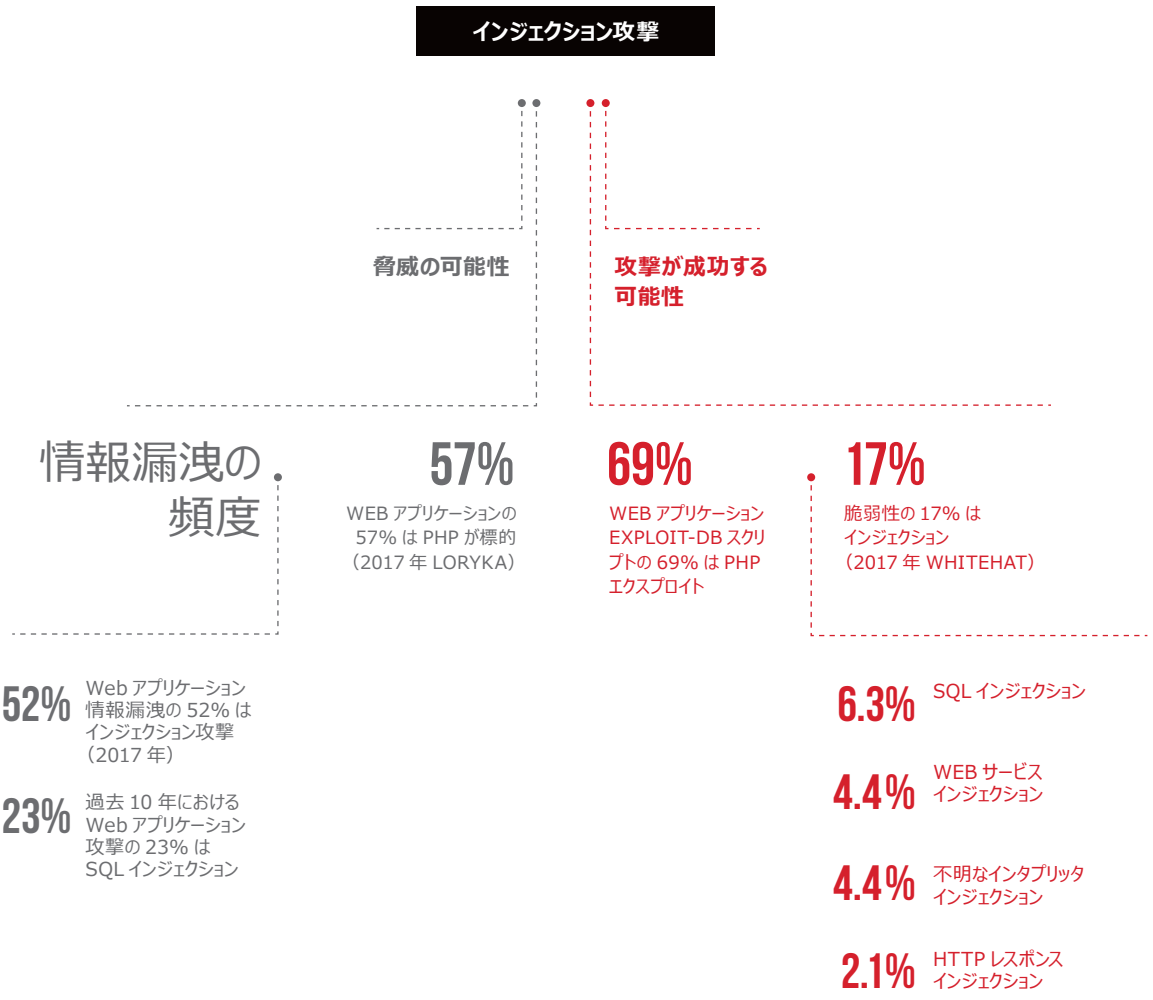

非常に人気のある PHP はハッカーにとって格好の標的です。PHP のセキュリティ ホールを見つければ、数百万のサイトに不正アクセスできます。

## インジェクション攻撃

インジェクション攻撃は、2017 年で最も重大な Web アプリケーション情報漏洩でした。これは、過去数十年のインジェクション攻撃で使用されている有名なショッピング カートの脆弱性の特定のセットに関与したサイバー犯罪者によるマルウェア キャンペーンが原因として考えられます。[10]

サイバー犯罪者は、まず、Web サイトのソース コードを変更できる電子商取引 Web サイトの脆弱性を見つけます。次に、このサイトにコードをインジェクトして、顧客のペイメント カード情報を無許可でコピーします。図 25 に、Point-of-Sale カード盗難マルウェアの Web アプリケーション バージョンを示します。[11] 攻撃は非常に高度で、マルウェアが別のサイトから提供され、データが別のリモート サイトに保存され、有効な HTTPS 証明書で会話全体が暗号化されます。

図 26：インジェクション攻撃の可能性と
その成功の可能性について



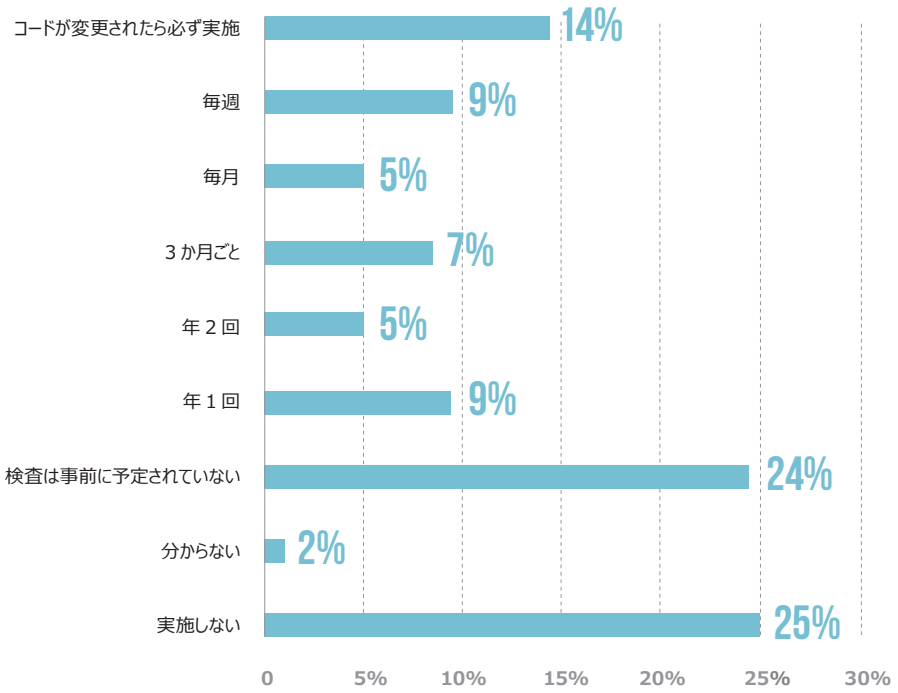

Web アプリケーション セキュリティ リスクの最も有名なリストは、アプリケーションがどのように不正アクセスされるかをまとめ、ランク付けした OWASP Top 10[12] です。2017 年、この Top 10 リストは今までで最もデータに基づいたバージョンで、業界専門家およびアプリケーション セキュリティ会社を見直し、Web アプリケーション リスクの悪用可能性、普及および検出可能性について調査しました。当然のことですが、そこでも、インジェクション攻撃が最も優先順位の高いリスクになりました。

当然のことですが、2017 年の OWASP TOP 10 リストでも、インジェクション攻撃が最も優先順位の高いリスクになりました。

これらの攻撃の被害者には 2 種類あります。顧客からペイメント カード情報を収集するコードを使用する電子商取引サイトの運営企業と、その顧客です。アプリケーションの簡単に不正アクセスされるショッピング カート コードは、OWASP リスク A9「既知の脆弱性があるコンポーネントの使用」に分類されます。[13] これらの攻撃から、Web アプリケーションでのセキュア コンポーネントの使用、脆弱性の検査およびサイト コードの完全性の監視が重要であることが分かります。

インジェクションの欠陥の大多数は、外部ライブラリにあります。つまり、通常はこれらの欠陥に対するパッチを利用できます。しかし、これらのセキュリティ ホールを見つけて、パッチを適用することで、すべてが完全になるわけではありません。WhiteHat Security によると、2017 年に発見された脆弱性の 8.2% は「パッチ処理されていない」コード ライブラリでした。

図 27: 組織が WEB アプリケーション脆弱性を検査する頻度




**60%**

回答者の 60% は、最新の WEB アプリケーション セキュリティへの対策が極めて不適切でした。

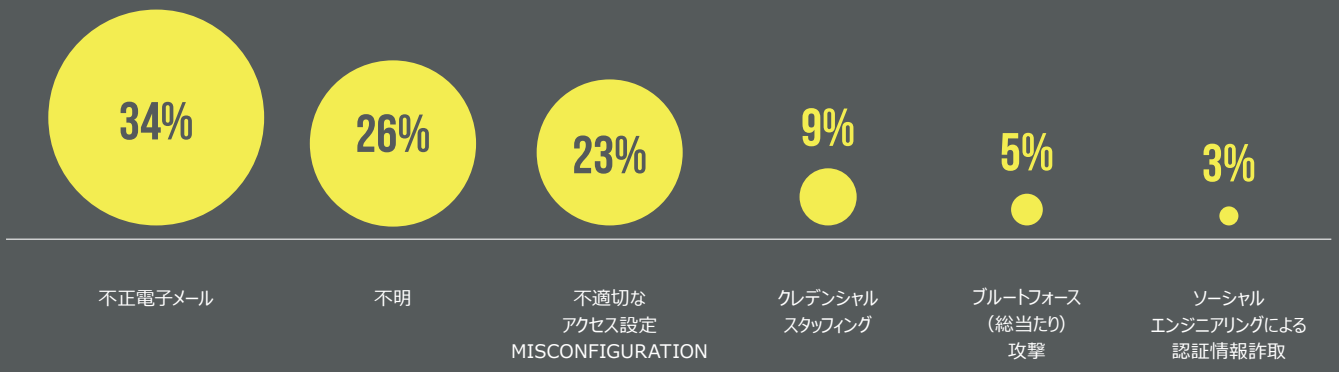
## 2017 年最悪の攻撃だった Equifax での Apache Struts 攻撃は、サーバ側テンプレートのインジェクション脆弱性でした。

F5 Ponemon セキュリティ調査で、組織が Web アプリケーションの脅威および脆弱性をどのくらいの頻度で検査しているかを調べましたが、その結果は不安になるものでした。回答者の 60% は、Web アプリケーションの脆弱性の検査は、「実施しない」、「 事前に予定していない」、「どのくらい実施しているか分からない」、または「年 1 回のみ」と答えていました。これら 4 つのすべての状況は、最新の Web アプリケーション セキュリティにおいて極めて不適切です。

図 26 に戻り、PHP はなぜ Web サイト ハッカーにこれほど狙われているのでしょうか。その理由の 1 つは、非常に人気があることです。そのため、セキュリティホールを見つければ、数百万のサイトに不正アクセスできる PHP は、攻撃者の格好の標的です。さらに重要なことは、PHP はアプリケーション セキュリティの経験不足の初心者プログラマに人気があるため、脆弱な Web サイトが大量に生まれていることです。PHP は、ディレクトリ構造で php 拡張子のファイルを探します。そのため、新しい PHP ファイルをアップロードすることで、新しいコードを簡単にインジェクトできます。PHP プログラマは OWASP PHP Security Cheat Sheet[14] を確認して、定期的にパッチをシステムに適用することをお勧めします。

2017 年最悪の攻撃だった Equifax での Apache Struts 攻撃は、サーバ側テンプレートのインジェクション脆弱性でした。[15]Apache Struts は、ユーザ提供のデータなどのデータ入力に基づいてテンプレートから動的な Web ページを生成するときに使用されるオープンソースのフレームワークです。この特別な脆弱性により、攻撃者は、コマンドを Web アプリケーションにインジェクトして、システムを乗っ取ることができました。そのため、適切な入力だけをアプリケーションに許可するユーザ入力の無害化が、インジェクション攻撃を防ぐ上で重要です。

図 28：不正アクセスの入手方法




## アカウント アクセスのハイジャック

アプリケーションの鍵となるのはアクセス認証情報です。攻撃者は、盗まれた認証情報を入手するか、処理中のログインをハイジャックすると、ユーザに完全になりすますことができます。

ボットネットは、アプリケーション アクセスに対する攻撃の大多数をオーケストレートします。ボットネットは、これまでは家庭用コンピュータで構成されていましたが、現在では主に IP カメラ、テレビおよび家庭用インターネット ルータなどの IoT デバイスで構成されています。これらのボットネットの船団は、アカウントへのブルートフォース（総当たり）攻撃、盗んだパスワードのテスト、またはアクセス制御の弱い Web アプリケーションの検出に使用できます。私たちが検証した 2017 年および 2018 年 Q1 の情報漏洩通知書に従い、図 28 に、報告されているアクセス認証情報の取得方法をいくつか示します。

認証情報は、XSS、Man in the Browser 攻撃、中間攻撃、マルウェアおよびフィッシング攻撃によりユーザから直接盗むこともできます（図 29 参照）。CSRF 攻撃は、処理中のユーザ セッションをハイジャックして、不正コマンドをサイトに直接インジェクトすることもできます。

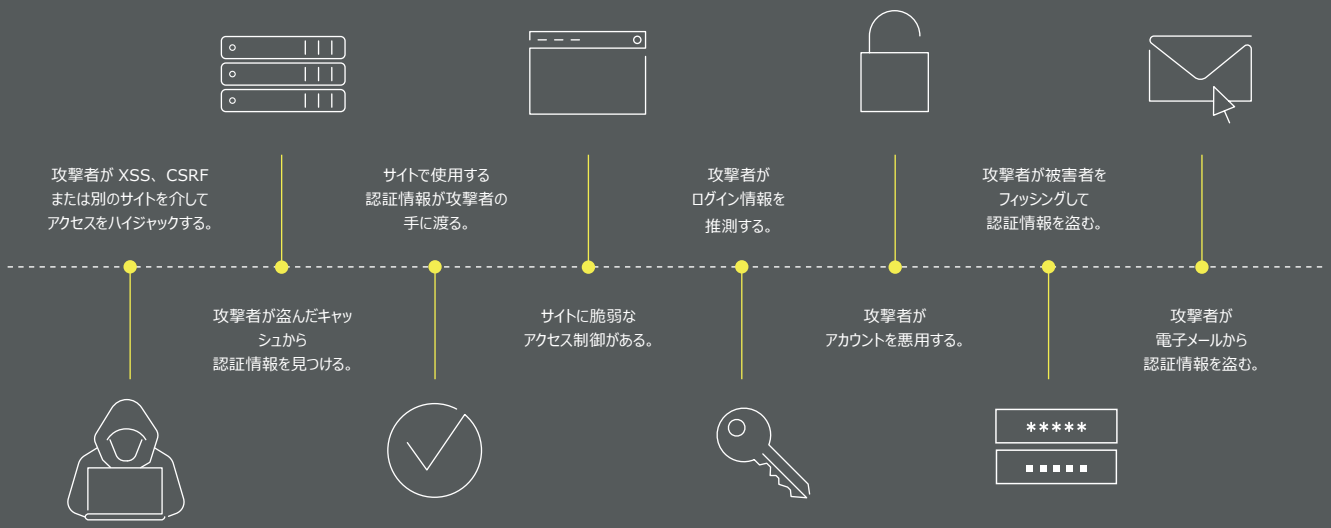
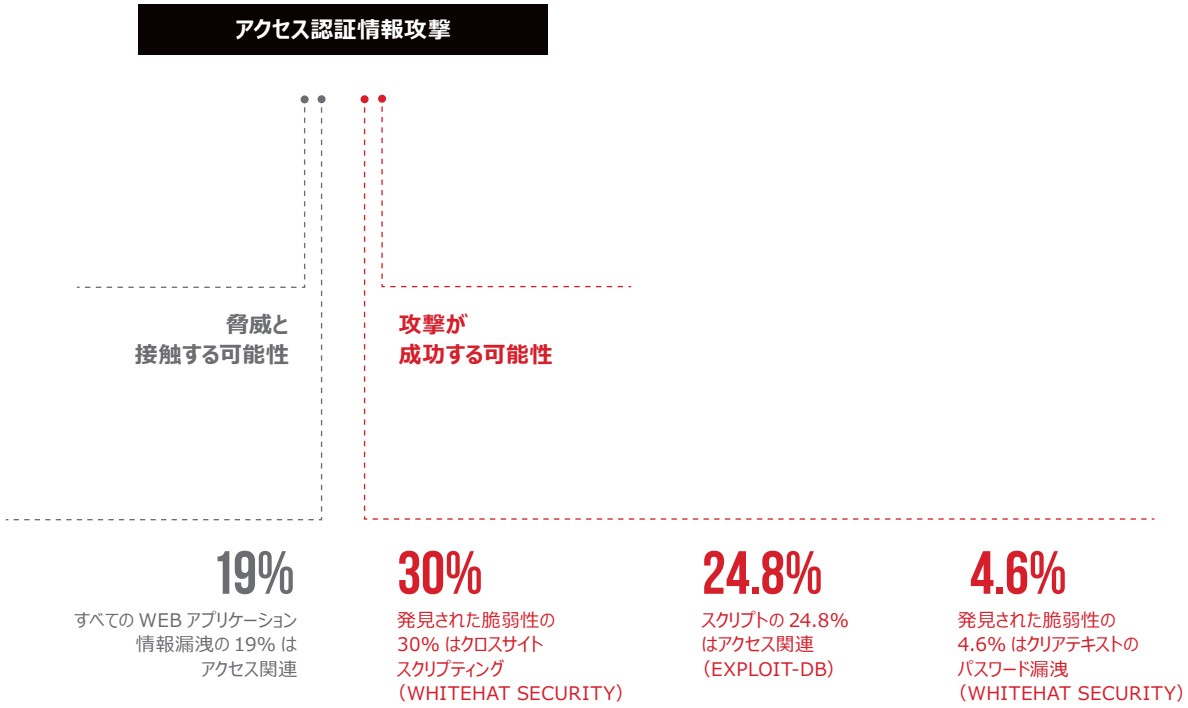



図 29：その他の一般的なユーザ攻撃の経路

図 30 : アクセス認証情報攻撃の可能性とその成功の可能性について




攻撃者が盗んだアクセス認証情報を使用して
正規ユーザになりすます場合、不正アクセスが行われても
気付かれないという厄介な副作用があります。

攻撃者が盗んだアクセス認証情報を使用して正規ユーザになりすます場合、不正アクセスが行われても気付かれないという厄介な副作用があります。この攻撃は、発生のかなり後、通常は被害者が自分の口座での不正取引を報告した後で発見されることがよくあります。カリフォルニア州、ワシントン州、アイダホ州およびオレゴン州の司法長官に報告されたアクセス攻撃では、不正アクセスが大規模な情報漏洩に発展していました。しかし、これらの統計情報は氷山の一角に過ぎません。情報漏洩が報告された事象は大規模ですが、特異な不正取引事件は公開されていません。個々のユーザが一度限りでアクセスをハイジャックされることもあるので、実際の情報漏洩の件数は、私たちが知っているよりも遥かに多い可能性があります。

図 30 は、アクセス攻撃のリスクについてのデータです。Exploit-DB スクリプトの 24.8% はアクセス関連です。WhiteHat Security により発見された脆弱性の 30% は XSS、4.6% はクリアテキストのパスワード漏洩で、これらはいずれもユーザ認証情報の盗難につながる可能性があります。すべての Web アプリケーション情報漏洩の 19% はアクセス関連です。

多くの Web アプリケーションでは、不正アクセスまたは盗まれた認証情報を検出および拒否できる洗練されたアクセス制御メカニズムが使用されていません。最悪のケースでは、アクセス制御が不適切に設定されていて、アプリケーションおよびデータベースがほぼ、または完全に無防備でさらされていました。不適切なアクセス制御設定は、OWASP Top 10 のリスク A5「不適切なセキュリティ設定」に当てはまります。[16]

# 回答者の 75% は、重要な WEB アプリケーションへのアクセスに、各アプリケーションに固有なユーザ名とパスワードの認証情報を使用しています。

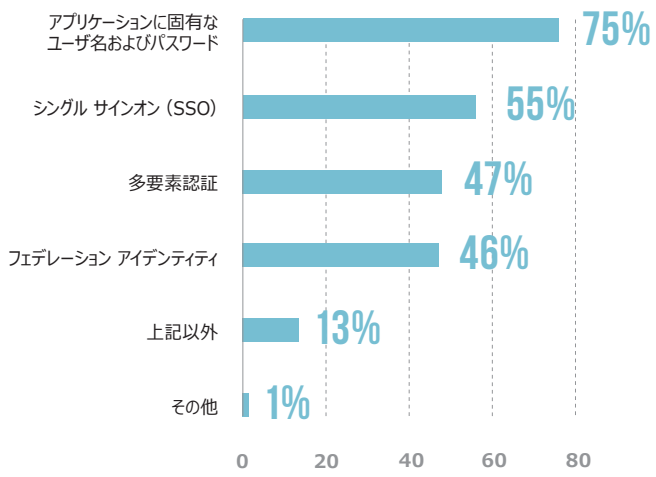



図 31：重要なアプリケーションへのアクセスの管理方法（複数回答可）

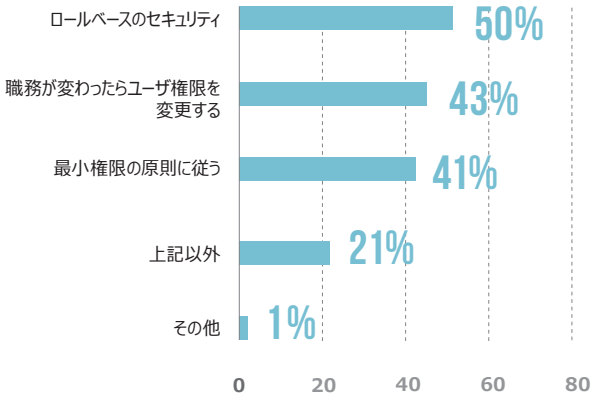
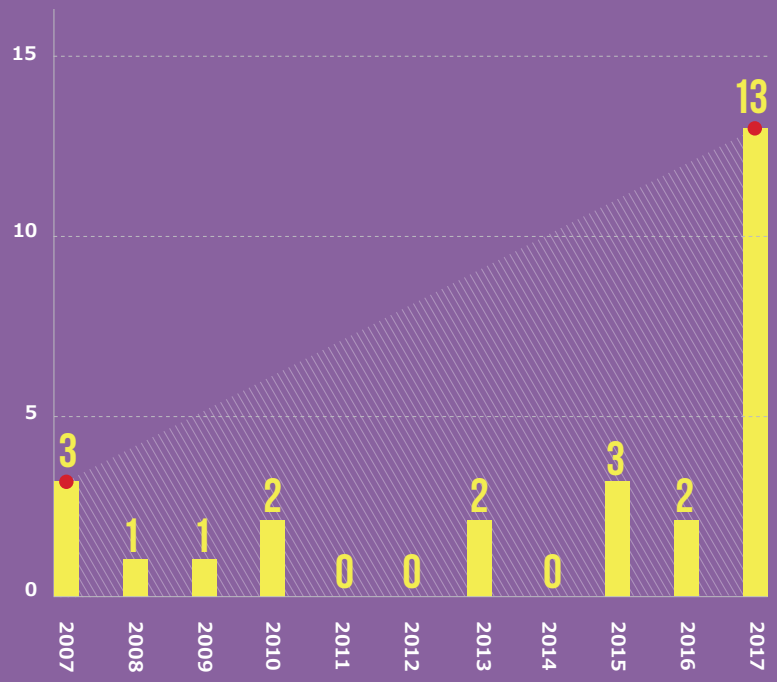



図 32：実施しているアクセス制御モデル

セキュリティの大半と同様に、アクセスは、技術プロセスでもあり、操作プロセスでもあります。組織がアプリケーションへのアクセスをどのように管理しているかについて、F5 Ponemon セキュリティ調査では、組織内で使用される重要な Web アプリケーションへのアクセスを組織がどのように認証しているか調べました。圧倒的大多数（75%）が各アプリケーションに固有なユーザ名とパスワードの認証情報を使用していました（図 31 参照）。また、55% はシングル サインオンを使用していました。回答者の約半数（47%）は二要素 / 多要素認証を使用し、46% はフェデレーション アイデンティティを使用していました。14% は、その他の認証方法を使用しているか、認証をまったくしていませんでした。

また、同調査では、組織が Web アプリケーションの使用およびアクセスをどのように許可しているか調べました。ここではやや安心できる結果で、多くのセキュリティ チームが、一般的な運用対策を実施していました。回答者の 50 パーセントはロールベースのセキュリティを使用し、41% は「最小権限」の原則に従い、職務に最低限必要なアクセスをユーザに提供していると答えています（図 32 参照）。また、43% は、役職が変わったらユーザの権限を変更すると答えています。しかし、その一方で、22% もの回答者が、他のアクセス制御方法を使用しているか、アプリケーションへのユーザ アクセスをまったく制御していていません。




デシリアライゼーション攻撃は 10 年以上前から存在していましたが 2017 年に急増しました。

図 33: 過去 10 年のデシリアライゼーション エクスプロイト

## デシリアライゼーション攻撃

デシリアライゼーション攻撃はこれまでは多くはありませんでしたが、最近増え始めていて、壊滅的な被害を与える可能性があります。2017 年、これらの攻撃は、別の Apache Struts 脆弱性とともに、デシリアライゼーションにより促進されるコマンド インジェクション攻撃として大きな不安をもたらしました。[17] これらの攻撃は、図 33 に示すように、10 年前から存在していましたが、2017 年に急増しました。

アプリケーションは、それぞれが異なるコンポーネントを提供するサービスの集合体として導入されることが普通なので、相互作用するための通信方法が必要です。通常、この通信は HTTP を介して行われますが、データの形式も重要です。シリアライゼーションは、アプリケーションがそのデータを一般的にはサーバから Web ブラウザ、Web ブラウザからサーバ、または API を介したマシン間におけるトランスポート用の形式(通常はバイナリ) に変換するときに発生します。Java は、トランスポート用にほとんどのオブジェクトをシリアライズし、Remote Method Invocation (RMI) または Java Management Extensions (JMX) などの形式で埋め込まれたライブラリを使用します。

図 34：デシリアライゼーション攻撃の経路

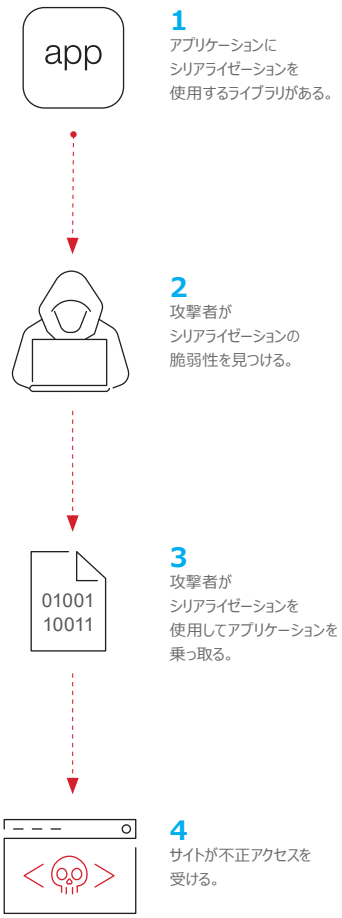
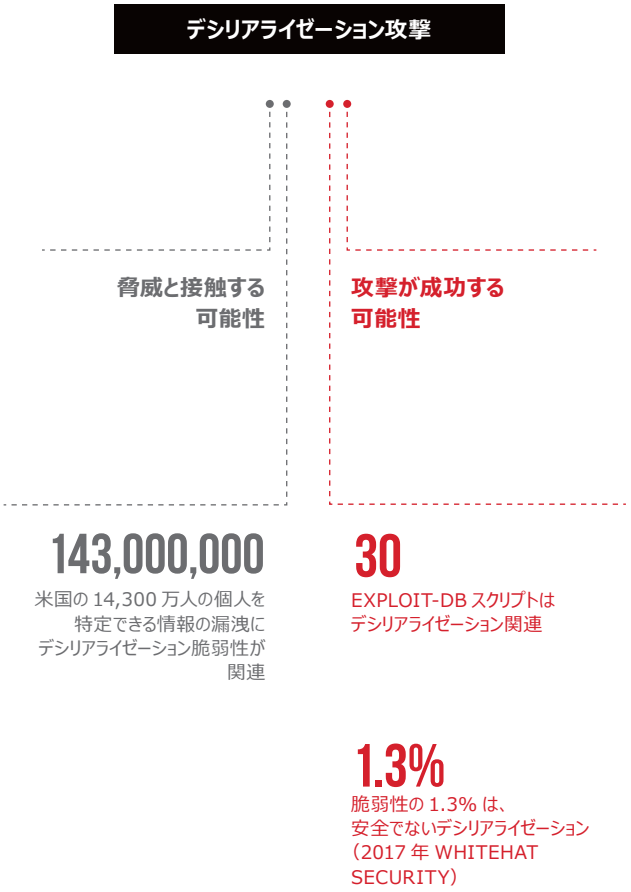



図 35：デシリアライゼーション攻撃の可能性と
その成功の可能性




攻撃者は、シリアライズされたデータ ストリームの既存のパラメータにコマンドを埋め込む（Apache Struts の場合と同様）、またはこれを改竄できます。アプリケーションがフィルタ [18] またはチェックなしでデータ ストリームをデシリアライズする場合、これらの攻撃はアプリケーションの中心に直接侵入できます。Apache Struts 脆弱性の場合では、攻撃者は、XStream プロセスを実行するサーバの範囲内で任意のコードまたはシェル コマンドを実行し、アプリケーション自体に完全アクセスしました。

セキュリティの基本的な考え方は、信頼できないソースからのデータがないか常にスキャンおよび検査することです。ファット クライアント用に設計された多くのアプリケーションは、シリアライゼーションを使用して、データをアプリケーションに提供します。しかし、Web クライアントは、信頼できるソースではありません。ブラウザで実行される内容が変更され攻撃コードが含められる可能性があります。

このような攻撃は、2017 年に OWASP Top 10 リスク A8「安全でないデシリアライゼーション」として追加された、今後も警戒が必要な脅威です。[19] これは、業界調査に基づいて実施されているので、かなりの数の Web アプリケーションの専門家がこれを重要な問題と見なしていることが分かります。

図 35 に示すように、デシリアライゼーションは、数は多くはありませんが、常に警戒が必要な新しい脅威です。


デシリアライゼーションは、
2017 年 OWASP TOP 10
に追加された新しい脅威です。
143,000,000
米国の 14,300 万人の個人を特定できる情報の漏洩に
デシリアライゼーション脆弱性が関連していました。

## アプリケーションに対する持続的標的型攻撃

持続的標的型攻撃（APT）は、カスタム エクスプロイトの偵察、テストおよび準備のための時間を必要とする攻撃です。標的が敵にとってどのくらい価値があるかによっては、このような攻撃はめったにありません。一般的な例は、Web ベースの電子メール システムを攻撃して反体制派の電子メールの内容を探る国家諜報機関です。[20] しかし、APT は、強力な影響を与え、検出が困難なので、十分考慮に値する脅威です。

**76%**

2017 年 WhiteHat Security が発見した脆弱性の 76% は、機能の悪用に分類されました。

### 機能の悪用

機能の悪用は、ビジネス ロジック攻撃とも呼ばれ、アプリケーション デザイン上の欠陥を巧みに操り、アプリケーションの意図された機能以上のことを実行します。これらの攻撃の影響は、アプリケーションの機能および目的により大きく異なります。以下に、機能の悪用の典型的な例をいくつか示します。

- 買い物客が購入金額の返金だけでなく購入商品も受け取れるようにショッピング アプリケーションを操作する
- パスワード リセット メカニズムを妨害してパスワードを検証する
- サイトのファイル アップロード添付機能を使用して、マルウェアをアップロードする
- ソースを消費する機能を繰り返し実行して、サービス拒否攻撃（DoS 攻撃）によりアプリケーションを完全に停止させる

2017 年 WhiteHat Security が発見した脆弱性の 76% は、機能の悪用に分類されました。このような脆弱性は、自動化された方法では検知が難しいため、WhiteHat Security は分析において手作業による侵入検査も行っています。それでも、このような攻撃の普及と影響は、攻撃者の工夫により左右されます。

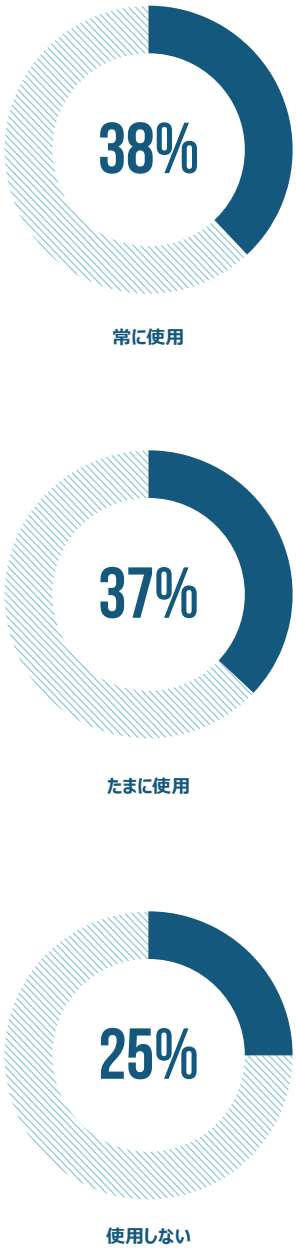



図 36：追加の API アクセス許可手順の使用

### アプリケーション プログラミング インターフェイス攻撃

アプリケーションがデータを受け取る場合、通常はアプリケーション プログラミング インターフェイス（API）を経由しますが、これは攻撃の標的となる可能性があります。アプリケーション所有者によっては、API は人間が直接触れる必要がないので攻撃者には見えないと考えています。しかし、API は、攻撃者の偵察スキャンにより簡単に見つかり、従来のほとんどの Web アプリケーション攻撃方法により攻撃されます。API は、アプリケーション内の管理機能を備え、価値あるデータ ストアに直接アクセスできることが多いので、特に狙われています。

## API は、アプリケーション内の管理機能を備え、価値あるデータ ストアに直接アクセスできることが多いので、特に狙われています。

API は、マシン間の対話を目的としているので、通常のユーザ インターフェイスとは認証方法が異なる場合があります。メインのユーザ ログイン ページの認証は強化されているものの、API の認証はそれほど強化されていないことがあります。一般的に見落とされるのは、変更されることがない、または管理が不十分な単一のパスワードまたは暗号キーを使用する API 認証です。最悪の場合、組織全体のアプリケーションへの API アクセスに単一の共有秘密キーが使用されています。GitHub の最も一般的なエラーは、アプリケーション プログラマがソース コードに API キーを誤って含めてしまうことです。GitHub が API キーを定期的にスキャンすることは普通のことです。実際、API はアプリケーションおよびそのデータに自由にアクセスできるので、アクセス制御はユーザよりも API を強化する必要があります。

F5 Ponemon セキュリティ調査では、追加の API 認証を使用しているか回答者に質問しました。回答者の 25% は「使用しない」と答えましたが、「たまに使用」または「常に使用」と答えた回答者が 75% いたことは安心できる結果でした。


TOP SECRET

# アプリ インフラストラクチャへの攻撃

攻撃者は、アプリケーションに侵入する、またはアクセス認証情報を盗むことが困難な場合はアプリケーションのサポート インフラストラクチャを攻撃すればよいことを心得ています。インフラストラクチャ攻撃を原因とする実際の情報漏洩の数は、アプリケーション自体に対する攻撃より少ないですが（通常はアプリケーション攻撃を成功させる方が簡単なため）、アプリケーション インフラストラクチャへの攻撃が簡単になる重大な状況があります。

**25.4%**

トランスポート層の不十分な暗号化が、報告されている動的検査脆弱性の 25.4% を占めていました。

ここでは、暗号化、証明書、Domain Name Service（DNS）およびネットワーク層などがまとまった、アプリケーションを支えるさまざまな層に対する数多くの攻撃について検証します。

WhiteHat Security の脆弱性データでは、トランスポート層の不十分な暗号化が、報告されている動的検査脆弱性の 25.4% を占めていました。さらに厄介なことに、図 37 に示すよう、2017 年の脆弱性は上昇傾向にあります。しかし、トランスポート層の不十分な保護だけが問題というわけではありません。暗号化情報漏洩と見られたいくつかの大きな情報漏洩は、実際には、DNS ハイジャック攻撃でした。これらの攻撃についても詳しく調べました。

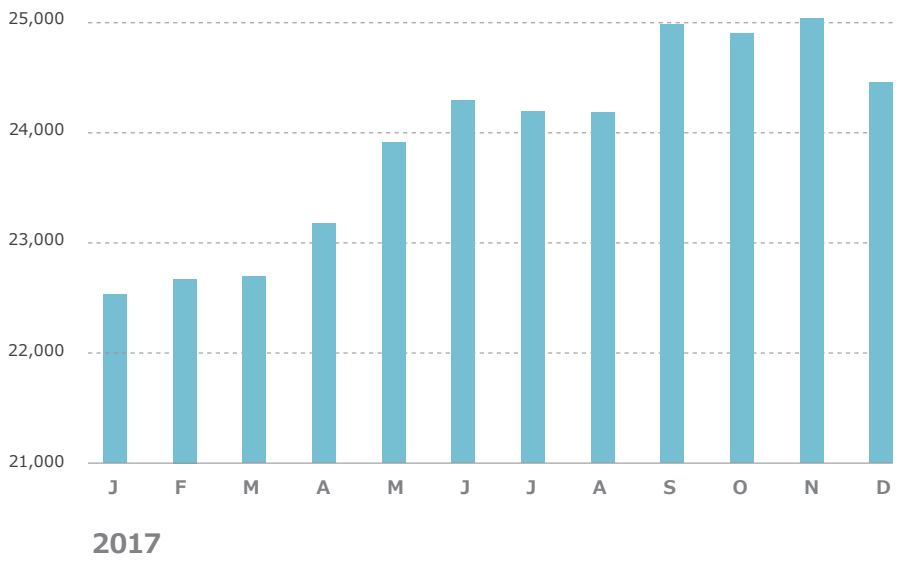


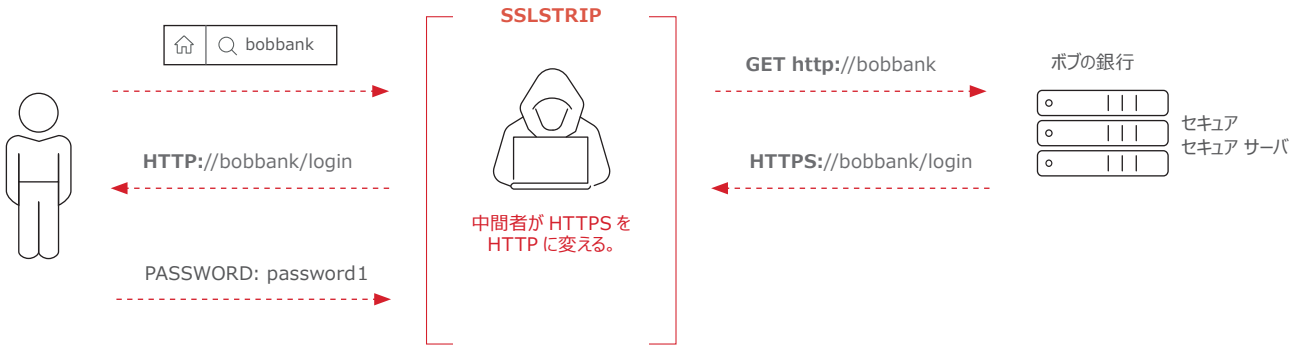

図 37：2017 年に見つかった月別のトランスポート層の不十分な暗号化

## トランスポート層に対する攻撃への保護 PROTECTION

最も一般的なトランスポート層保護は、Transport Layer Security（TLS）プロトコルとその元となる Secure Sockets Layer（SSL）です。これらは、従来 VPN に使用されていた IPsec をずっと以前から上回る事実上の世界的なトランスポート セキュリティ プロトコルです。

暗号化がない場合、ネットワークは、信頼できないネットワーク（インターネットや公衆無線ネットワークなど）を通過するときにスパイ活動やデータ改竄を受けやすくなります。最も強力な攻撃は、中間者攻撃（MitM）です。この攻撃では、攻撃者は、すべてのプライバシー情報を抜き取り、被害者になりすまし、まったく気付かれずにデータを変更できます。図 38 に、SSLStrip[21]v という MitM 攻撃により被害者(この例ではボブ）がどのように騙されるかを示します。この場合、被害者は、目的のサーバ（ボブの銀行）との暗号化された接続を確立できなくなり、攻撃者は、ボブの銀行ログイン認証情報を盗むことができます。

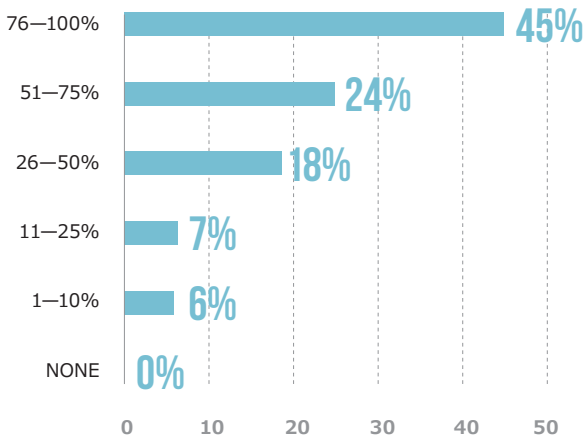
図 38：SSLSTRIP 中間者攻撃の経路




影響はそれほど大きくはありませんが、迷惑で厄介な攻撃が、暗号化されていない Web セッションでの広告インジェクションです。報告によると、これは、空港 [22] やホテル [23] の無線ホットスポット、およびインターネット サービス プロバイダ [24] で発生しています。より積極的な MitM 攻撃であれば、危険なマルウェアまたはコード インジェクションを取り込むことができます。

トランスポート層保護において「不十分な保護」とは何を意味するのでしょうか。最悪の場合、トランスポート層保護をまったく使用しないことを意味します。調査参加者に、Web アプリケーションで暗号化を使用する割合について質問しました。回答者の半数弱（45%）は、Web アプリケーションのほとんど（76% ～ 100%）で TLS/SSL を使用していると答えました（図 39 参照）。

図 39：SSL または TLS を使用するアプリケーションの割合



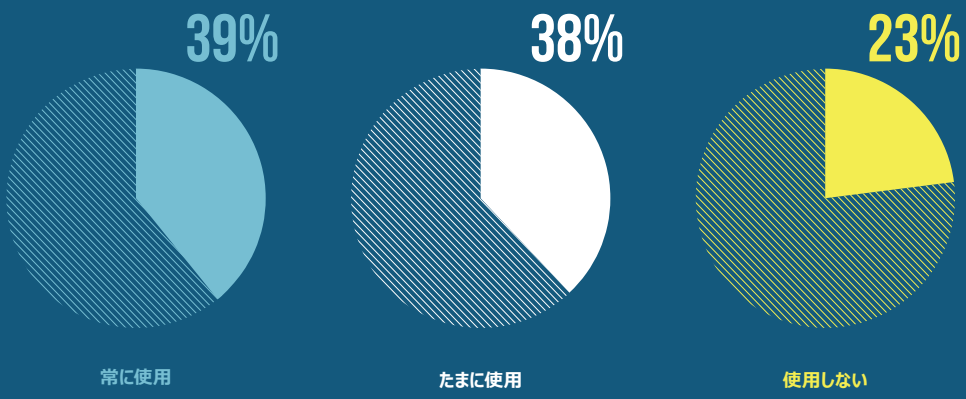



図 40：送信中のデータおよびアプリケーションに暗号化を使用している組織

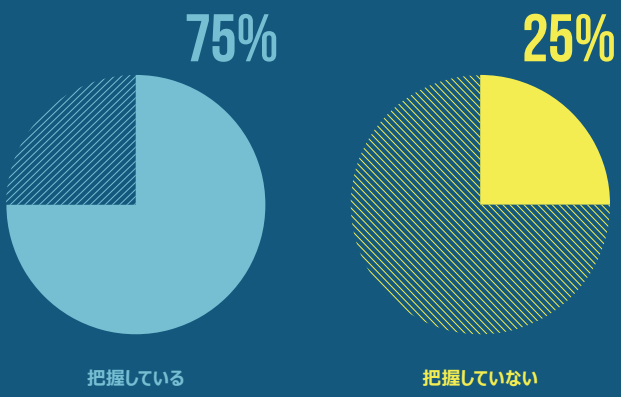



図 41：暗号化のレベルを把握している組織の割合

送信中のデータの暗号化について、回答者の 77% は、「たまに使用」、「ほとんど使用」または「常に使用」のいずれかと答え、23% は「使用しない」と答えました（図 40 参照）。

暗号化を使用しない場合に続く次の最悪なケースは、古くて脆弱な暗号または短い暗号化キーを使用することです。トランスポート層暗号化を強化する第一段階は、最新の標準と比較できるように、使用している暗号化キーを把握することです。F5 Ponemon セキュリティ調査の結果、4 分の 1 の組織が使用している暗号化キーを把握していませんでした。

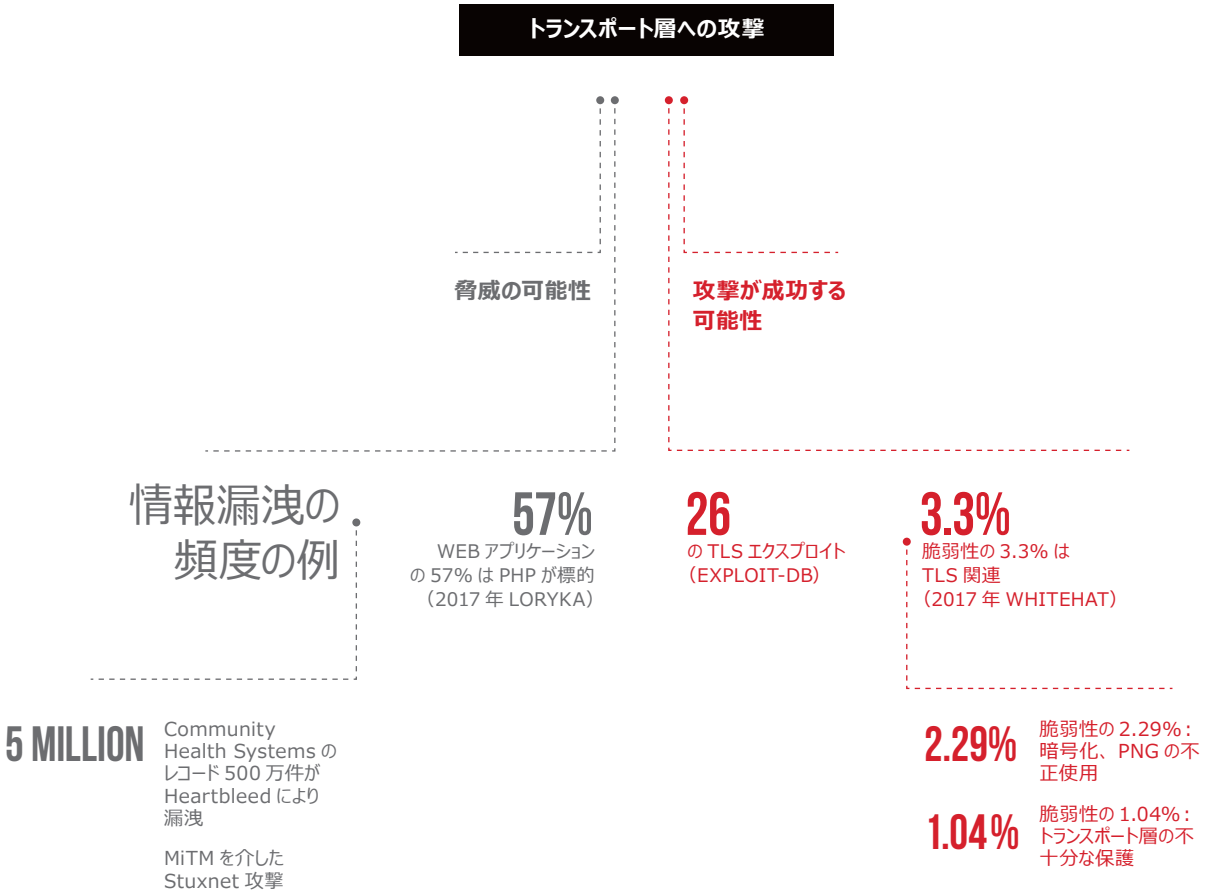
暗号化を使用しない場合に続く次の最悪なケースは、古くて脆弱な暗号または短い暗号化キーを使用することです。

TLS の不十分な設定とは具体的に何でしょうか。以下に、破壊される、または簡単に侵害されることが明らかにされている条件を示します。

- SSL のすべてのバージョン
- DES、RC4-40、DHE-RSA-Export、MD5、RC4 アルゴリズム
- 128 ビットより小さいキー
- 2048 ビットより小さい SSL 証明書

暗号化コンプライアンス標準は、新しい暗号化攻撃が見つかるたび、およびコンピューティング ハードウェアが改善されるたびに新しくなります。そのため、米国 National Institute of Standards FIPS PUB 140-2 暗号モジュール[25] のためのセキュリティ要求または標準追跡サイトなどの、強力な暗号化標準に常に注意しておくことが重要です。[26]

図 42：TLS 攻撃の可能性とその成功の可能性について



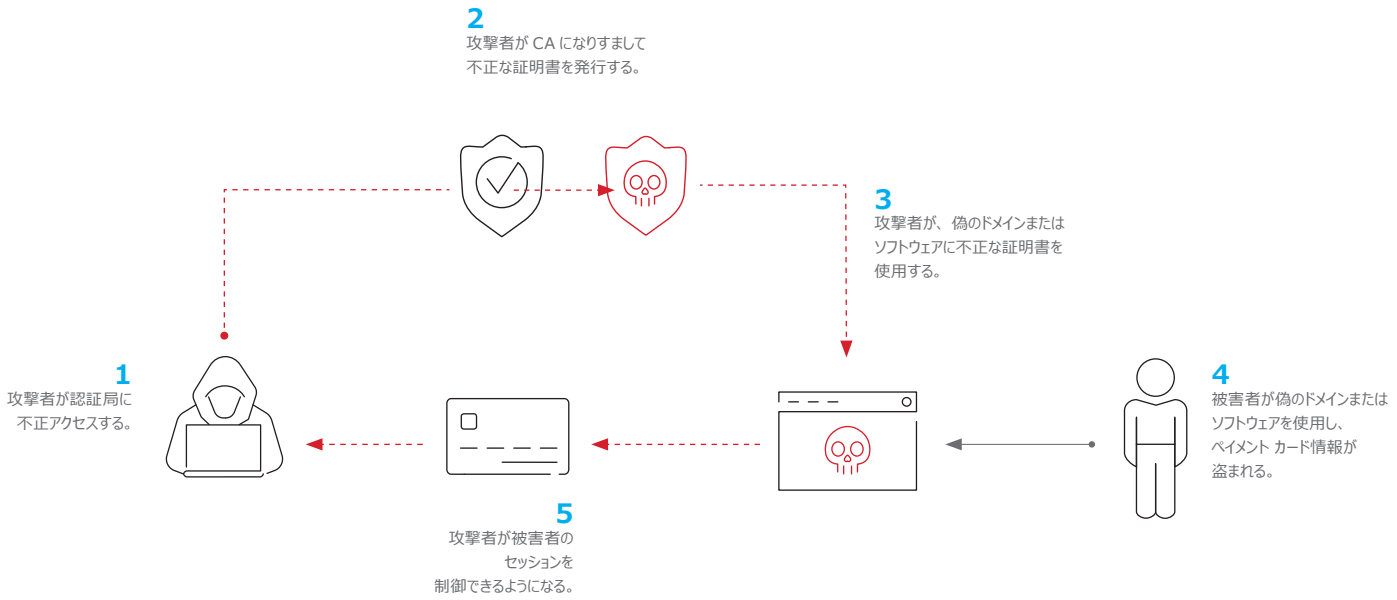

具体的なトランスポート層攻撃および潜在的な攻撃に関するデータについて、図 42 に、この脅威に関連するデータの内訳を示します。

新しい TLS プロトコル脆弱性は、年に約 2 回発表されています。しかし、Heartbleed は例外として、発表される TLS プロトコル脆弱性の大部分は、非現実的であり、実際の情報漏洩でもあまり使用されていません。最大の被害の 1 つが、2014 年の Community Health Systems（CHS）の事例です。攻撃者が Heartbleed 脆弱性 [27] を使用して Juniper Networks SSL/VPN を侵害したことで、CHS は約 500 万の社会保障番号を失いました。このとき、これは、Heartbleed を原因とする最大の情報漏洩でした。

史上最大の TLS/SSL 攻撃と言えば恐らく、イランの原子炉の遠心分離器制御システムを標的とした Stuxnet マルウェア [28] です。この感染ベクトルの 1 つは、Windows Update サーバに対する MitM 攻撃でした。Windows Update は、TLS/SSL で保護されていますが、攻撃を成功させた国家は、MD5 ハッシュ衝突をリアルタイムで計算することで、クライアントを騙して、SSL/TLS 接続に対する MitM 攻撃を使用できました。このような攻撃は、それまでは完全に理論上のものでした。

TLS の脆弱性として言及されることは少ないものの、最も影響があるものの 1 つは、全体的な基礎の根底にある乱数発生器のセキュリティです。完全な乱数をコンピュータが入手することは困難ですが、2008 年の有名な Debian-SSH キー脆弱性 [29] で発覚したように、不適切なランダム データは予測可能なキー生成の原因になります。同様に、研究者により、100 の SSL キーのうち 1 つは、不適切なランダム シードが原因で推測可能であることが分かっています。[30] これは、総当たり攻撃によるキーの推測に数千年かかるとされる非対称暗号システムでは許容できない確率です。

図 43：不正なデジタル証明書の攻撃経路




サイバー犯罪者および国家ぐるみのハッカーは、スパイ活動やマルウェアの偽装に不正な証明書を利用し続けています。

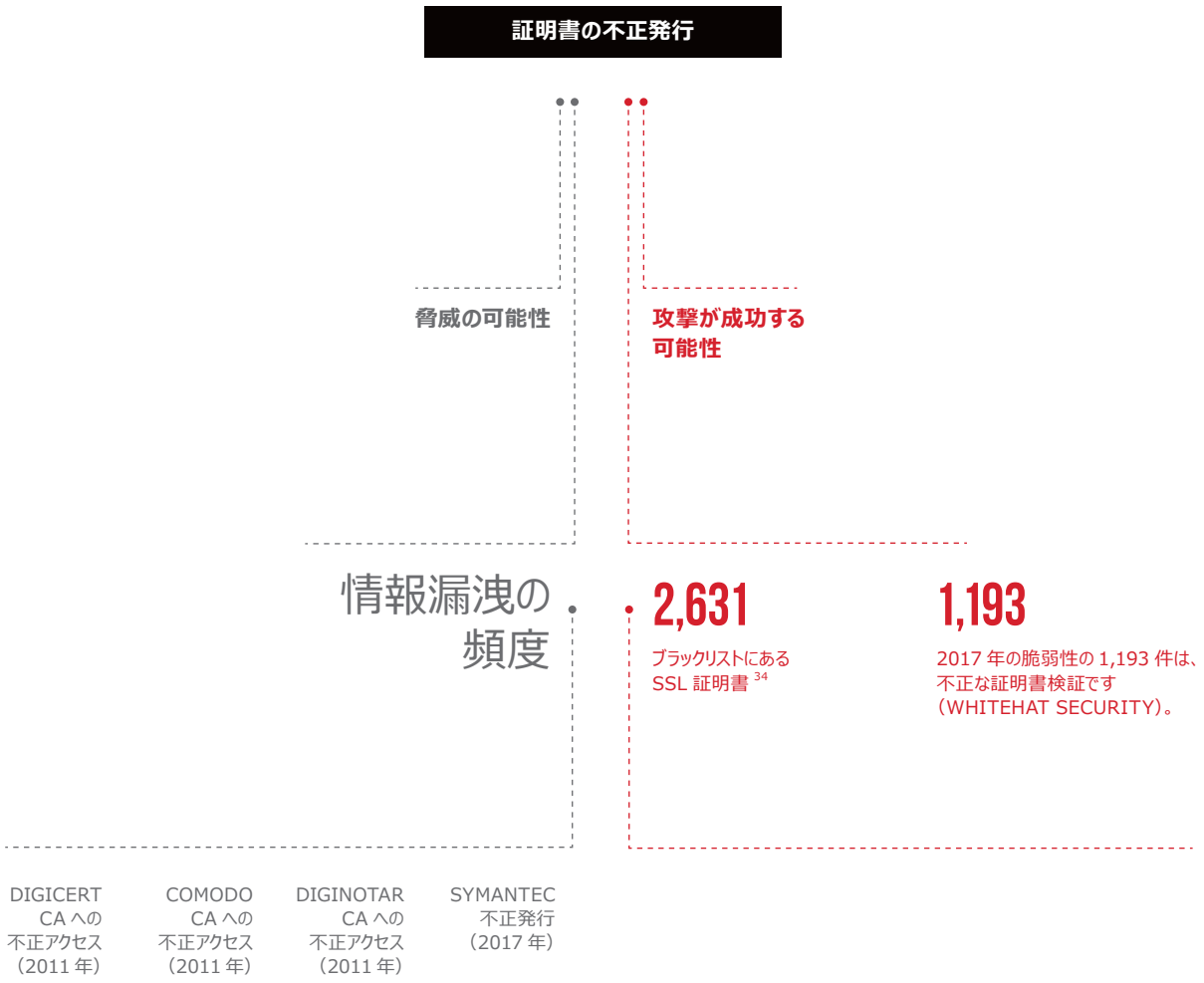
## 不正な証明書

デジタル証明書は、ユーザのアプリケーション サーバを認証する TLS/SSL のサーバ側のアンカーです。信頼できる認証局（CA）からの有効な証明書は、証明書の識別情報を検証します。無効な証明書が受け入れられることがあるので、「有効」および「信頼できる」という言葉は重要です。さらに悪質なのは、信頼できる CA が信頼できないように見えることです。
いずれの場合も、被害者は偽のアプリケーションを実行または使用するように騙され、マルウェアがインジェクションされたり、認証情報が盗まれたりします。

証明書への信頼が大きいことを考えると、不正行為が発生した場合の被害は壊滅的になります。以下の主な事例を考えてみます。

- 2011 年、攻撃者は、DigiCert Group の証明書を不正に取得して、Google、Yahoo、Microsoft および Mozilla になりすましました。[31]
- この事例では、攻撃はイランに端を発しました。洗練されたサイバー犯罪者および国家ぐるみのハッカーは、スパイ活動やマルウェアの偽装に不正な証明書を利用し続けています。
- さらに 2011 年、DigiNotar CA[32] が不正アクセスを受けたことで、DigiNotar が Mozilla の独自の証明書数件に署名していたにもかかわらず、Mozilla に情報漏洩を通知できなかった DigiNotar は崩壊しました。
- 2015 年、China Internet Network Information Center（CNNIC）は、エジプトの MCS Holdings に運用規定なしで中間証明書を発行しました。この証明書は、証明書の偽造に使用される可能性がありました。Google は、これらの証明書を Chrome ブラウザで信頼できない証明書として扱うことにしました。[33]

図 44：証明書の不正発行の可能性とその成功の可能性について




秘密鍵を含むプライベートの証明書が、所有者以外の企業に電子メールで送られたとして、23,000 件以上の証明書が失効しました。

これらの大規模な攻撃の後も、CA の信頼は引き続き問題になっています。2017 年、Symantec の認証局ビジネスは、Google 証明書を不正に発行したとして Google から非難されました。[35]

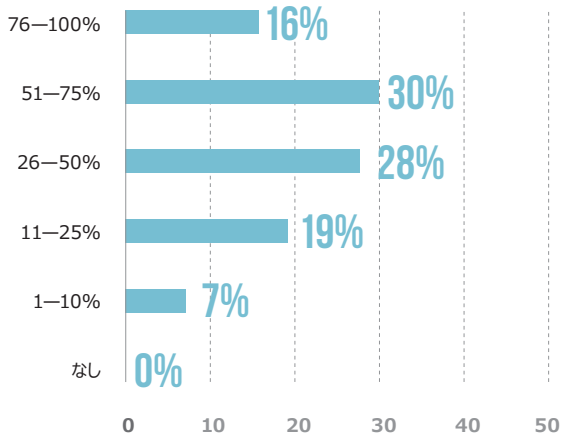
2018 年、Trustico は Symantec、GeoTrust、Thawte および RapidSSL の証明書が信頼できないと宣言しました。[36] 秘密鍵を含むプライベートの証明書が、所有者以外の企業に電子メールで送られたとして、23,000 件以上の証明書が失効しました。

**自己署名証明書を使用することのリスクの増加**

証明書の重要な価値の 1 つは、実行するサイトを認証することです。これを有効に行うため、証明書は、信頼できる第三者により署名される必要があります。通常、これは、Comodo、Entrust、GlobalSign または Let's Encrypt などの認証局です。信頼できる署名がない場合、ユーザは、証明書の信頼性を確認できません。つまり、アプリケーションの信頼性も問題になります。

しかし、アプリケーションの妥当性の証明を提供しない自己署名証明書が、数多くのアプリケーションで使用されています。通常、自己署名された証明書を使用する Web サイトは、設定されていないデバイスであり、信頼できないインターネットに接続すべきではありません。F5 Ponemon セキュリティ調査では、やや不安のある結果を示し、46% もの回答者が Web アプリケーションの半分以上 (51% ～ 100%) で自己署名証明書を使用していると答えています (図 45 参照)。一方で明るい話題は、F5 Labs の研究によると、自己署名証明書の使用率が急落していることです。

図 45 : 自己署名証明書を使用する WEB アプリケーションの割合




明るい話題は、F5 LABS の研究によると、
自己署名証明書の使用率が急落していることです。

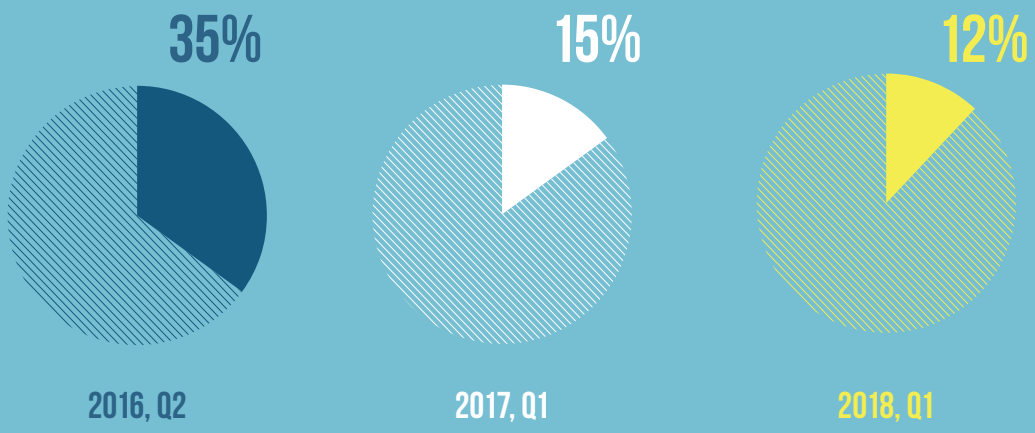



図 46 : 自己署名証明書の普及率

攻撃者が DNS に不正アクセスする方法は 2 種類あります。DNS トラフィック自体を傍受するか、ドメインのレジストラを攻撃することです。

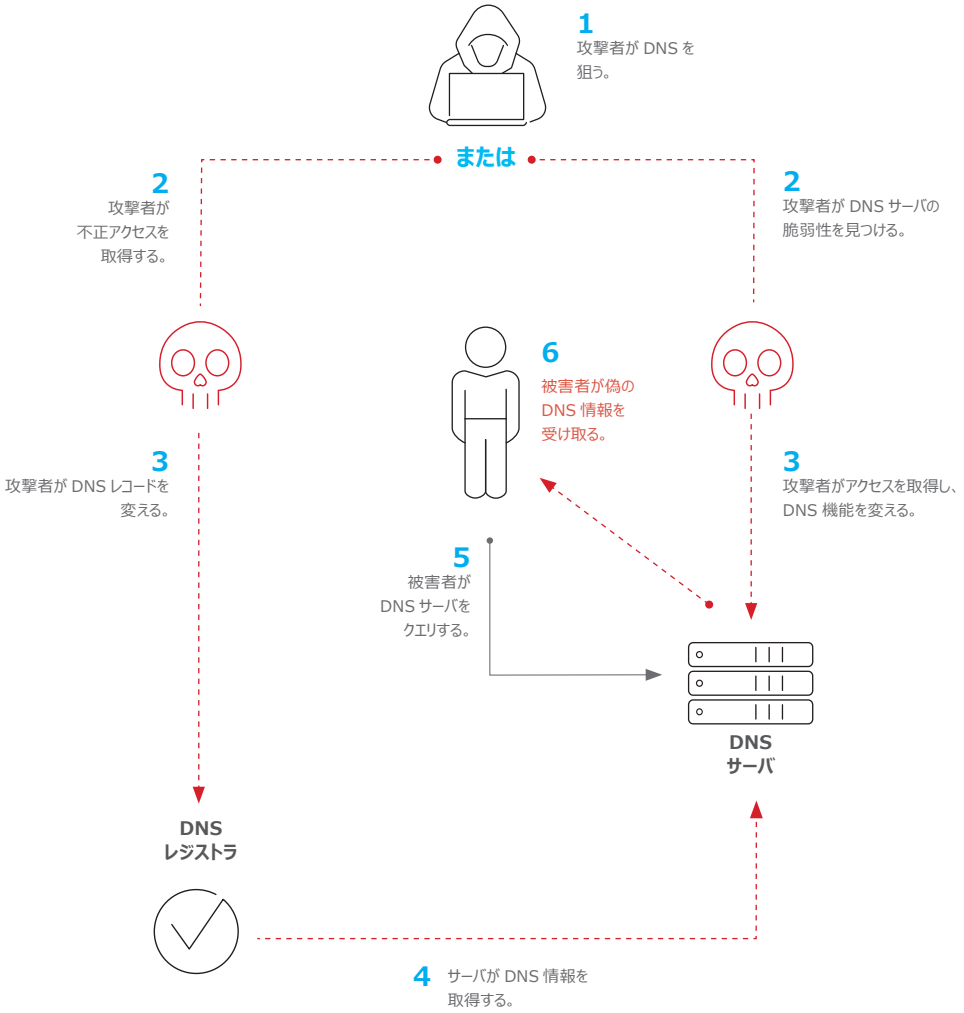



図 47: DNS ハイジャック攻撃の経路

## DOMAIN NAME SERVICE ハイジャック

アプリケーションを間接的に攻撃するもう 1 つの経路は DNS です。アプリケーション自体が完全に保護されていても、DNS 攻撃はアプリケーションを破壊または停止できます。

攻撃者が DNS に不正アクセスするときの主なベクトルは 2 つあります。DNS トラフィック自体を傍受するか、ドメインのレジストラを攻撃して DNS レコードを変更または独自の DNS レコードを挿入するかです。これは簡単で、ドメイン所有者の電子メールに不正アクセスして、ドメイン レジストラのアプリケーションのログイン認証情報を取得するだけです。これらの変更は、図 47 に示すように、正規の DNS サーバを通ってユーザに送られます。

DNS は、アプリケーションのインフラストラクチャの不可欠な部分です。最もリスクの高いアプリケーションは、価値のあるサービスまたはデータにアクセスする Web サイトです。DNS 攻撃がこれらの大規模な Web アプリケーションに仕掛けられると、被害は簡単に数千規模に及び、被害額は数百万ドルまで達します。

以下は、過去数年における主な DNS 情報漏洩のリストです。

## 2018

- BGP ポイズニングと DNS スプーフによる MyEtherWallet 暗号通貨の盗難 [37]
- Point-of-Sale マルウェアが DNS クエリを介してクレジット カード データを盗む [38]
- ドメイン盗難により数千の Web サイトに障害 [39]

## 2017

- ブラジルの銀行のドメインがハイジャックされ、顧客が略奪被害に遭う [40]
- ドメイン レジストラのハッキングにより Fox-It.com の MitM 攻撃が発生 [41]

## 2015

- セントルイス連邦準備銀行が DNS 情報漏洩を受ける [42]

## 2014

- ICANN が攻撃され、DNS 管理システムが不正アクセスされる [43]
- 攻撃者が catholichealth.net ドメインをリダイレクトして、Catholic Health Initiatives の患者の電子メールが公開される [44]

## 2013

- オランダの DNS サーバがハッキングされる：数千のサイトがマルウェアを提供 [45]

DNS 攻撃は一般的ではありませんが、アプリケーションに壊滅的な被害を与えます。

図 48：DNS 攻撃の可能性とその成功の可能性について

# サービス拒否攻撃（DoS 攻撃）

ここでは、アプリケーションに対するサービス拒否（DoS）攻撃と分散型サービス拒否（DDoS）攻撃を取り上げます。また、ネットワーク DDoS 攻撃とアプリケーション DDoS 攻撃は区別しています。

**$20**

**強力なTHINGBOTを構築する IOT エクスプロイトが増加したことで、攻撃者は、わずか $20 で 300 GBPS 規模の攻撃を仕掛けることが可能になりました。**

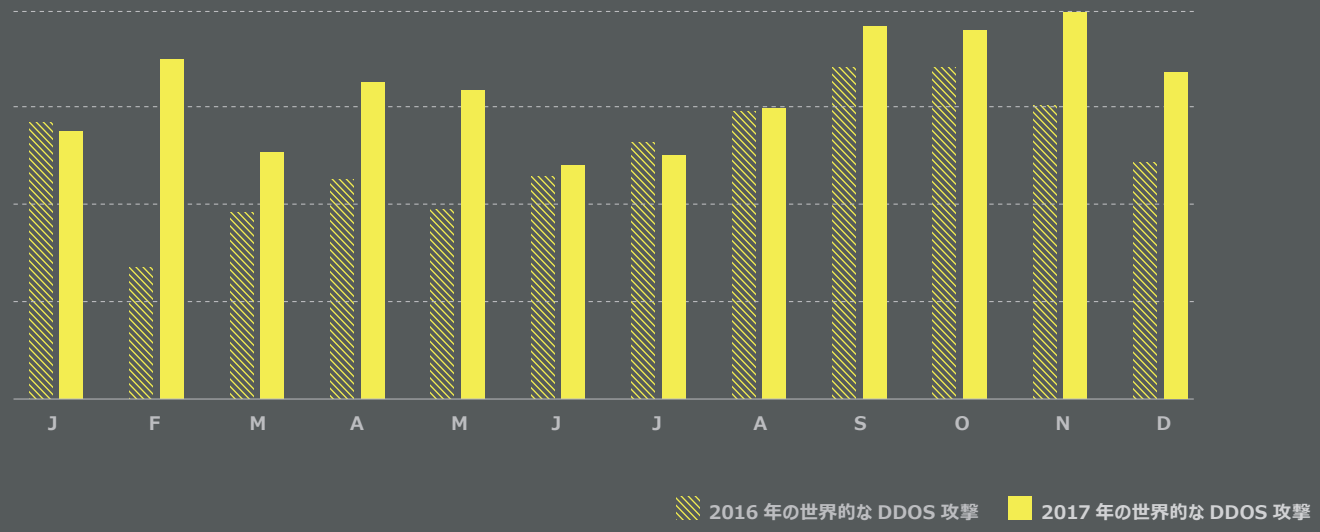
ネットワーク DDoS 攻撃では、パケット タイプが TCP や UDP などに限定されていること、TCP ヘッダが URG や RST などに限定されていること、および、ほとんどの場合、スプーフィング メカニズムが限定されていることが知られています。これらすべての限定状況から、ネットワーク DDoS 攻撃への対策は芸術というよりも科学です。一方、アプリケーションに対する攻撃への対策はさらに難解で、カスタムのスクラビングや才能のあるセキュリティ担当者が要求されます。アプリケーション DDoS 攻撃はマルチホスト、マルチオリジンおよびマルチベクトルな攻撃なので、より強力であり、阻止することもより困難です。攻撃者は、戦略に新しい武器を取り入れ、アプリケーション DDoS 攻撃をさらに多く使用しています。アプリケーション攻撃は、通常、単独の攻撃ですが、何日も何週間も続く攻撃キャンペーンに拡大することもあります。

アプリケーションは、機密性および完全性攻撃から保護できますが、アプリケーション自体は DDoS 攻撃からの可用性損失に非常に脆弱です。攻撃者はいとも簡単にボットネットから大規模な負荷を掛けてアプリケーション サイトを破壊できます。アプリケーションに対する DoS 攻撃はなくならず、いたずらや破壊行為ではなく、攻撃者にとっての便利な道具となっています。

F5 Labs の Hunt for IoT レポート シリーズでは、巨大な勢力の DDoS 攻撃に利用されている大規模な DDoS thingbot（IoT デバイスで構成されるボットネット）を特集しています。IoT デバイスを改悪して DDoS 攻撃の発射台に利用することは、thingbot の発見とほぼ同時に成熟している成長産業です。

- DDoS は、thingbot から発動する最も一般的な攻撃です。その後、暗号通貨のマイニング、バンキングを標的としたトロイの木馬の提供、および IoT デバイスをプラスチックとシリコンでできた操作不可能な塊に変える永続的な Permanent Denial-of-Service（PDoS）攻撃の起動が行われます。
- 強力な thingbot を構築する IoT エクスプロイトが増加したことで、攻撃者は、わずか $20 で 300Gbps 規模の攻撃を仕掛けることが可能になりました。
- DDoS thingbot は 2008 年から確認されていますが、現在認識している DDoS thingbot の 64% が確認されたのはわずか 2016 年以降です。

図 49：2016 年～2017 年における月別の F5 SILVERLINE DDOS 攻撃傾向




F5 の Silverline DDoS スクラビング サービスにおける社内統計情報（図 49）によると、DDoS 攻撃は年々着実に増加しています。

一般的に、サービス拒否攻撃（DoS 攻撃）にはいくつかの種類があります（Exploit-DB のデータベースには 5,665 の DoS エクスプロイトがあります）。 最も基本的な攻撃では、実行するアプリケーションまたはアプリケーションをサポートするサービスを破壊する既知のエクスプロイトが使用されます。 現在でもネットワーク ボリューメトリック攻撃は非常に人気です。これらの攻撃には、複数のソースから大量のトラフィックを直接送り込む方法と、悪用可能なサービスを踏み台とする非対称リフレクション攻撃方法の 2 種類があります。しかし、アプリケーションに対するサービス拒否攻撃（DoS 攻撃）は無視できません。アプリケーションに対する DDoS 攻撃を初めて分類した 2016 年以降、アプリケーションを標的とした攻撃は年々着実に増加しています。

図 50：2016 ～ 2017 年のカテゴリ別 DDOS 攻撃

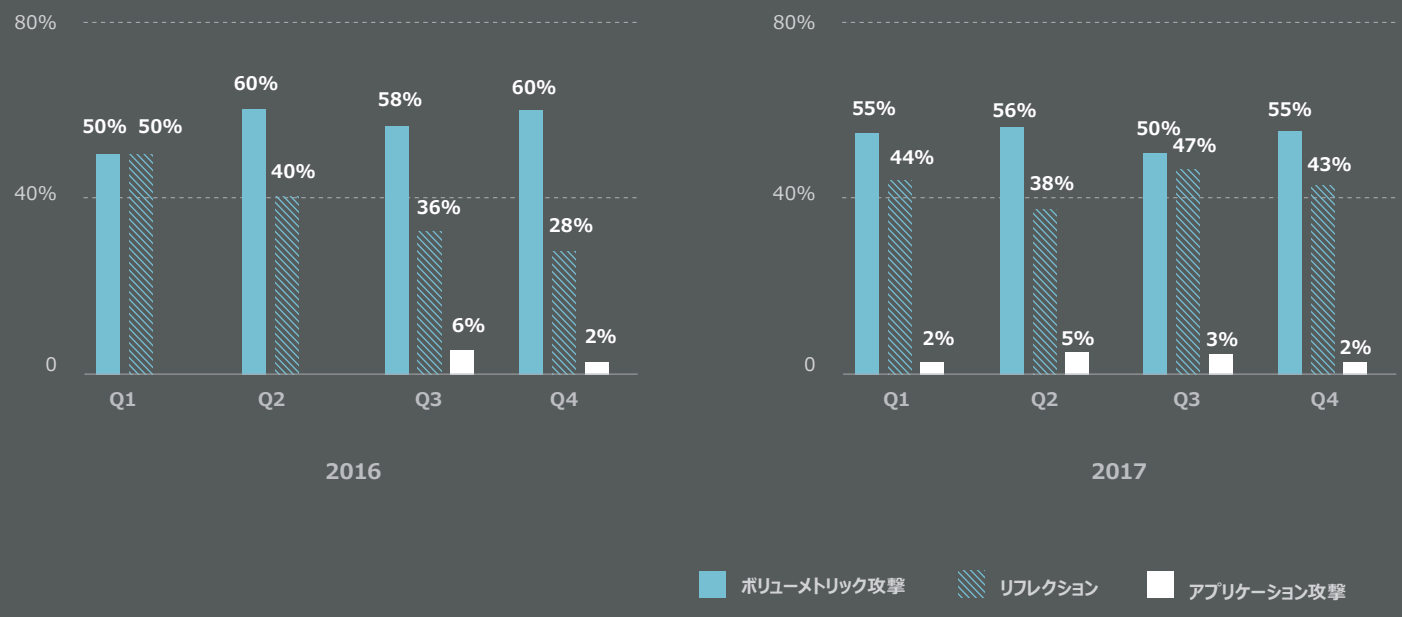
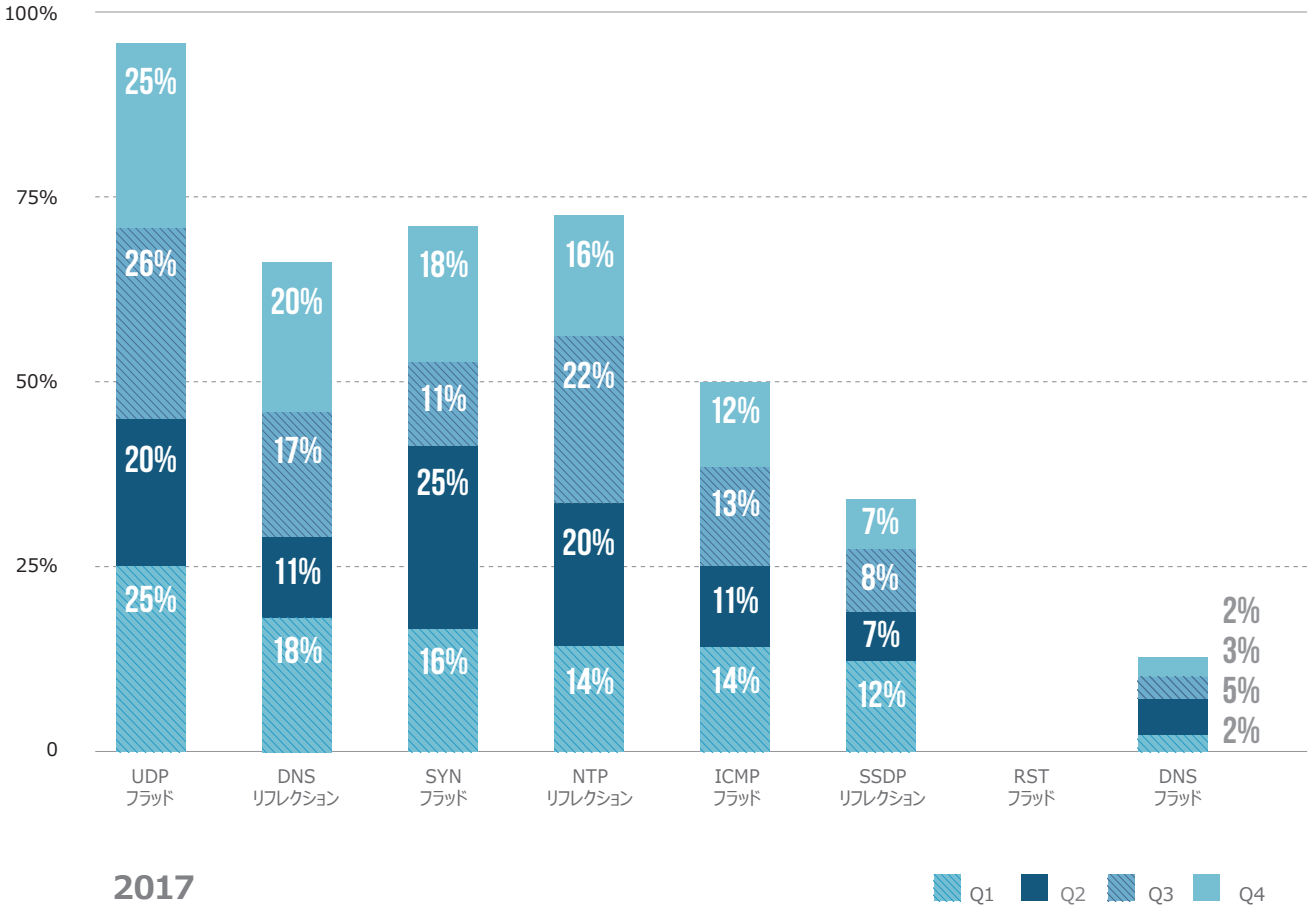


図 51：2017 年のプロトコル別 DDOS 攻撃




リソース枯渇攻撃に脆弱なアプリケーションにおいて、ハイジャックされたボットで実行される簡単なスクリプトは、そのアプリケーションで最もリソースを消費するコンポーネントにリクエストを集中させ、大量の負荷を生成できます。通常、攻撃者は、マルチスレッドを使用して、このような攻撃を増幅します。

時代遅れの UDP および SYN ネットワーク フラッド攻撃もまだよく使われていますが、セキュリティ担当者は、これらの簡単な攻撃を阻止する方法を身に付けています。そのため、図 51 に示すように、攻撃者は、より新しく効果的なネットワーク攻撃でその技術を高める必要がありました。その 1 つが、一部のアンチ DDoS ソリューションでは阻止できない、IPv6 DDoS 攻撃です。[46] インターネット アドレス空間での IPv6 採用率が 25% に近付く中、これらの攻撃は今後も増えるはずです。[47]

DDoS のもう 1 つの新しいベクトルは、一般的なポイントツーポイントのカプセル化プロトコルを使用した Generic Routing Encapsulation（GRE）攻撃です。GRE はスプーフできませんが、至る所でネットワーク トンネリングに使用されているので、通常、この攻撃はファイアウォールおよびルータ フィルタを介して可能になります。最近、Mirai thingbot は、Dyn および Krebs on Security のホステッド DNS サービスに対する攻撃の一部に GRE 攻撃を使用し、大規模なサービス停止を引き起こしました。

## DDoS 攻撃者は、一部のサイトが顧客にパッチ処理およびメンテナンスのためのステータス アップデートを提供している事実に気付いています。

DDoS 攻撃者は、一部のサイトが顧客にパッチ処理およびメンテナンスのためのステータス アップデートを提供している事実に気付いています。攻撃者は、これらのメンテナンス期間を攻撃の標的にして、運用に最大級の大混乱をもたらすことができます。さらに厄介なことに、攻撃者は、管理者の気をそらしデータ盗難と不正行為攻撃を同時に成功させるための陽動作戦として DDoS 攻撃を利用しています。

# DDoS 攻撃は、高可用性のインターネット アプリケーションに依存する電子商取引および金融業界に多額の損失を与えます。

一般的に、組織は DDoS 攻撃を重大な問題だと認識しています。F5 Ponemon セキュリティ調査において、アプリケーションの可用性に特有なこのような攻撃の影響について質問したところ、回答者の 81%（図 52 参照）は、アプリケーションまたはデータにアクセスできなくなる DDoS 攻撃の被害は非常に大きくなると思っていることが分かりました。

実際、DDoS は、金融サービス業界または電子商取引などの高可用性のインターネット アプリケーションに依存する業界において非常に大きな被害を与えています（図 53 参照）。決算処理機関がクレジット カードをわずかな間でも受け付けることができない場合に起きる影響を考えてみてください。

Ponemon 調査では、アプリケーションまたはデータへのアクセスに影響する DDoS 攻撃による被害額が $10,000 未満で済むと考えている回答者はいませんでした。回答者の 4 分の 3 以上は、DDoS 攻撃による被害額が $500,000 ～ $10,000,000 になると考えていました。

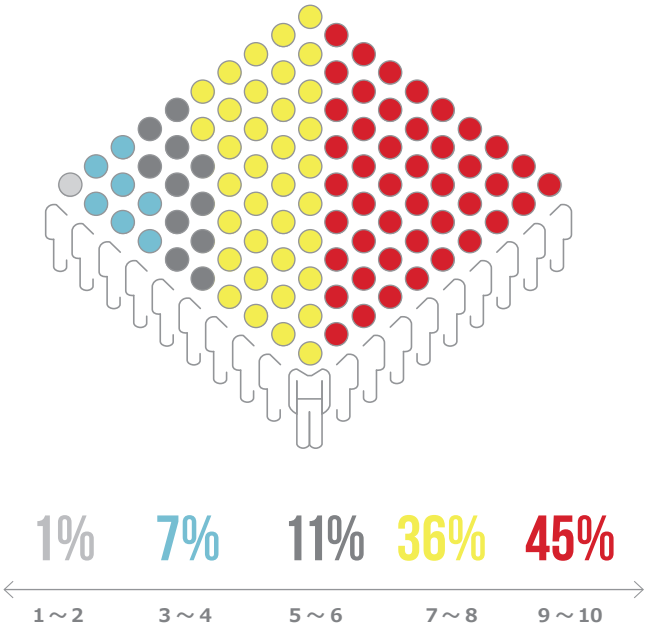



図 52：アプリケーションまたはデータへのアクセスができなくなる DDOS 攻撃の被害のしきい値

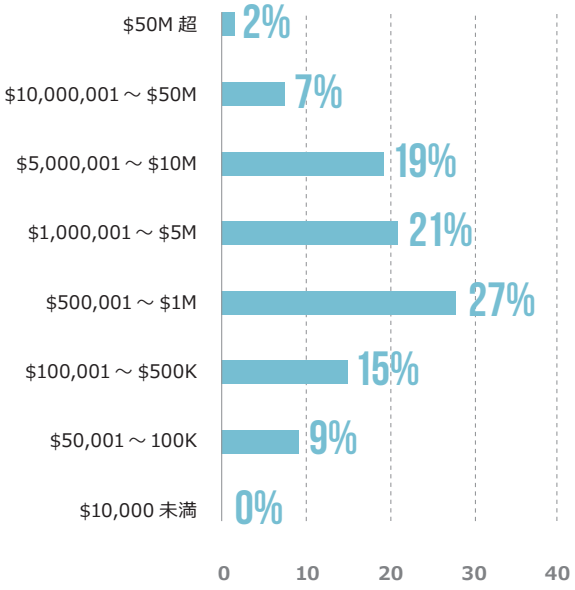
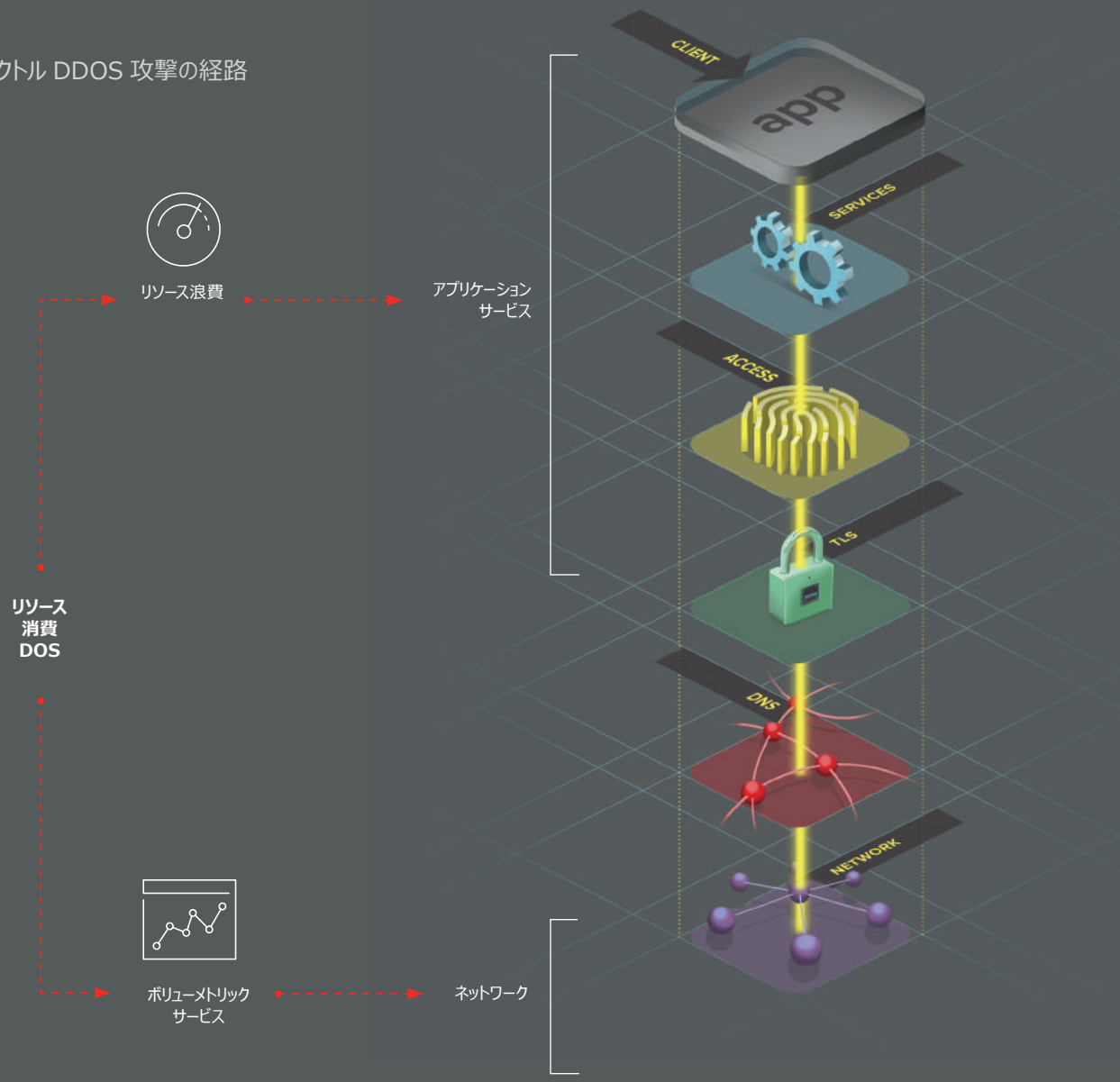



図 53：アプリケーションまたはデータへのアクセスができなくなる DDOS 攻撃の被害額

図 54：マルチベクトル DDOS 攻撃の経路



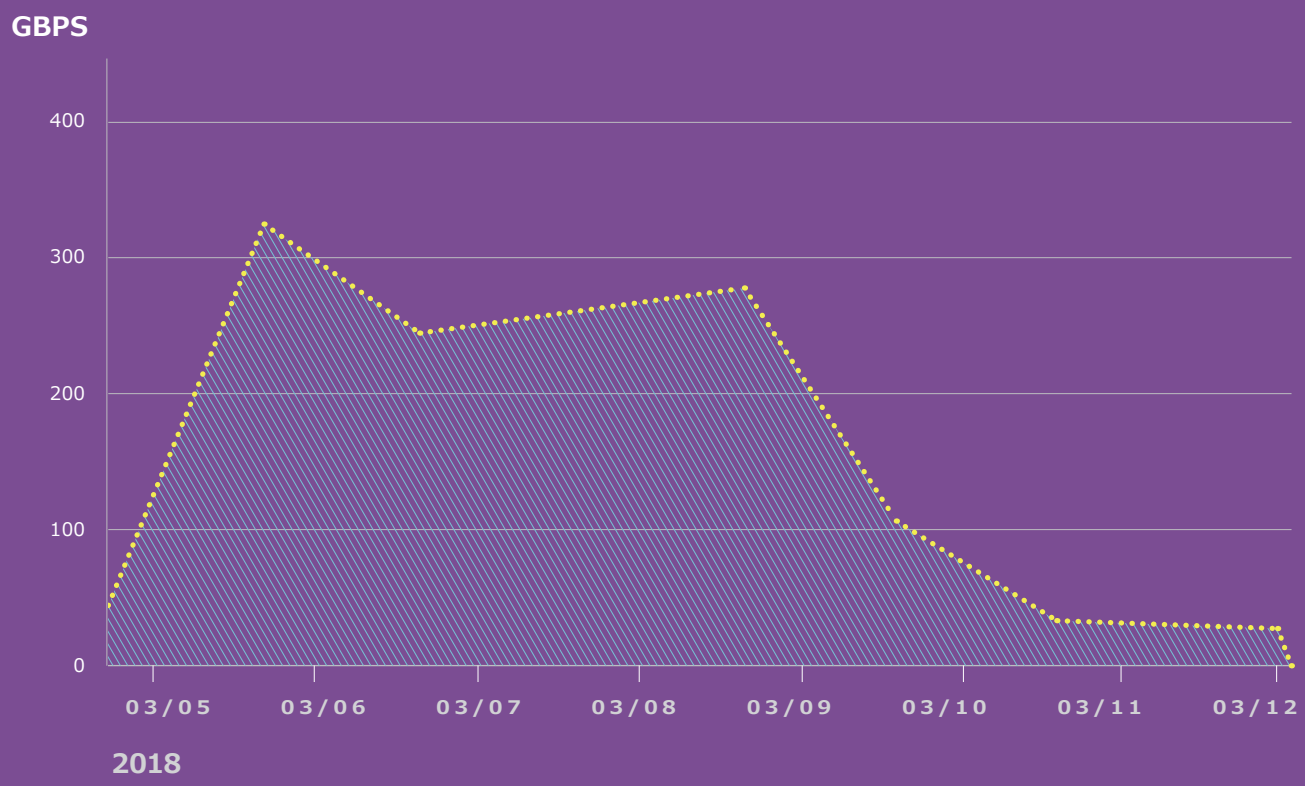

**マルチベクトル DDoS**

最も影響のある DDoS 攻撃では、使用される攻撃ベクトルは、1 つではなく、多くあります。複数の種類のサービス拒否攻撃（DoS 攻撃）を組み合わせた一連の攻撃は、トラフィックのスクラビングまたはドロップのいずれかに必要な対策ソリューションを混乱させるだけでなく、攻撃の規模を拡大できます。そのため、2018 年 3 月に確認された 1.35 Tbps 規模の GitHub 攻撃のように、可能な限り最大の混乱を Web アプリケーションにもたらします。[48]

以下に、マルチベクトル DDoS 攻撃が発生する一般的な経路を示します（図 54 参照）。

1. 偵察ネットワーク スキャンが発生します。通常、これは数百のプローブおよびポート スキャンとして現れます。この攻撃の目的は、その後の攻撃の標的を探して品定めすることです。通常、数万のネットワーク サービスがアプリケーション インフラストラクチャ全体でプローブされます。攻撃者は、ネットワーク ボトルネック、バックエンド サーバおよびリソースを消費するアプリケーション サービスを見つけようとします。
2. データが分析されると、「金を払わないとサイトがダメになる」と脅し、身代金が要求されます。
3. DDoS 攻撃はすぐに仕掛けられます。攻撃は、ネットワークを詰まらせ、ルーティングを混乱させる従来のボリューメトリック ネットワーク フラッド攻撃から始まります。規模がギガバイト単位になることもありますが、これはネットワーク運用チームの時間を潰すためのただの妨害行為です。
4. 本当の攻撃はその後に始まります。この攻撃は、ポート 80 を対象としたアプリケーション固有の DDoS 攻撃（レイヤ 7 攻撃）です。これらの新しい DDoS 攻撃は、バックエンド コンテンツ デリバリ サーバ、過剰に負荷がかかったルータ、リソースを消費するアプリケーション サービスを標的とします。一般的な戦略は、複雑なクエリをトリガしてアプリケーション全体を完全停止させる Web リクエストを送信することです。

図 55：マルチベクトル大容量攻撃

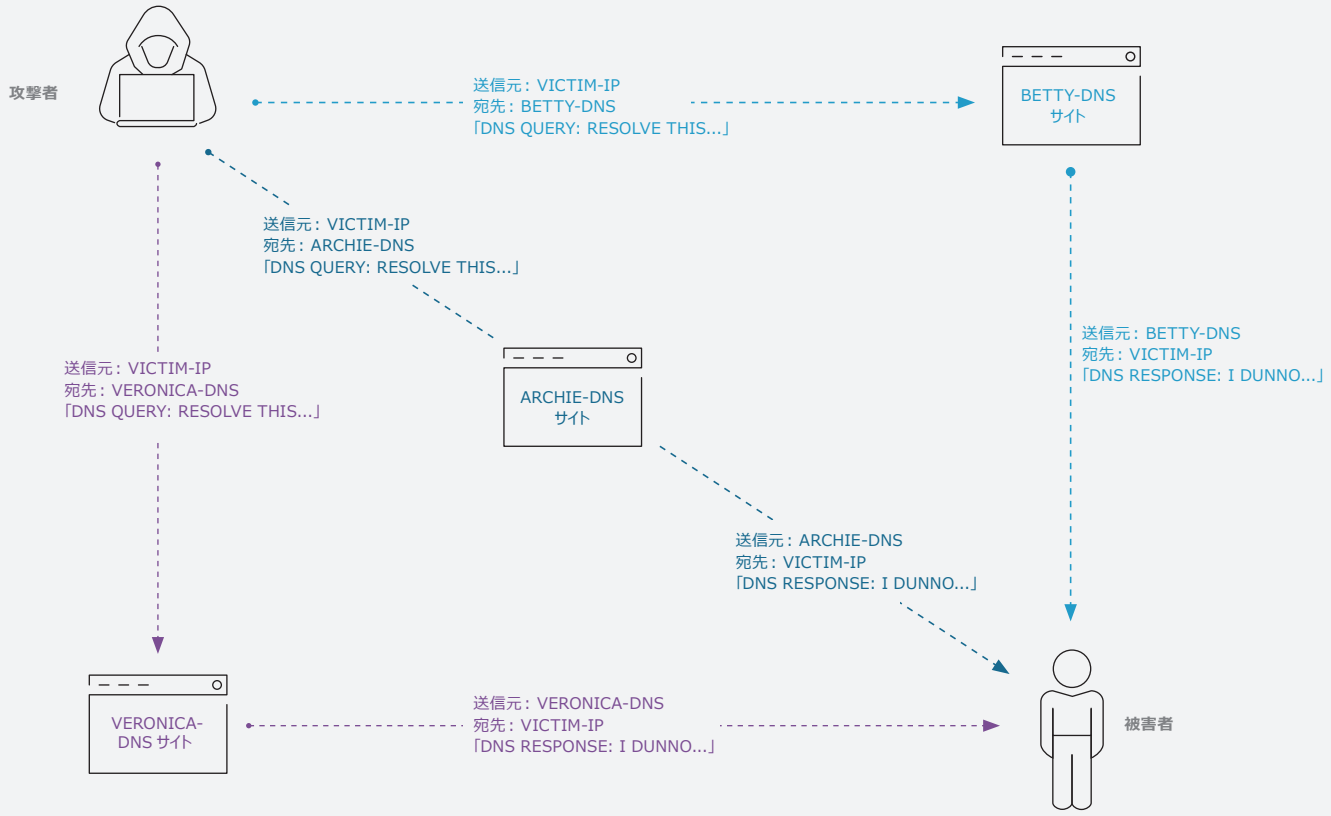



これらの攻撃について知っておくべきことは、攻撃者はインフラストラクチャで見つけたどのような脆弱性でも標的にするということです。

図 54 に示す特定のパターンの攻撃は、F5 の DDoS 対策チームがこの 1 年で驚くほどの頻度で確認しています。これらは、特定の方法論を示し、同じツールと攻撃を使用します。このような攻撃では、1 分間に 2,000 ページ以上のリクエストが数十万の個別の IP アドレスから送信されます。図 55 に示すように、通常、平均で 1 秒あたり約 170 ギガバイトですが、最高で 1 秒あたり 325 ギガバイトになります。

これらの攻撃について知っておくべきことは、攻撃者はインフラストラクチャで見つけたどのような脆弱性でも標的にするということです。たとえば、特定の URL に対する呼び出しを繰り返して集中的な CPU 負荷をトリガする、アプリケーション サイトにフィードするコンテンツ配信サーバにフラッド攻撃を仕掛ける、または冗長ではないネットワーク パスにトラフィックを集中させ妨害することもあります。アプリケーションを機能低下または不安定化させることができればどこでも標的になります。

図 56：リフレクション DDOS 攻撃の経路




**リフレクションおよびアンプ攻撃**

攻撃者は、最も基本的な DDoS 技術も改善しようとしています。マルチベクトル DDoS 攻撃に見られるように、攻撃スクリプトを改善するだけでなく、アプリケーション サービスを破壊し攻撃に関与させています。

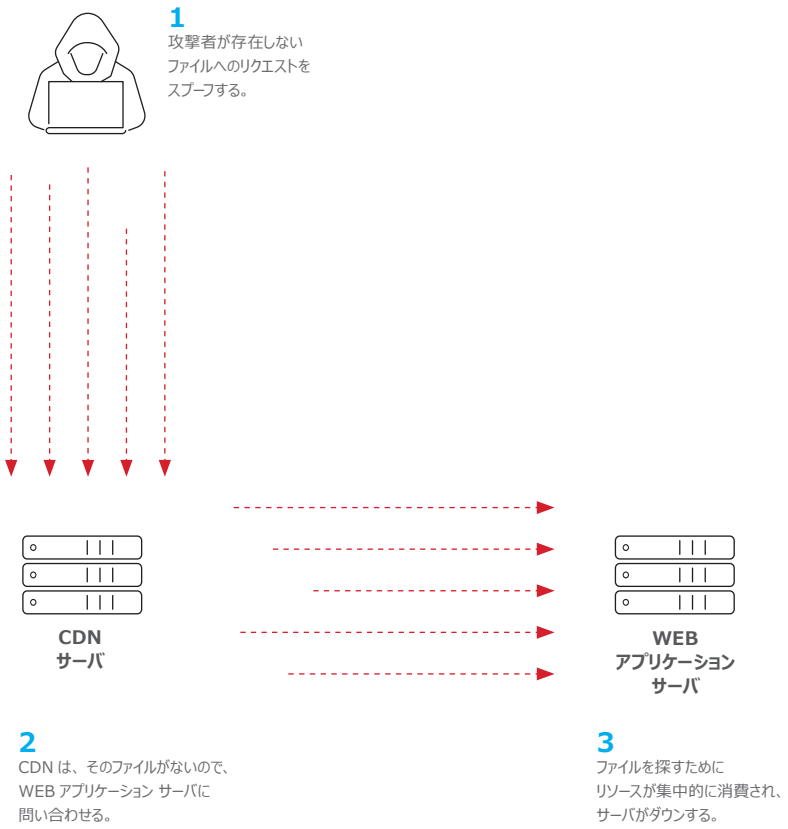
ネットワーク リフレクション攻撃は、アプリケーションを利用して、被害者自身および他の被害者を攻撃します。多くの場合、設計上、送信者のアドレスが検証されず容易にスプーフィングできる簡単な UDP プロトコルが使用されます。一般的に、UDP は、信頼できるミッションクリティカルなネットワーク用に設計されていません。RFC が明言しているように、「データのストリームを規則正しく安定して実現する必要があるアプリケーションが Transmission Control Protocol（TCP）を使用します」。[49] しかし、多くの便利なインフラストラクチャ サービスは UDP に基づいているので、リフレクション攻撃に使用されることに対して脆弱です。これらには以下のプロトコルが含まれます。

- Simple Service Discovery Protocol（SSDP）
- Lightweight Directory Access Protocol（LDAP）
- Universal Plug and Play（UPnP）
- Character Generator Protocol（CHARGEN）
- Session Initiation Protocol（SIP）

Internet Control Message Protocol（ICMP）は、もう 1 つのスプーフ可能なプロトコルで、由緒ある Smurf 攻撃などのリフレクション攻撃にも使用されています。[50] リフレクション攻撃は増幅することもでき、これにより攻撃者は、図 56 に示すように、ネットワーク パケットのフラッドを拡大し、実際の送信先アドレスをマスクできます。

増幅型 DDoS 技術は、非常に強力で、数 10 年も前から存在していますが、攻撃者は、ネットワーク フラッドの枠を超え、アプリケーションを中心としてリフレクション増幅攻撃を成功させています。つまり、攻撃者は、正規のアプリケーション サービスおよびインフラストラクチャを弱体化させています。一般的な方法では、オープン DNS サーバで UDP を使用してスプーフした DNS リクエストを送信することで、DNS リフレクション攻撃を行います。スプーフされたリクエストは、DDoS 被害者から送られているように見えるので、DNS 回答がサーバから返され被害者が攻撃されます。このような DNS 攻撃は、拡大率が非常に高いので、小さいリクエストでも、トラフィックを 100 倍に増幅できます。

図 57：CDN リフレクション攻撃の経路




## 攻撃者は、これまでより効率的に、アプリケーション インフラストラクチャを物色して、スプーフまたは破壊できる獲物を探しています。

攻撃者は、他のアプリケーション サービスも増幅型リフレクション攻撃に利用しています。2018 年初めに注目を集めた攻撃の 1 つに、テラビット規模に及んだ Memcached DDoS という攻撃がありました。[51] Memcached は、サービス層の重要なコンポーネントですが、増幅型リフレクション攻撃に悪用できる唯一のコンポーネントというわけではありません。その他のリフレクトされるコンポーネントは、コンテンツ配信ネットワーク（CDN）デバイスです。CDN サーバは、キャッシュされた人気のコンテンツを保持することで、Web サイトおよびアプリケーションの速度向上をサポートします。しかし、存在しないファイルまたは画像に対するスプーフされたハッシュリクエストにより、CDN は、（存在しない）データのために主要なアプリケーション サーバにコールバックします。これにより、負荷は軽減せず、CDN からメイン Web サイトに余分な負荷がかけられます。このような DDoS 攻撃では、攻撃者は、いくつかの Web 呼出しを送信するだけで、アプリケーションのインフラストラクチャを分断できます（図 57 参照）。

リフレクション攻撃は、現在、Web サイトにレイヤ 7 フラッド攻撃を仕掛ける XSS を介したブラウザ マルウェアにより Web クライアントを乗っ取ります。最近、新しい WordPress Pingback DDoS 攻撃が出回っています。これは、WordPress ブログ サイトの自動化された通知メカニズムを悪用し、「ピンバック」POST リクエストをスプーフされた被害者の IP アドレスに送信します。

攻撃者は、武器に使うためのこのようなアプリケーションをより効率的に探しています。攻撃者は、アプリケーション インフラストラクチャ、スプーフまたは破壊できる獲物を探しています。今後はアプリケーション サービスを踏み台とする増幅型リフレクション DDoS 攻撃が増えることが予想されます。

### Transport Layer Security（TLS）サービス拒否攻撃（DoS 攻撃）

理論上の「総当たり、暗号化なし」の TLS 攻撃の新しいセットがあります。この攻撃は、クライアントがランダム ジャンクを TLS スタックに送信して、復号化しようとするだけです。[52] クライアントが暗号化を行わず、ランダム バイトをサーバに送信しているだけという事実から、攻撃の非対称性が復元します。この攻撃は約数回確認されていますが、まだ出回ってはいません。TLS Internet Engineering Task Force（IETF）委員会は、この攻撃が普及した場合に備えて対策を計画していますが、プロトコル自体の変更が必要になるでしょう。[53]

フランスの団体「The Hacker' s Choice」のオリジナルの SSL 再ネゴシエーション攻撃ツールは、脆弱なサーバとの RSA キー交換を繰り返し要求しました。これらのハンドシェイクでは、CPU 消費が攻撃クライアントよりサーバが 10 倍 [54] 多くなりました。この攻撃は、米国の大手銀行で確認され、そのセキュリティ分析の 1 つで説明されていましたが、[55] 最近は見られなくなりました。その理由の 1 つは、オリジナルの再ネゴシエーション攻撃が効力を発していた非対称性を持たない楕円曲線暗号化への世界的な移行です。

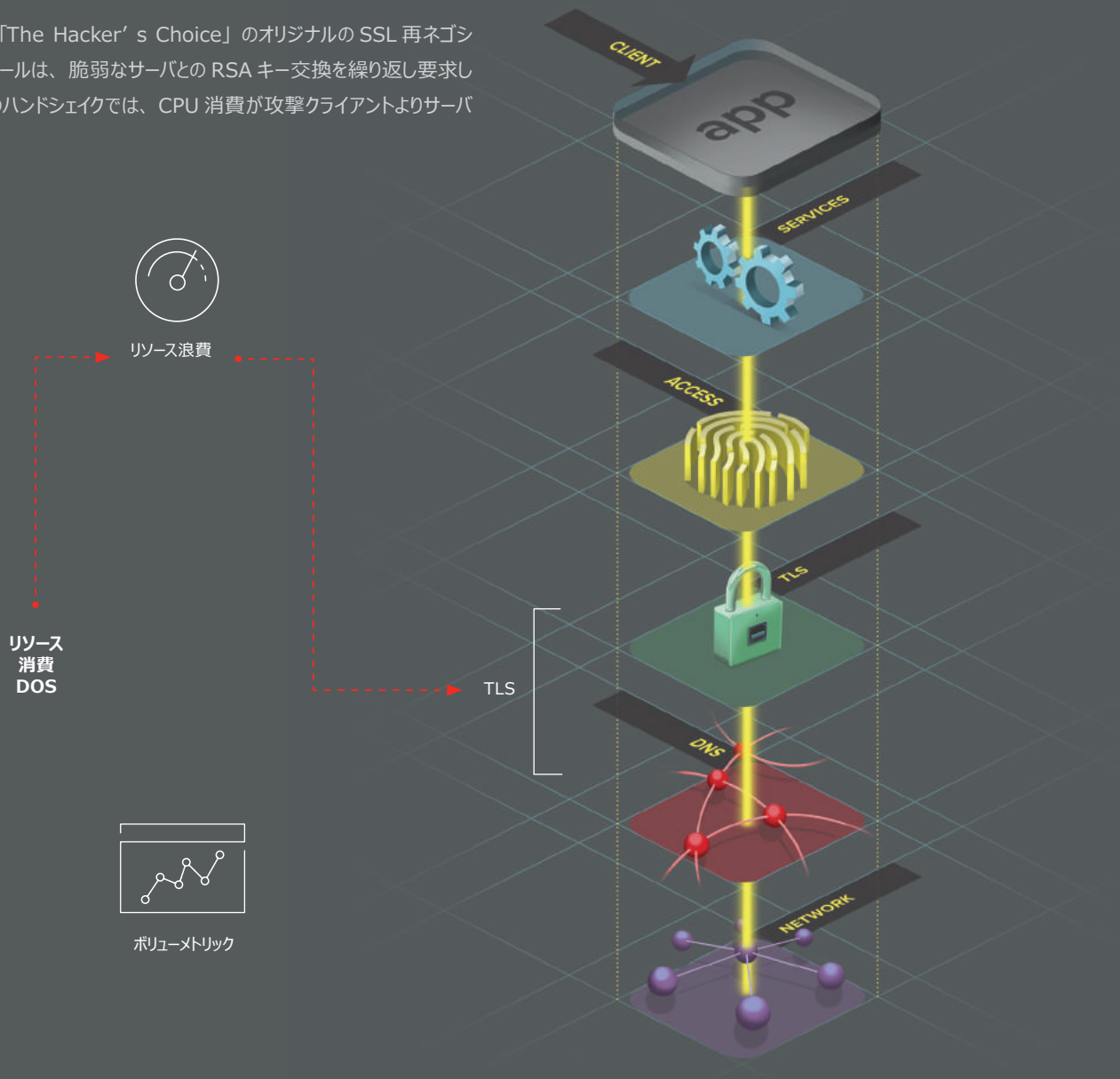



図 58：理論上の TLS DOS 攻撃の経路

# クライアント攻撃

攻撃者はアプリケーションまたはサポート インフラストラクチャに不正アクセスできない場合、アプリケーション クライアントを乗っ取ろうとします。アプリケーション クライアントに対するほとんどの攻撃は、ユーザの認証情報を直接盗むか、許可された進行中のセッションをハイジャックして、アクセスを盗むことを目的としています。

クライアントに対する攻撃は、個人への攻撃であるため、大きく取り上げられることや公開の義務がほぼないので、広く公表されていません。

クライアントに対する攻撃は、通常、特にアプリケーション情報漏洩と比較すると、広く公表されません。その理由は、個人への攻撃であるため、大きく取り上げられることや公開の義務がほぼないためです。しかし、このような攻撃はまとめて、被害者のユーザだけでなく、アプリケーションが騙し取られることも多いのでそれ自体にも、重大な影響を与える可能性があります。つまり、組織は、クリーンアップ費用に対処しなければなりません。

ここでは、ブラウザまたはモバイル アプリケーションにも関連する可能性がある、クライアントを対象としたアプリケーション関連の重大な攻撃について調査します。いずれの場合でも、クライアントは、データを取得、保存および処理するために Web を介してアプリケーションと通信します。認証された接続は、このような攻撃において重要です。

F5 Ponemon セキュリティ調査では、組織に最も壊滅的な被害を与える攻撃の種類（DDoS 以外）について質問しました。その結果、最も多くの回答者（66%）が選んだのは、中間者攻撃、Man in the Browser 攻撃および認証情報の盗難です。その後に 51% で Web 不正行為が続きます。その他クロスサイト スクリプティング（XSS）、SQL インジェクション、クリックジャッキングおよびクロスサイト リクエスト フォージェリ（CSRF）もすべて同様に問題です（図 59 参照）。

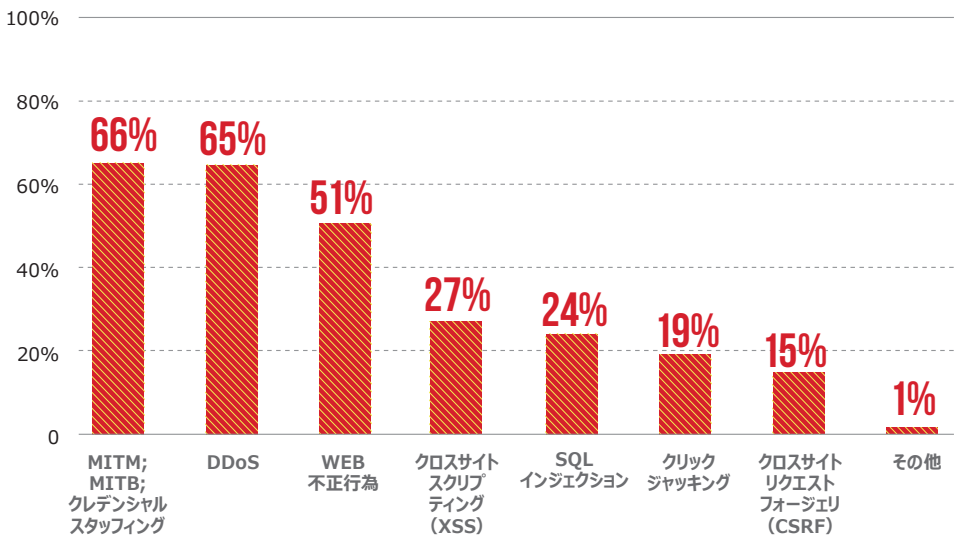
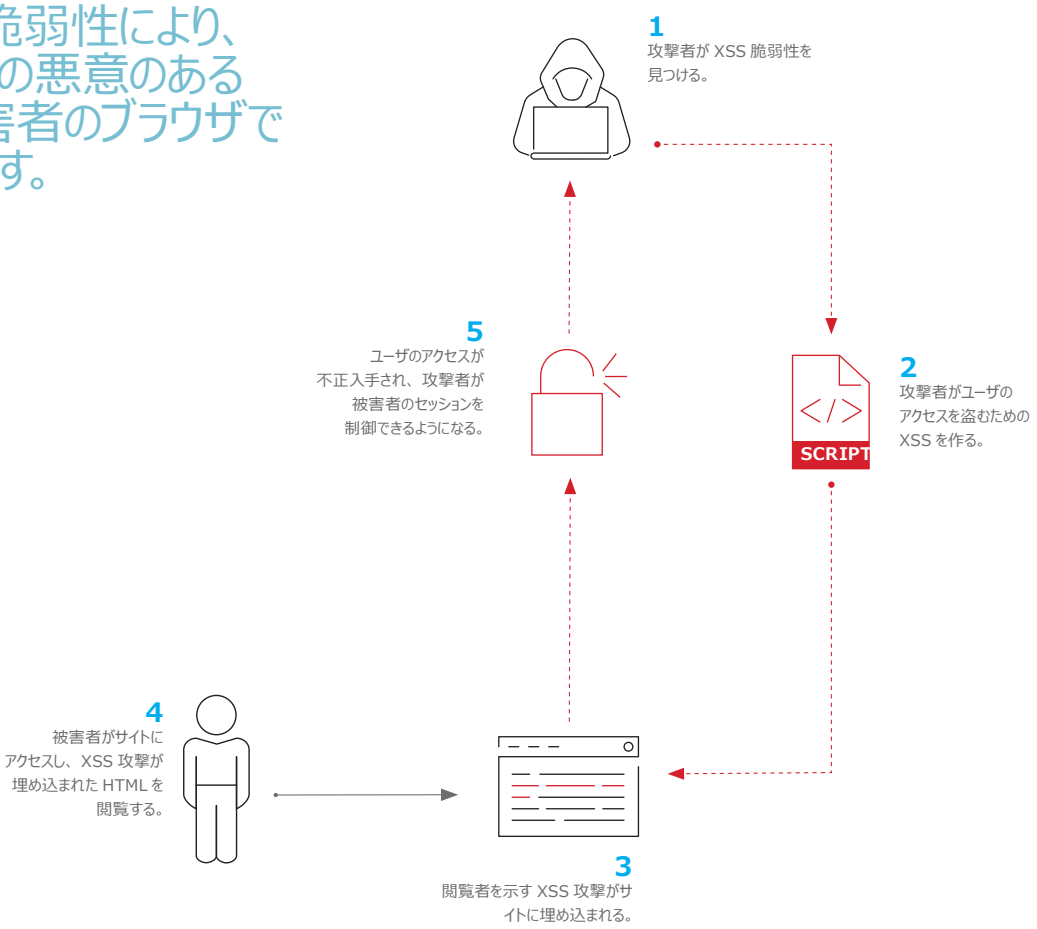



図 59：最も壊滅的な被害を与えるサイバー攻撃（複数回答可）

図 60 : XSS エクスプロイトの経路

クロスサイト スクリプティング(XSS) は、Web サイトの脆弱性により、攻撃者が独自の悪意のあるスクリプトを被害者のブラウザで実行することです。



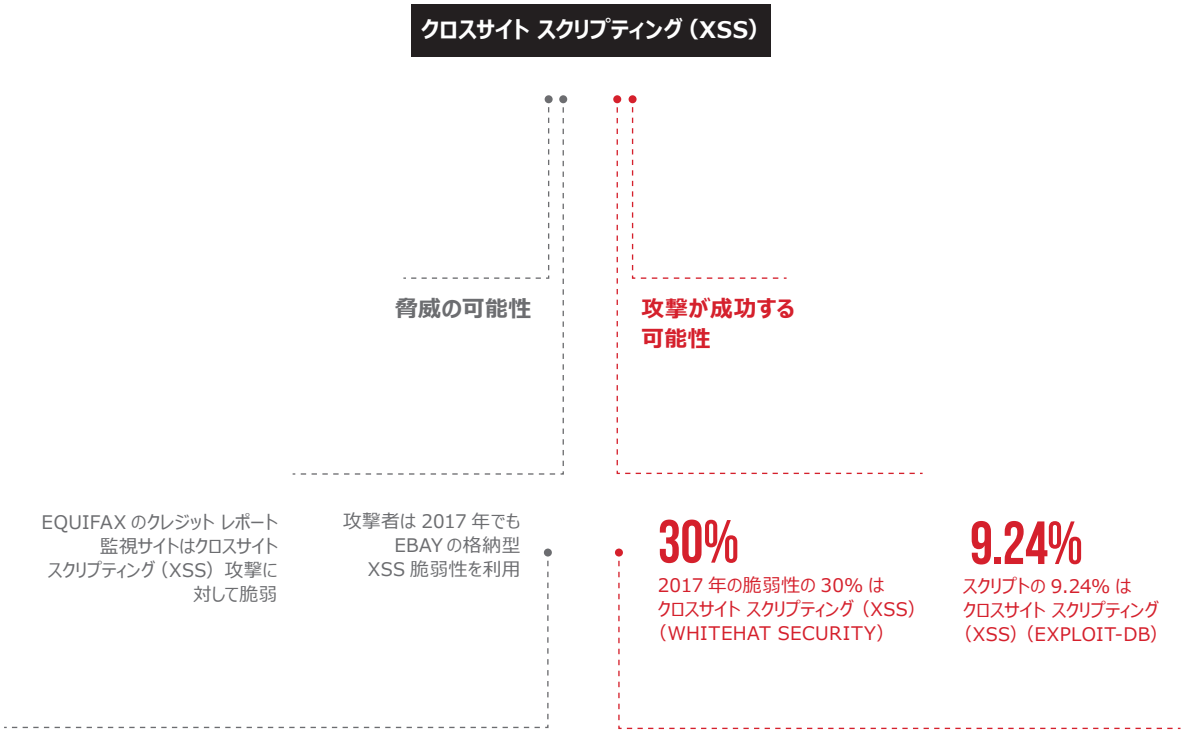

## アクセスをハイジャックするためのスクリプティング攻撃

アプリケーション アクセスをハイジャックするための最も一般的なクライアント スクリプティング攻撃の 1 つは、クロスサイト スクリプティング (XSS) です。XSS とは、被害者がアクセスしている Web サイトの脆弱性により、そのサイトの信頼できる接続内で攻撃者が独自の悪意のあるスクリプトを被害者のブラウザ内で実行できることです (図 60 参照)。XSS が強力である理由は、ユーザが信頼している Web サイトに対して実行されるので、任意のコマンドおよびメッセージがそのサイトにより生成されているように見えるためです。この攻撃では、ページ内のデータに発信元が異なる Web ページがアクセスできないようにする、ブラウザの同一発信元防御メカニズムを回避することもできます。

攻撃者は、XSS を使用して、セッション トークンを盗む、または偽の Web ページを生成して、ユーザの認証情報またはその他の個人情報を取得できます。より高度な XSS 攻撃では、キー ロガーを被害者のコンピュータに仕掛け、入力されるパスワードを監視します。

XSS は、外部ユーザが Web サイトにコンテンツを提供できればどこでも発生できるので、脆弱性の最も一般的なタイプの 1 つになっています。XSS では、ページを表示する Web サイトによりサポートおよび要求される一般的な HTML コマンドが使用されるので、その発見と排除は困難です。

図 61：クロスサイト スクリプティング（XSS）攻撃の可能性




XSS が強力である理由は、ユーザが信頼している Web サイトに対して実行されるので、任意のコマンドおよびメッセージがそのサイトにより生成されているように見えるためです。

2017 年、2 つの非常に大規模な高トラフィックの Web サイトで、重大な XSS 脆弱性が攻撃者に悪用されていたことが分かりました。

- **EBAY：** 攻撃者は、EBAY がまだ修正していない XSS 脆弱性を悪用しています。[56]
- **EQUIFAX：** 最初の情報漏洩が大事ではなかったかのように、すでに被害を受けている消費者を保護すべき EQUIFAX のクレジット レポート監視サイトは XSS 攻撃に対してまだ脆弱です。[57]

これらは、重大ニュースとなった 2 つの有名な事例です。疑う余地もなく、この他にも多くの悪用可能な XSS 脆弱性がインターネット上に広がっています。

## クロスサイト リクエスト フォージェリ（CSRF）攻撃

アプリケーション クライアントに対するもう 1 つのスクリプティング攻撃は、偽造した Web リクエストを介してアプリケーション クライアントをハイジャックする、クロスサイト リクエスト フォージェリ（CSRF）です。攻撃者は、ユーザの認識または許可なしに、そのクライアントにリクエストを送信させることができれば、クライアントはこれに従い、認証を送信します。その結果、ユーザは、現在使用しているアプリケーションで、望まれないアクションを知らないうちに実行します。この攻撃が可能な理由は、Web ブラウザおよびアプリケーション クライアントは、保存されている適切なアクセス認可トークンをすべての Web リクエストで自動的に送信するためです。この攻撃では、その時点のセッション トークンが有効になるように被害者が問題のアプリケーションにログインしている必要があります。

## アプリケーション クライアントに対するマルウェア攻撃

アプリケーションを攻撃するもう 1 つの方法は、マルウェアを使用してアプリケーション クライアントを制御することです。マルウェアがクライアント アプリケーションに感染する方法は多数あります (図 62 および 63 参照)。通常これは、ソーシャル エンジニアリング、フィッシングまたはトロイの木馬アプリケーションを介して行われます。

もう 1 つのベクトルは、通常 Web サイトまたは Web アドバータイジングに埋め込まれているドライブバイダウンロードによるブラウザの脆弱性を介した感染です。この場合、ユーザは何もクリックしなくても感染します。マルウェアは、一度感染すると、ユーザがクライアントにアプリケーション認証情報およびアカウント情報を提供するときにこれらを直接盗むことができます。

悪名高いバンキングを狙うトロイの木馬型マルウェア TrickBot のような一部のマルウェアでは [58]、Web インジェクションを使用して、ユーザに表示される Web ページを変更できます(たとえば、ユーザが入力するデータを盗むための別のフィールドを追加します)。

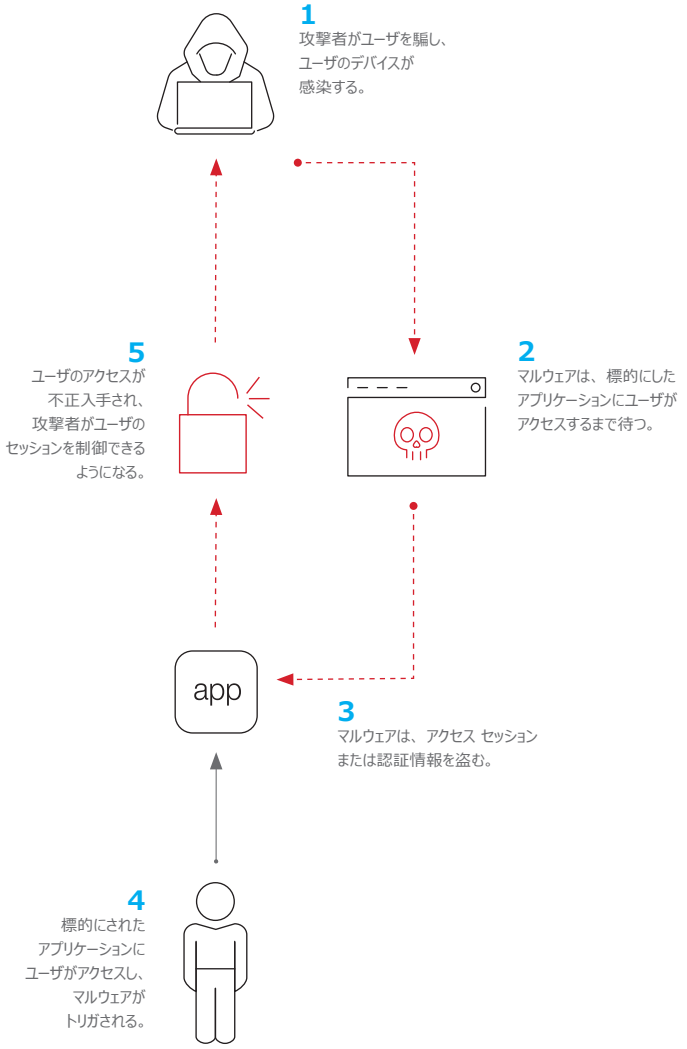



図 62 : クライアント マルウェア感染攻撃の経路

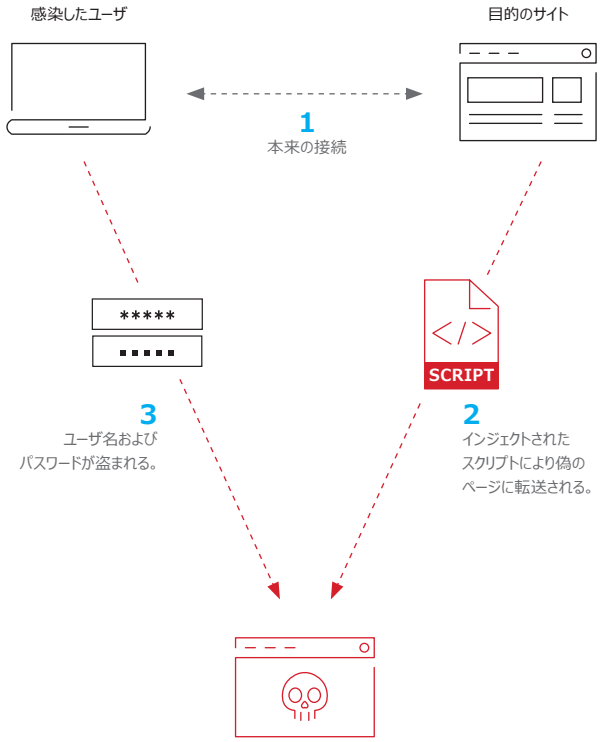



図 63 : マルウェア中間者攻撃の経路

マルウェアが組織に潜入する一般的な経路は、ネットワーク侵入検知システムが検証できない、暗号化されたWebサイト内です。F5 Ponemonセキュリティ調査では、回答者の組織が暗号化されたマルウェアを検知できる能力についてどのくらいの自信があるか調べました。その結果、この問題の重大さが分かりました。

残念なことに、暗号化されたトラフィック内のマルウェアを検知する能力に自信があるかについて、回答者の半数以上（51%）は「あまり自信がない」または「まったく自信がない」と答えていました。「多少自信がある」と答えた回答者はわずか15%でした。「自信がある」または「非常に自信がある」と答えた回答者はわずか34%でした（図64参照）。

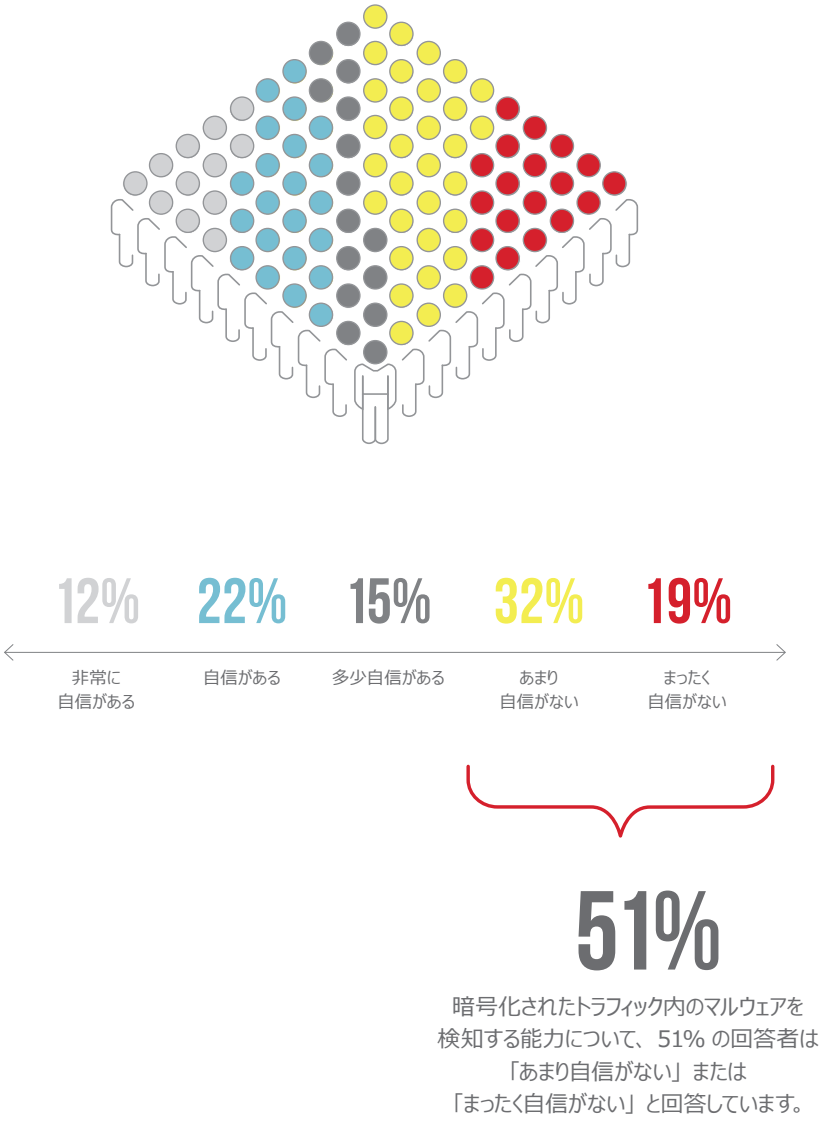



図64：暗号化されたトラフィック検査の確実性


04

# アプリケーションの保護

アプリケーションに対する脅威があることは分かっていますが、これらの脅威は防ぐには何をすべきでしょうか。ここでは、脅威を防ぐためだけの戦略について調査します。この推奨案を紹介する前に、F5 Ponemon セキュリティ調査の回答者が所属する組織について、および回答者がアプリケーションの脅威に対して実施している防御について説明します。

## アプリケーション セキュリティをどのように管理しているか

アプリケーション セキュリティの明確な責任者を 1 人決めることは、保護とコストのバランスと取る上で重要です。しかし、調査回答者の 90% は、アプリケーション セキュリティはセキュリティ オフィス外で運用していると答えています。アプリケーション情報漏洩が発生した、または DDoS 攻撃を受けた場合、誰が責任を取るのでしょうか。

調査回答（図 65 参照）によると、アプリケーション セキュリティ強化における主な障害は、可視化、熟練の担当者またはその他の要素が不足しているかどうかに関係なく、アプリケーション内の現状を把握することです。アプリケーションを検出する CASB など、これに役立つ技術的なツールがいくつかあります。

## アプリケーション脆弱性にどのように対処しているか

この報告書の脅威の説明にあるように、アプリケーションには悪用可能な脆弱性が何層もあり、攻撃者はこれを利用してアプリケーションを破壊または侵入できます。組織は、このような脆弱性を攻撃者より先に見つけて修正しなければなりません。しかし、アプリケーションの脆弱性を検知できるだけの十分なリソースが組織にあるかという質問に対し、調査回答者の 46% は「ない」または「まったくない」と答え、これらを修復できるかという質問に対し、49% がほぼ同様に答えています。多くの組織は、既知の脆弱性をスキャンするだけでは不十分だと分かっています。カスタマイズおよび内部作成したアプリケーションでは、侵入検査やコード レビューなど、より深い（およびより広い）分析が必要です。

脆弱性対策は、庭の草むしりのようなもので、新しい問題が悪用されないうちに解決できるように、頻繁に定期的に行う必要があります。調査回答（図 66 参照）によると、十分な脆弱性管理は、いまだに多くの組織にとっての課題です。

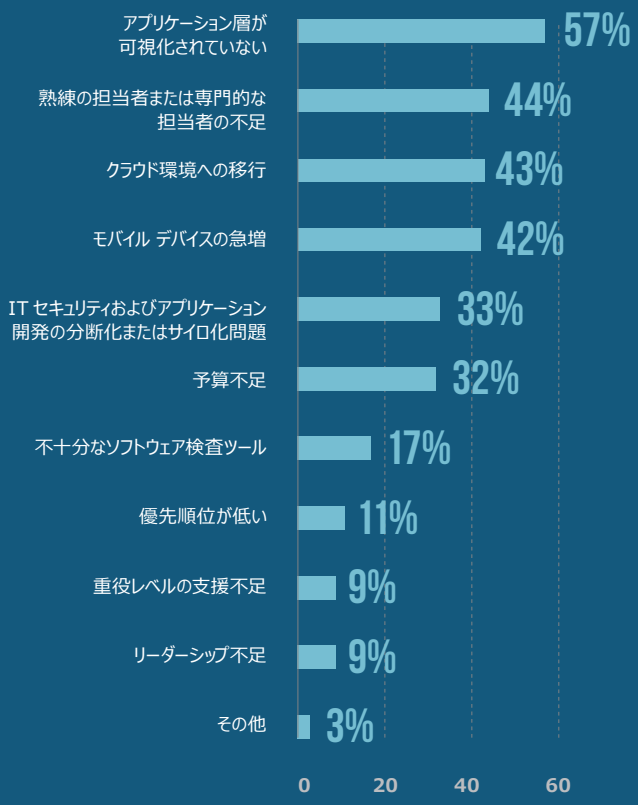



図 65：アプリケーション セキュリティ対策を強化する上での障害（3 つまで選択可）

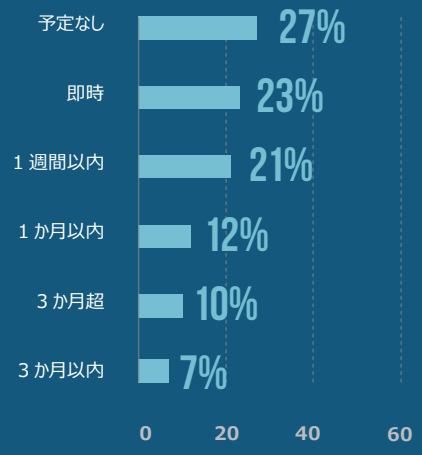



図 66：WEB アプリケーション脆弱性に対するパッチ処理の頻度

## どのようなセキュリティ制御を整備しているか

セキュリティ ホールを発見およびパッチ処理するほか、アプリケーション セキュリティ担当者は、セキュリティ ツールを使用して、攻撃を防ぐ必要があります。調査回答者が使用している主なツールは、Web アプリケーション ファイアウォール（26%）、アプリケーション スキャン（20%）および侵入検査（19%）です。専用のセキュリティ ツールの他、調査回答者は、セグメンテーション（41%）、Linux および Windows ベースのコンテナ（36%）、マネージド クラウドベースのアプリケーション サービスなど、攻撃への抵抗に役立つ多くの運用技術を使用しています。

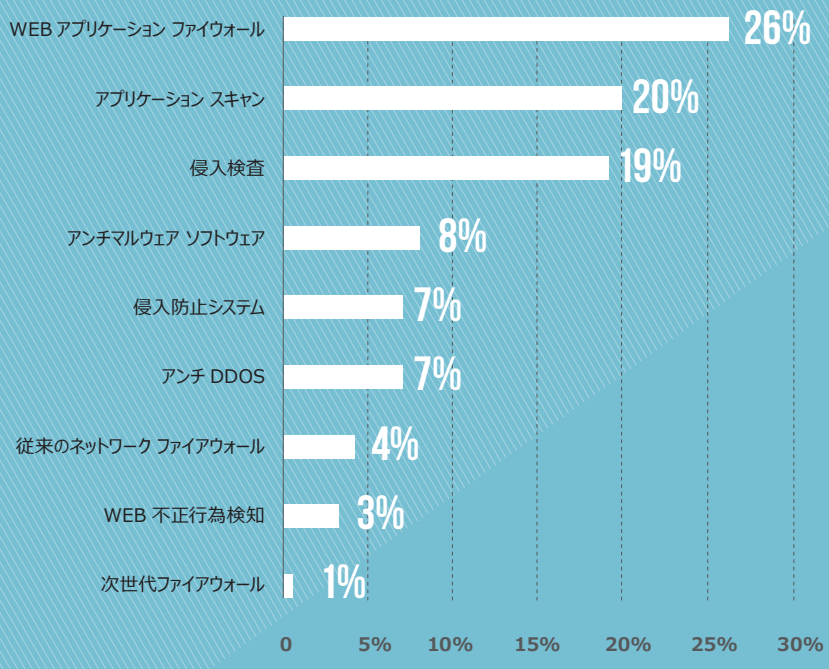
アプリケーションを実稼働環境に導入する前にアプリケーション強化を実施することは、セキュアなアプリケーションを開始する効果的な方法ですが、これをほとんどの場合に行っているのは調査回答者のわずか 36% でした。

サービス拒否攻撃（DoS 攻撃）および災害は珍しいことではないので、アプリケーション セキュリティ担当者は、アプリケーションを常時稼働させるための特殊なツールおよび方法を必要としています。包括的なバックアップ（68%）および強力な DDoS 対策（61%）は、高可用性を実現するために組織が導入している主要な制御で、冗長なアプリケーション（46%）および抵抗検査（43%）がこれに続きます。

インジェクション攻撃が普及し、攻撃者はアプリケーションを改竄することを好むことから、セキュリティ担当者は、コードが密かに変更されないようにする必要があります。ほとんどの組織は、アプリケーションの完全性を確保するために、データ チェックサム（69%）、および監査の変更、アクセス、処理ログ（64%）を導入しています。

組織がどのように自己防衛しているか説明したので、脅威インテリジェンスに基づいた F5 の推奨案について検証します。

図 67：使用しているアプリケーション セキュリティ制御




回答者が使用している主な 3 つのセキュリティ ツールは、WAF、アプリケーション スキャンおよび侵入検査です。

# アプリケーション防御戦略

すべてのセキュリティ担当者が抱いている疑問は、ハッキングされる可能性です。
組織またはアプリケーションが特別に狙われているので、攻撃の標的となっています。
他のすべての人にとっては、ただサイバー犯罪者にとって魅力的すぎる機会であるに過ぎません。

攻撃者は付け入る隙のある防御の穴を探して、絶えずインターネットを捜索しています。攻撃者は、専用の検索エンジンを使うこともあれば、単に Web または IP スペース全体の情報を収集することもあります。つまり、インターネット フットプリントが大きいほど、オンラインのアプリケーションは多くなり、攻撃者からより多くの注目を浴びます。

一般的に、攻撃者は、日和見型と標的型の 2 種類に分かれます。いずれも投資利益率（ROI）を重視し、困難を避け、簡単な獲物から手を付けようとします。しかし、その目的と方法は異なります。

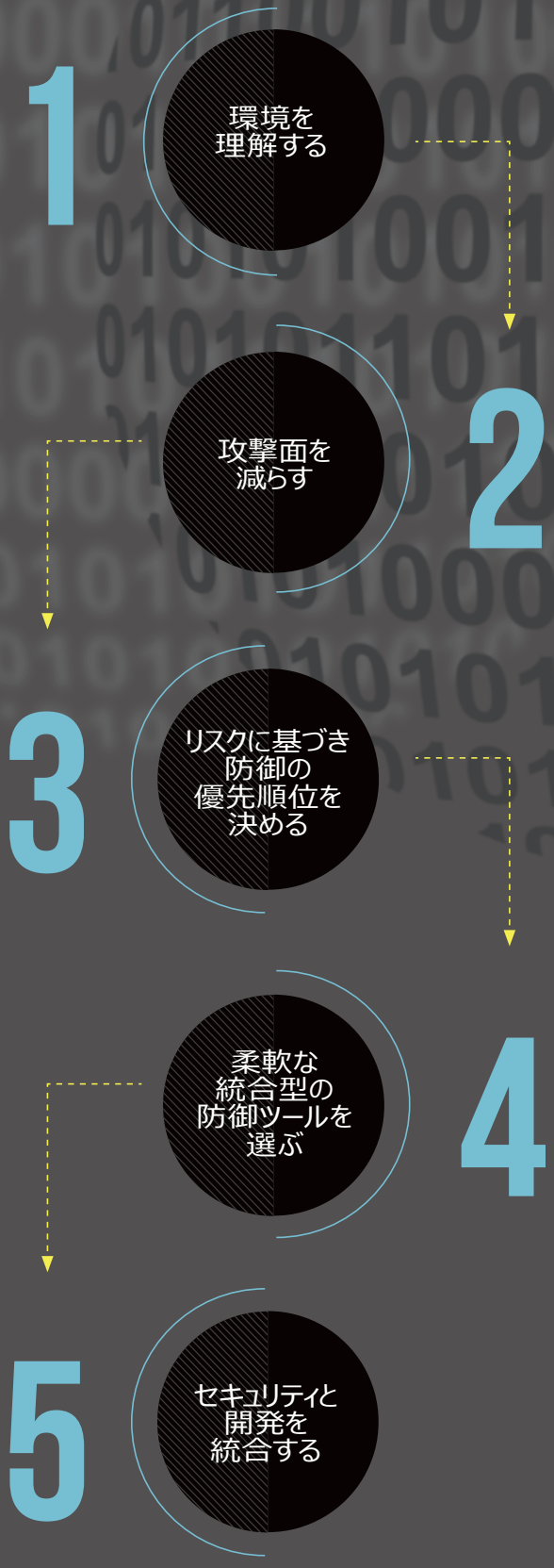
日和見型の攻撃者は、低いコストで高い ROI を狙います。このような攻撃者は、「下手な鉄砲も数撃てば当たる」で、簡単な獲物を探してインターネットを捜索します。ジャッカルが群れの中で弱っている獲物を狙うのと似ています。狙われたのは、個人的な恨みではなく、その時にたまたま遅すぎたためです。このような攻撃者は、簡単に儲けることができるように、あらかじめ準備されたエクスプロイトと既知の効果が証明された方法を使います。撃退された場合は、すぐに次の標的に切り替えます。ただし、このような攻撃者は世の中にたくさんいます。日和見型攻撃により数分ごとにアプリケーションが詮索されていることが確認できない場合は、インターネット接続に何か問題があります。

標的型の攻撃者は、標的を慎重に選びます。このような攻撃者の目的は、スパイ行為または高額な報酬が考えられますが、一度標的にされると追い掛けられる可能性があります。標的型の攻撃者は、高い価値のアセットを求めることで ROI を維持します。つまり、侵入のための時間と労力を惜しみません。実際、標的型の攻撃者は、攻撃に投資すると、攻撃が数回阻止された程度では引き下がらないので、攻撃は持続します（そのため持続的標的型攻撃と呼ばれます）。また、このような攻撃者は、

## 攻撃者は、日和見型と標的型の 2 種類に分かれます。いずれも ROI を重視します。

何らかの技術で阻止しても、すぐに戦略を変えて、それらの防御を破ります。阻止不可能ではありませんが、諦めさせるのは簡単ではありません。幸いなことに、このような攻撃は非常に珍しく、通常、その目標（大抵の場合はお金）を達成する大きな成果だけ追求されます。

図 68 : アプリケーション プロテクションの基本手順




防御戦略を意味があるものにするため、日和見型の攻撃者の脅威も確実に対応できる必要があります。このような攻撃は、シアトルの雨のように、あちこちで発生するので、すべての穴をふさがなければ濡れてしまいます。

その他にも、標的型攻撃に対応する戦略も数種類必要です。標的型攻撃は個人的な恨みで仕掛けられることもあります。従業員の 1 人が、ゲーム フォーラムでエリート ハッカーと手を組む場合もあります。あるいは、顧客の 1 人が重要な政府公認サプライヤで、国家がサプライ チェーンに影響を及ぼそうとしているかもしれません。

いずれにしても、標的型攻撃者に対する防御戦略には、検知と対策を含める必要があります。攻撃を阻止できない場合、少なくても、いつ侵入し、後でどのように事態を収拾できるかを知っておく必要があります。

# 1 環境を理解する

攻撃者はネットワークから何を盗み、または何を破壊したいのでしょうか。まず初めに、何を保護すべきかを知る必要があります。残念なことに、F5 Ponemon セキュリティ調査で、アプリケーション プロテクションにおける問題として最も多く選ばれたのが、「アプリケーション層が可視化されていない」（57%）でした。しかし、可視化はどのようにできるのでしょうか。

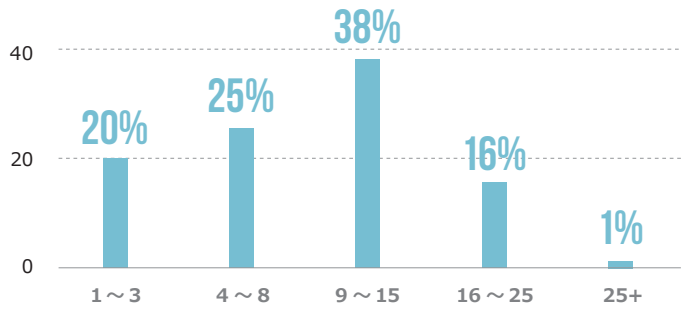
アプリケーションの使用状況の管理は、組織内の重要なビジネスおよび情報フローを引き出すことから始めることができます。これらのフローは、アプリケーションおよび重要なデータ リポジトリに自然と到達します。前述のように、インターネットの存在の脆弱性スキャンは、弱点を探す一般的なツールです。また、これらは、外部からどのアプリケーションが見えるかについての最新の状況を把握できる素晴らしい方法でもあります。

## 開発

組織はアプリケーションを開発または変更する場合（ほとんどの組織がしています）、開発および IT チームと話し合い、アプリケーションに使用されるプログラミング ツールおよび環境を知る必要があります。動的な組織でこの情報を追跡することは困難であり、新しい開発ツールおよびフレームワークに無計画に手を出すことで、これがさらに難しくなることがよくあります。

F5 Ponemon セキュリティ調査では、平均的な組織が何をしているか調べるため、まず、組織が管理する必要があるアプリケーション フレームワークのセットの違いを調べました。フレームワークや環境が増えるほど、これらを保護するために必要になるリソース、トレーニングを受けた担当者およびスキャン ツールは多くなります（図 69 参照）。Web アプリケーション フレームワークまたは環境がごくわずかでも、攻撃者の標的プロファイルが広がるだけでなく、パッチ処理やセキュリティ評価の負荷が大きくなります。賢い CISO であれば、使用するアプリケーション開発プラットフォームの成長および成熟の状況に厳しい目を向けます。

図 69：使用している WEB アプリケーション フレームワークの数




## 外部アプリケーション

組織の外部のアプリケーションを管理するには、Cloud Access Security Broker（CASB）などのツールが非常に役に立ちます。CASB は、ユーザとインターネットの間で、すべてのアプリケーション アクティビティを監視および報告します。CASB は、従業員が使用する主要なアプリケーション（およびそのアクセス方法）を示すだけでなく、シャドー IT アプリケーションの使用状況に関するインサイトを提供します。

使用されているアプリケーション、およびどのアプリケーション（恐らくはすべて）が外部から見えているか把握できたら、組織の運用における重要度に基づいてこれらを評価する必要があります。手順 3 では、リスクに基づいて防御の優先順位を決定します。リスク分析を行うには、何が重要で、何が重要でないか把握する必要があります。これは単独で行わないでください。どれが本当にミッションクリティカルかリーダーと話し合ってください。思っている答えと違う場合もあります。

最後に、これは、一度限りのことではありません。年に 1 回でも足りないかもしれません。稼働中のアプリケーションおよびデータ リポジトリを管理、ユーザが何をする必要があるか監視、および開発環境がどのように進化しているか評価することは常に行ってください。

# 2 攻撃面を減らす

使用されているアプリケーションをすべて把握したら、アプリケーション プロテクションの第一段階を始めます。まず、日和見型の攻撃者に対応します。つまり、明らかなセキュリティ ホールおよび既知の攻撃経路をすべて封鎖します。すべての外部アプリケーション サービスは、組織に対して使用される場合でも、他者へのリフレクション攻撃に利用される場合でも標的になります。

基本的な強化およびロック ダウンを最新にすることも簡単でない場合があります。大規模な現代的な企業では、互換性、サポート、コードの陳腐化、ライブラリ依存関係、品質保証検査、ベンダ バージョニング、運用リソースの不足およびコンプライアンス更新期間といった、パッチ処理を複雑にする要素が多くなります。

ここでも、これに役立つセキュリティ ツールがあります。優れた Web アプリケーション ファイアウォール（調査回答者に最も選ばれたツール）は、パッチ処理の時間を稼ぐことができます。これは、発見されたセキュリティ ホールの新しいシグネチャを自動的に作成する静的または動的アプリケーション セキュリティ検査（SAST/DAST）ツールにリンクすることによる「仮想パッチ」で行われます。また、優れた WAF は、アプリケーション トラフィックを分析して、既知のエクスプロイト攻撃を検知および阻止します。

図 70: WEB アプリケーション ファイアウォールによる保護

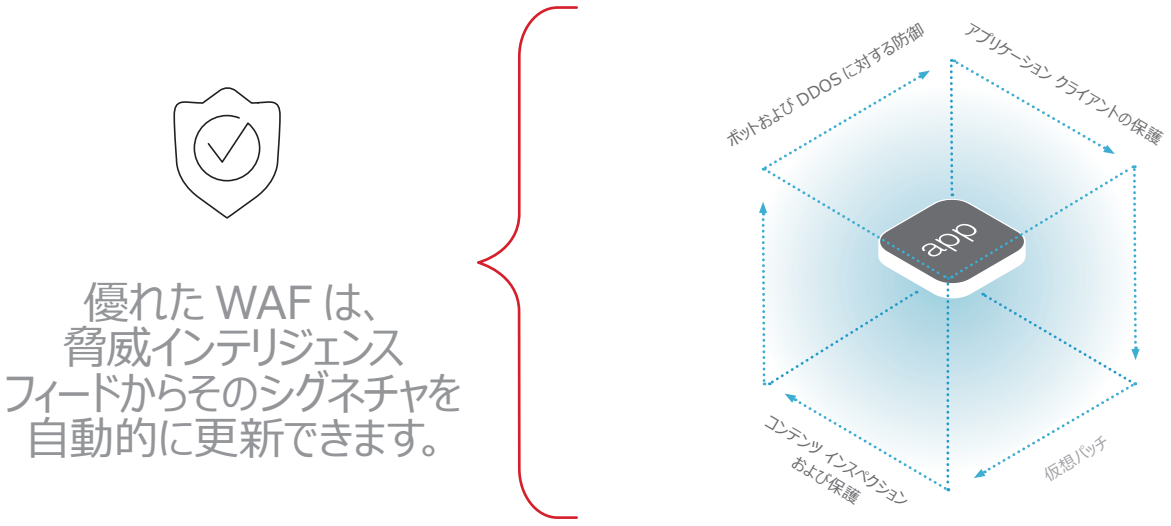
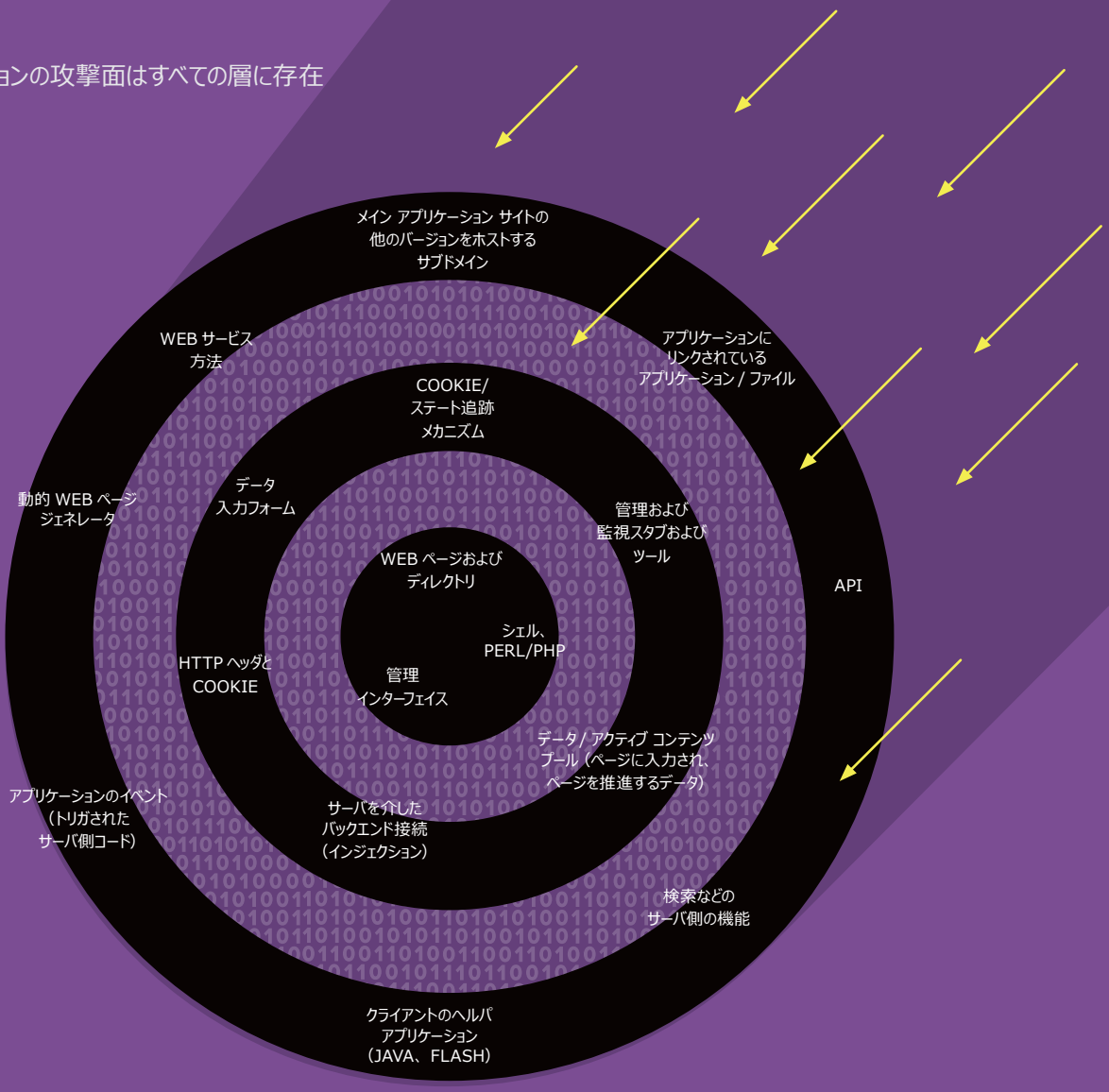


図 71：アプリケーションの攻撃面はすべての層に存在




WAF は、環境の脅威インテリジェンス フィードおよび脆弱性スキャンからシグネチャを自動的に更新できるので、新しいエクスプロイトが公開されたときに慌てずに済みます。これにより、すぐにパッチ処理しなければならないという時間的圧力が緩和され、運用チームは余裕を持って、修正を適切に検査してから公開できます。このような WAF 機能は、日和見型の攻撃者が仕掛ける明確な攻撃を保護する上で優れています。しかし、インターネット フットプリント全体が標的なので、このようなソリューションは全体的に導入する必要があります。セキュリティ チームが WAF 境界で必要なセキュリティ ブロック機能を有効にしていないために発生している回避可能なセキュリティ インシデントは数多くあります。

**オンラインをできるだけ少なくする**

攻撃者は、インターネットに公開されていれば何でも攻撃するので、必要最低限のものだけオンラインにすることが優れた戦略です。

それ以外のものは、厳しいアクセス制御およびファイアウォールで管理する必要があります。ネットワーク境界 [59] を超えた世界でこれが簡単なタスクであるとは思いませんが、やるべきではないという意味ではありません。

特定のアプリケーションまたはアプリケーション コンポーネントが攻撃される可能性は、情報漏洩のリスクを理解する上で重要です。さらに、攻撃の可能性に関して、公開されているリソースの脆弱性と影響はすべてが同じというわけではありません。全体として、これは「攻撃面」と呼ばれ、外部ユーザが利用できる運用のすべてのリスニング サービスと方法の総数を表します。すべての条件が同じだとすると、アプリケーション開発者がアプリケーションにライブラリとサービスを追加すると、そのフットプリントは膨張および増加します。これらのサービスの大部分は、図 71 に示すように、全体のあらゆる層のアプリケーション サービス内にあります。

### セグレゲーションとパーティション

アプリケーションの数を把握し、何が重要であるか理解したら、アセット グループ間にいくつかの内部的な障壁を設けます。優先順位の低いアプリケーションに不正アクセスされるのと、これがより高い優先順位のシステムへの経路として攻撃者に利用されるのは別の問題です。

このセグレゲーションは、内部開発したアプリケーションのコード内でも行うことができます。一般に、信頼できないシステムに公開されるアプリケーション機能の量を減らし、残りのシステムからこれらの機能をパーティションする必要があります。これは、一般的にアプリケーションおよびデータへの完全な経路となる API およびデータベースのような強力なインターフェイスに特に当てはまります。これらの機能は、最小権限の原則に従いアクセス制御する必要があります。セキュリティ侵害によりアプリケーション全体への自由なアクセスを提供するコードまたは機能は、隔離して慎重に監視する必要があります。これは、コード上や、サーバ隔離、サンドボックス、ユーザの最小権限の規則およびファイアウォールでも実行できます。

## 3 リスクに基づき防御の優先順位を決める

アプリケーションがハッキングまたは破壊されると、コストがかかりますが、これと防御の運用および資本コストとのバランスを取る必要があります。ここでリスクについて考えます。リスク対策のポイントは、不要なコストを回避すると同時に、使用できるリソースを最大限活用することです。リスク分析は完璧である必要はありません。適当な「ベスト プラクティス」リスト [60] から適当に制御を選択する、または現時点での重大な脅威 [61] を気にするだけでなければ十分です。リスク分析は、攻撃者が何を求め、組織にとって何が重要かに基づいて行う必要があります。効果的かつ効率的にするため、最適な防御を整備し、これらの最大のリスクを軽減します。

このために、データによりリスク戦略を推進します。攻撃者が何を求めているか理解することが、リスク分析で重要です。この報告書、および F5 Labs の進行中の脅威研究では、攻撃者の行動に関して十分に考えるべき多くのことが提供されています。組織はさまざまなので、狙われやすい攻撃面を知っておく必要があります。

### コードのリスクを理解する

アプリケーションを内部開発する場合、これらのアプリケーションが悪用される可能性を必ず理解しておく必要があります。新しいアプリケーションは組織になかったものなので、それぞれのセキュリティは検査されていません。検査には、内部スキャナやコード レビューなどさまざまな方法があります。OWASP Dependency-Check ユーティリティ [62] は、外部ライブラリを使用することからコードに潜んでいる可能性がある明確な欠陥を発見できます。この検査を第三者に委託すると、知識に基づいた独自の見解を得ることができるので、欠点がより明確になります。この情報に基づき、内部開発したアプリケーションのリスクを適切に評価できます。アプリケーションが顧客により使用される場合、ビジネスを持続するために顧客のデータおよび独自のデータを保護する責任が生じるので、この検査の義務はより強くなります。

# 4 柔軟な統合型の防御ツールを選ぶ

アプリケーション プロテクションを適切に行うために大量の技術制御ソリューションは必要ありません。ソリューションの固定費のほかに、制御を導入、実行および保守する運転費もかかります。

必要なことは、リスクと、防止、検知および攻撃からの回復の機能を重視することです。これは、チームが既存および新しい脅威に使用できるいくつかの柔軟かつ強力なソリューションを導入することで実施できます。前述のように、一般的な 3 つの主要な技術制御は、Web アプリケーション ファイアウォール、脆弱性スキャン ソリューションおよび CASB です。

一般的な 3 つの技術制御により、リスク、防止、検知および回復に集中しやすくなります。

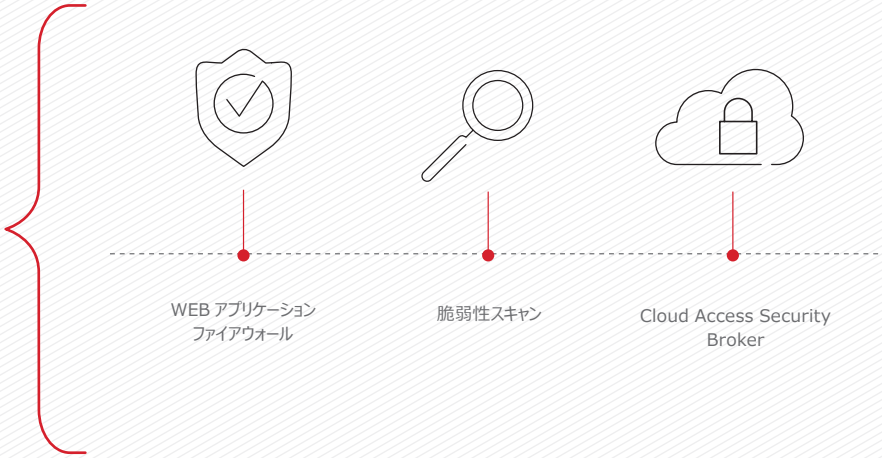



図 72：防止、検知および回復のための技術制御

これらの 3 つのツールは、十分に強力かつ柔軟であれば、ネットワーク アクセス制御、脆弱性管理、インベントリ、リスク分析およびラングリング認証をサポートできます。

DDoS の脅威、アプリケーション クライアント、DNS およびネットワーク トランスポートに必要なツールもあります。これらについては、この項の後半で詳しく説明します。

# 5 セキュリティと開発を統合する

より効率的な方法は、新たに発見された問題の修正をバックポートするのではなく、最初からアプリケーションのセキュリティ脆弱性ができないようにすることです。このためには開発チームとの連携が必要です。堅実な第一歩として、Web およびモバイル アプリケーションの生産におけるすべての関係者と OWASP Top 10（図 73）を共有して、問題を認識させます。

脅威およびアプリケーション エクスプロイトがどのように機能するかに関する実用的な知識を得ることで、開発者は、セキュリティ チームに指摘されなくてもアプリケーションを保護できます。これにより、最終製品のセキュリティ ホールが減るだけでなく、新しいセキュリティ ホールをすぐに発見および修正できます。F5 Ponemon セキュリティ調査によると、回答者の 66% は、スキルまたは資格のあるアプリケーション開発者が不足していることで組織のセキュリティ体制に悪影響があると感じています。

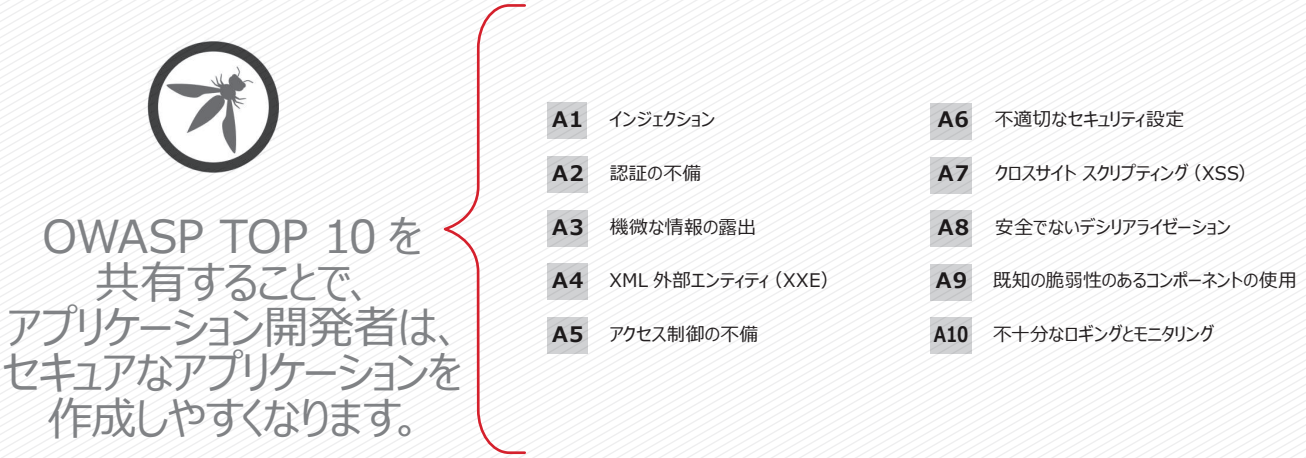



図 73：OWASP TOP 10

もう 1 つの強力な開発セキュリティ対策は、機密アプリケーション データの保存時の暗号化です。アプリケーションは、情報漏洩およびデータ盗難の危険に常にさらされているので、重要なデータ フィールドは、適切な暗号化キーなしでは読み込むことができないようにしておく必要があります。また、暗号化復号化キーを保護することも重要です。アプリケーションが攻撃者に乗っ取られた場合でも、簡単にキーをコピーして、盗んだデータを復号化できないようにする必要があります。OWASP CRYPTOGRAPHIC STORAGE CHEAT SHEET[63] に、この管理方法について実践すべきいくつかの提案があります。

## DOMAIN NAME SYSTEM サービスの保護

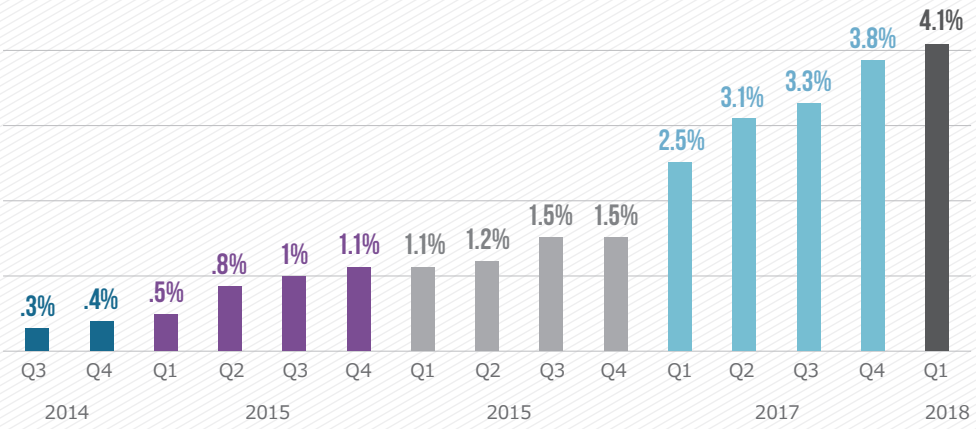
重要なアプリケーション インフラストラクチャ サービスのように、DNS サーバは、アクセス制御により保護し、高可用性を維持する必要があります。アクセス制御は、強化、パッチ処理およびファイアウォールで形成できます。ファイアウォールは、53 以外のすべてのポートをブロックするだけでなく、DNS 固有のエクスプロイトおよび DDoS 攻撃もブロックする必要があります。DNS サーバは、冗長および緊密な監視により高可用性を維持できます。

## トランスポート層の保護

インターネット上の Web サイトの半分以上は、暗号化を使用していますが、これは十分な暗号化なのでしょうか。トランスポート層で強力な暗号化を保証するためには、まず、最新の許容基準に遅れないように従い、これらの基準から逸脱していないか監視します。

アプリケーションでの MitM 攻撃の可能性を減らすには、Web サーバが HTTP Strict Transport Security（HSTS）と呼ばれる HTTP セキュリティ ヘッダを使用する必要があります。しかし、F5 2017 TLS Telemetry report によると、HSTS はサーバ レベルで広く導入されていません（10%）。

図 74：HTTP STRICT TRANSPORT SECURITY（HSTS）の普及率




ドメイン名の盗難または証明書の不適切な発行の脅威に対処する主なソリューションは、Certificate Transparency です。Certificate Pinning は、サーバ証明書をクライアント（ブラウザまたはモバイル クライアント）に埋め込むことで、クライアントはシリアル番号により正確な証明書を要求します。検索可能な証明書リポジトリを使用し、認証局にその発行するすべての証明書の公開を求める Certificate Transparency（CT）システムがあります。これにより、サイト所有者は、リポジトリを定期的に検索して、そのサイトに発行された証明書を確認できます。

詳しくは、https://crt.sh/ または https://censys.io/ をご覧ください。この CT システムは、不正発行された Symantec Google 証明書を識別した CT なので、少なくとも Google には、すでに何らかの利益を生み出しています。

## DDOS に対する保護

多くのアプリケーションでは、サイトまたはそのコンポーネントを誰も攻撃しないという未熟さが設計に含まれています。しかし、インターネットのすべてのアプリケーションまたはサイトは、攻撃の標的になる可能性があるだけでなく、他者への DDoS 攻撃に利用できるか綿密に調べられています。そのため、不明瞭性だけで隠蔽された不安定なインフラストラクチャは遠い昔のことです。重要なコンポーネントは、強化およびアクセス制御され、弾力的にフェイルオーバできる必要があります。

組織は、DDoS 攻撃が必ずあることを前提に、適切なリスク分析を行う必要があります。潜在的な影響に基づいて、アプリケーションをネットワーク層、アプリケーション層およびサポート インフラストラクチャで DDoS 攻撃から保護する必要があります。アンチ DDoS ソリューションのレベルは、オンプレミス スクラビング設備からホステッド ソリューションまでさまざまです。重要なことは、アプリケーションへのリスクおよび可能性のある脅威に基づきソリューションを調整することです。

## アプリケーション クライアントの保護

アプリケーション クライアントの保護については、公開するアプリケーションにアクセスする顧客と、インターネット上のアプリケーションにアクセスする組織の内部ユーザの 2 つのクラスを考慮してください。まず、後者の組織のユーザについて説明します。

ユーザの保護は、信頼できるアクセス制御の実装から始めます。パスワードは常に認証における必須のソリューションです。パスワードは安くて簡単に実装できますが、必ず弱点があります。前述のように、F5 Ponemon セキュリティ調査により、組織の 74% がいまだにアプリケーション固有のユーザ名 / パスワードを認証に使用していることが分かりました。重要なアプリケーションでは、組織は、フェデレーション アイデンティティや多要素認証などのより強力な認証ソリューションに移行し始めています。

**74%**

組織の 74% はいまだにアプリケーション固有のユーザ名 / パスワードを認証に使用しています。

残念なことに、ユーザがアクセスしている可能性があるアプリケーションの一部は、ユーザ名 / パスワードの認証のみサポートされている可能性があるので、ユーザはパスワード推測や認証情報の盗難などのアクセス攻撃のリスクに常にさらされています。前述の CASB のような技術ソリューションは、保護の第一歩に適しています。CASB は、ユーザが使用しているアプリケーションを明確にするだけでなく、その主な目的として、外部アプリケーションの認証を統合および強化することでアクセス制御を管理します。

また、CASB は、管理されたデータまたは機密データが不適切な場所に保存されないようにするために、組織から外部アプリケーションにアップロードされるデータを分類およびフィルタリングするときにも役立ちます。さらに、許可（および強化）されたクライアントだけに外部アプリケーションのアクセスを許可することで、クライアント自体をロック ダウンできます。

## 顧客のアプリケーション クライアントの保護

ユーザの保護と同様に、顧客のアプリケーション クライアント セッションの保護にも取り組む必要があります。これらも同様に、盗まれた認証情報のスタッフィング、フィッシングまたはマルウェア攻撃などのクライアント攻撃を受ける可能性があります。より強力で柔軟な WAF システムでは、ボット攻撃、ブルートフォース（総当たり）攻撃および疑わしい場所からのログインを検知することで、アプリケーション クライアントの保護を提供できます。この簡単な検証は、アプリケーションにアクセスする顧客の保護を強化する優れた方法です。

ユーザの保護は、
信頼できる
アクセス制御の
実装から始めます。

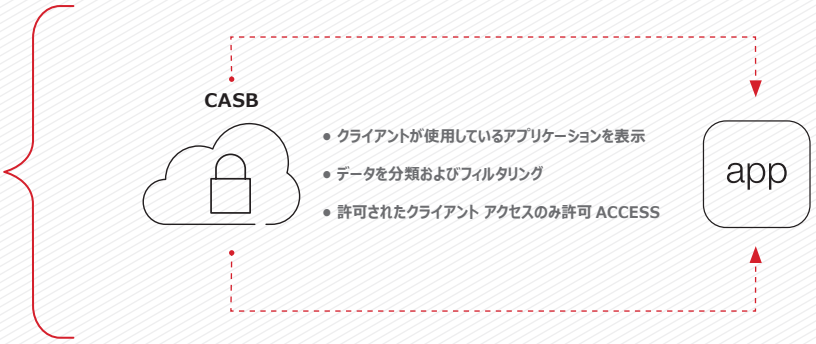



図 75：CLOUD ACCESS SECURITY BROKER の機能

トランスポート層暗号化でもこれらのクライアント セッションを保護できます。Web アプリケーションでトランスポート層暗号化を適切に行うことは、プライバシーだけでなく、MitM 攻撃による転送への介入を防ぐ上でも重要です。アプリケーションの信頼性の検証を補う上で、組織がサイトの立証された合法的な組織であることをより強く断言するために、Extended Validation(EV)証明書 [64] を使用することを考慮してください。EV は多少コストがかかるかもしれませんが、顧客に提供するアプリケーションにおいて、コストに見合う信頼性を得ることができます。最新のアプリケーション クライアントでは、EV がサポートされ、使用される場合はユーザに通知が提供されます。

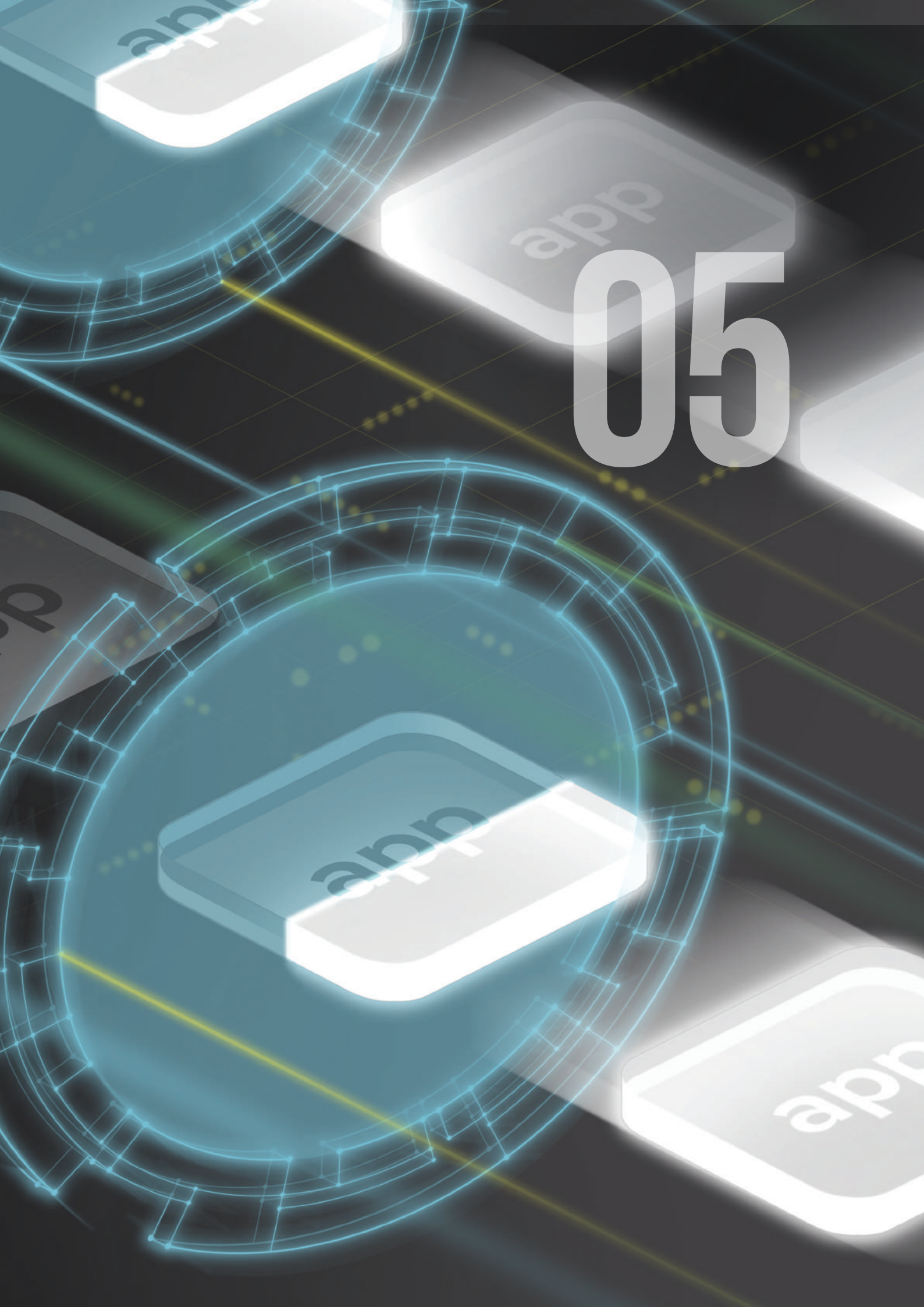
セッション cookie を HTTP のみ [65] に設定し、ドメインを制限して、X フレーム オプションを DENY に設定することで、トランスポート層の保護を強化することもできます。これは、クリックジャッキングおよび認証情報の盗難を防ぐ上で役立ちます。Web サーバまたはゲートウェイ設定を簡単に微調整するだけで、アプリケーションに導入できます。

## 攻撃タイプおよび防御ツールの概要

以下の表に、防御ツールとその効果が発揮される既知のアプリケーション攻撃の組み合わせを示します。

| 攻撃のタイプ | 影響のある層 | WAF | 脆弱性スキャン | アンチ DDOS | MFA | アプリケーション開発セキュリティ | 保存データの暗号化トレーニング |
|---|---|---|---|---|---|---|---|
| インジェクション | アプリケーション サービス | ✓ | ✓ | | | ✓ | ✓ |
| デシリアライゼーション攻撃 | アプリケーション サービス | ✓ | ✓ | | | ✓ | ✓ |
| 機能の悪用 | アプリケーション サービス | | | | | ✓ | ✓ |
| API 攻撃 | アクセス | ✓ | ✓ | | ✓ | ✓ | ✓ |
| アカウント アクセス攻撃 | アクセス | ✓ | ✓ | | ✓ | ✓ | ✓ |
| TLS 攻撃 | TLS、ネットワーク | ✓ | | ✓ | | | ✓ |
| DNS ハイジャック | DNS | ✓ | | | | | |
| ハイブリッド DDoS | すべてのアプリケーション層 | ✓ | | ✓ | | | |
| リフレクション DDoS | すべてのアプリケーション層 | ✓ | | ✓ | | | |
| TLS サービス拒否 | TLS | | | ✓ | | | |
| アクセスをハイジャックするためのクロスサイト スクリプティング攻撃 | クライアント | ✓ | ✓ | | ✓ | ✓ | |
| マルウェア攻撃 | クライアント | | | | ✓ | | |

表 1：攻撃とそれに対処できるソリューションの組み合わせ


05

# アプリケーションプロテクションの将来

アプリケーションの現代の脅威ランドスケープを調べていますが、技術の世界は常に進化しています。実際、現在訪れている変化のいくつかは、ただの進化ではなく革命的な変化です。これらの変化により、アプリケーションの脅威および危険への変化も訪れています。そのため、変化に遅れないように適切に対応できるように準備する必要があります。

確実に予想できる将来の形は、持続的成長とアプリケーションへの依存です。アプリケーションは生活に浸透し、これまで以上に重要になっています。


# アプリケーション セキュリティ

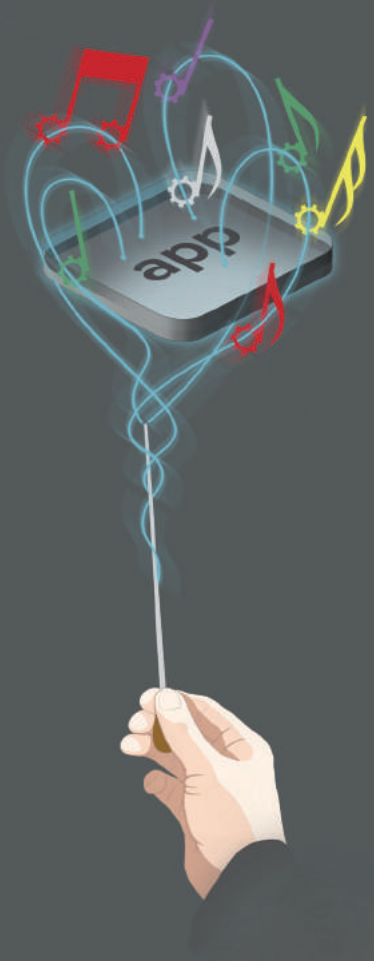
この報告書全体で指摘しているように、アプリケーションはもはやモノリシックなソフトウェア プログラムではなく、さまざまなスクリプト、ライブラリ、サービスおよびデバイスで構成される群体生物です。将来、セキュリティ ツールがこのパラダイムに追い付くことを望んでいます。

敵に単独で立ち向かうことができるだけ強力なソフトウェア コンポーネントおよびフレームワークにより、アプリケーション セキュリティがこの革命を受け入れるようになることを期待します。API またはコード ライブラリをアクセス、制御およびフィルタ処理すべきとは考えていません。しかし、かなり多くのセキュリティがすぐに使えるようになるでしょう。将来的には、開発者は、現在の壊れやすく、過度に依存するエコシステムとは大きく異なり、より優れた方法で選択できるセキュア コンポーネントおよびフレームワークを利用できるようになります。

このような傾向のほかに、サーバレス コンピューティング、アプリケーション セキュリティのアウトソーシングおよび TLS セキュリティの必然的な改善に変化の兆しがあります。

## アプリケーション プロテクションの将来において注意すべき主な傾向

- 「デフォルトでセキュア」なアプリケーション フレームワーク
- 標準化された形式でセキュリティ ステータスおよびイベントを報告できるアプリケーション コンポーネント
- 安全かつ便利な方法で、生産中のアプリケーション コンポーネントを継続的に検査できるセキュリティ スキャナ




# サーバレス コンピューティングおよびアプリケーション

サーバレス コンピューティングは、新しいコンピューティング モデルで、開発者は、アプリケーションが稼働するサーバを気にすることなく Web アプリケーションを構築できます。サーバレスといっても、アプリケーションはサーバ上で稼働するので、分かりづらい用語ではありますが、この概念は、継続的な進化の一部で、開発者がビジネス ニーズに必要なものだけを構築できるようにサポートします。

サーバレス アプリケーションは、何もなければクラウド プラットフォーム内で休眠状態になっている関数をトリガするイベントとのユーザ インターフェイスで主に使用されます。アプリケーションのコードおよびスクリプトは、関数およびサービスを直接トリガする API と接続します。これは、コードがサーバ内で実行し、他のサーバを呼び出す従来の方法とは正反対の方法です。サーバレス開発のメリットは、アプリケーション ソリューションの提供に、より速く合理的に集中できることです。また、サーバレス開発は、サーバおよびシステムの従来のアプリケーション スタックからアプリケーションが離れるので、より柔軟で拡張性にも優れています。

しかし、デメリットもあります。サーバレス アプリケーションによりすべてのアプリケーション セキュリティ問題が解決されるわけではありません。実際、サーバレス アプリケーションでは、サービスおよび関数が増えることでフットプリントが大きくなります。つまり、悪用または破壊される可能性のある攻撃面が広がります。公開される関数が増えるほど、機能を悪用する攻撃の脅威も増加します。また、API への関数呼出しに依存することで、アクセス制御およびトランスポート暗号化がますます必要になります。サーバレス アプリケーションでは、XSS、CSRF およびインジェクションなどの問題はなくなりません。さらに、孤立した多様なシステムにアプリケーションがさらに分散するので、インベントリおよび監視がさらに複雑になります。

## サーバレス アプリケーションで注意すべき重要な脅威

- 特にサーバレス API に対する、アクセス制御攻撃
- さらに拡大および分散する依存インフラストラクチャに対する DDoS 攻撃
- XSS、CSRF およびインジェクション攻撃などの主要なアプリケーション エクスプロイト
- 拡大した連結ネットワーク メッシュに対するトランスポート層攻撃


# アプリケーション セキュリティのアウトソースの増加

F5 Ponemon の『*The Evolving Role of CISOs and their Importance to the Business*』 報告書 [66] では、 58% の CISO が、 IT セキュリティ運用の一部をアウトソースしていると答えています。 最新の F5 Ponemon セキュリティ調査では、 回答者の 44% は、 アプリケーション セキュリティ対策を強化する上での主な障害として、「熟練の担当者または専門的な担当者が不足している」ことを挙げています。

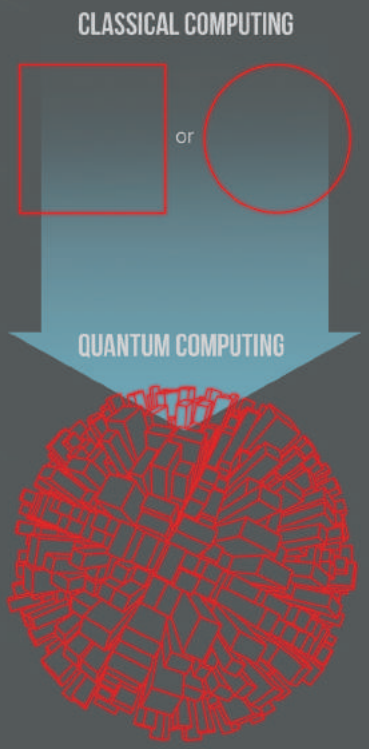
同調査によると、 一般的な組織は、 平均 756 種類の Web アプリケーションを使用し、 このうち 34% は重要だと考えています。これらの傾向とアプリケーション攻撃の増加傾向を合わせると、アプリケーション セキュリティをアウトソースする組織は増加すると考えられます。 つまり、 アンチ DDoS または Web アプリケーション セキュリティ モニタリングなどのアプリケーション セキュリティ機能をアウトソースするか、 製品の一部としてこのようなアウトソースされたサービスを提供するホステッド プラットフォームに移行する可能性があります。

ほとんどの一般的な組織とは異なり、 アウトソーシング セキュリティ企業には、 24x7x365 の運用のための強力なセキュリティ ツールに投資できるリソースがあり、 高度なトレーニングを受けたセキュリティ スタッフがいます。 セキュリティ サービスを顧客に提供することが、 このような組織の存在理由なので、 防御能力は顧客満足度と収益に直結します。 そのため、 セキュリティの改善およびコンプライアンスの証明は、 セキュリティ チームが直面する従来の抵抗とは対照的に、 ビジネスの成功となります。

さらに、 多くの組織のセキュリティを管理することで、 アウトソーシング セキュリティ企業は、 最も効果的な制御に関する優れた脅威インテリジェンスとフィードバックを備えています。 これらの組織は、 新しい脅威が勢いを増し、 より多くのインターネット人口を脅かし始めるとすぐにその傾向に気付くことができます。 これは、 F5 が、 この報告書のように、 タイムリーで詳しい報告書を提供できる大きな理由です。

### セキュリティをアウトソースするときの主な考慮事項

- 組織のセキュリティ要件と比較した能力とセキュリティへの献身
- 社内では実現できない可能性がある具体的なアプリケーション セキュリティ ニーズ
- 独自の具体的なニーズを満たすアプリケーション セキュリティを提供できるアウトソーシング セキュリティ企業の実績と経験




# トランスポート層セキュリティにおける今後の課題

早いうちに、暗号コミュニティは、より安全かつ高速な暗号化プロトコルである TLS の最新バージョン 1.3 の採用に取り組むことになります。以下を含め、このプロトコルに関するほぼすべてが変わっています。

- Forward Secret 暗号のみ許可。つまり、多種多様な楕円曲線および Diffie-Hellman キー交換が可能になります。
- セッションベースのサーバ キャッシュの排除。
- クライアントが最初の TLS ClientHello でデータを送信するゼロ ラウンド トリップ（高速起動）TLS。
- ChangeCipherSpec などこれまでは安定していた TLS メッセージおよびフィールドの削除。

これらすべての変更（一部は表面的）により、TLS 1.3 は、これまでのところ、インターネット販売が難しくなっています。削除されたフィールドの再導入など、一部の変更は継続中です。TLS 1.3 への移行を複雑にしている理由として、現状では TLS 1.2 があれば十分だという事実があります。TLS 1.2 には特に問題がなく、プロトコルの脆弱性もありません。そのため、TLS 1.3 への移行を急ぐ理由がなく、設計者がこれを理解するまで多少の時間があります。長期的には量子コンピュータ[67] の不安も Transport Layer Security を脅かします。

実用的な量子コンピュータを約 4,000 量子ビットで構築できれば、世界中すべての TLS 接続を破壊できます。楕円曲線または Diffie- Hellman キー交換を使用している場合でもです。しかしこれは本当に「もしも」の話です。現在の現実の量子コンピュータが構築できる量子ビットはごくわずかです。私たちの見解では、Transport Layer Security は、量子コンピュータ以外からさらに激しい衝撃を受けると予想していますが、これらの衝撃が何かはまだ分かりません。

## トランスポート層セキュリティで注意すべき重要な脅威

- TLS 1.3[68] のブラウザ サポート
- ネットワーク暗号化[69] に関する主なコンプライアンス標準
- 量子コンピューティングの進化[70]

# 結論とさらなる問題

ここまで読み終えて、Web アプリケーション セキュリティは近寄り難いという印象を得たかもしれません。しかし、簡単に始める方法はいくつかありそうです。また、Web アプリケーション ファイアウォール（WAF）を使用していない少数派であれば、これは最初に調査すべきソリューションです。適切なトレーニングを実施して、アプリケーション プロテクションに役立つように WAF のすべての機能を活用してください。私たちの WAF 設定ガイドの言葉を引用すれば、重要なのは「小さくても何かを始めてください」。[71]

この報告書の目的は、「アプリケーションを保護するためにできる最も重要なこと」を伝えることです。私たちはできるだけ、CISO がアプリケーション セキュリティに関して抱えている問題を予測し、これに答えました。この報告書を使用することで、アプリケーション セキュリティ戦略のギャップが埋まり、リスクと防御に関する実りある会話が進めば幸いです。

その後は何をすべきでしょうか。F5 は、アプリケーションの脅威およびリスクの調査を続け、情報を提供します。ご期待ください。

# 付録

## 文献レビュー

報告書の形式と意義を決める上で、アプリケーション セキュリティ分野でこれまでに公開された文献を大規模に検証しました。ここで、報告書を作成するにあたり参考にした方々に深く感謝申し上げます。

Open Web Application Security Project (OWASP) は、この報告書の主な情報およびインスピレーションの源となりました。OWASP プロジェクト内にある以下の数多くの貴重な資料から、この報告書の制作に関する非常に明確なアイデアを得ることができました。

- Dr. Dan Geer, Application Security Matters, OWASP AppSecDC 2012 keynote
  http://geer.tinho.net/geer.owasp.4iv12.txt
- OWASP Testing Guide v4
  https://www.owasp.org/index.php/OWASP_Testing_Project
- OWASP Automated Threat Handbook Web Applications
  https://www.owasp.org/index.php/OWASP_Automated_Threats_to_Web_Applications

European Union Agency for Network and Information Security (ENISA) Threat Landscape 2016
https://www.enisa.europa.eu/news/enisa-news/enisa-threat-landscape-2016-report-cyber-threats-becoming-top-priority

Web Application Security Consortium: seminal work on the Web Security Glossary
http://www.webappsec.org/projects/glossary/

Contrast Security’ s State of Application Security: Libraries & Software Composition Analysis
https://www.contrastsecurity.com/state-of-application-security-libraries

Kenna Security: What You Miss When You Rely on CVSS Scores, Michael Roytman
https://blog.kennasecurity.com/2015/02/miss-when-rely-on-cvss-scores/

Advisen and Zurich, 2017 Information Security and Cyber Risk Management Survey
https://www.advisenltd.com/2017/10/25/2017-information-security-cyber-risk-management-survey/

Quantifying the Attack Surface of a Web Application, Keller & Turpe
http://www.feu.de/imperia/md/content/fakultaetfuermathematikundinformatik/pv/97-08/sicherheit2010_heumann-keller-tuerpe_neu.pdf

# 図の目次

# 巻末の注

[1] http://cve.mitre.org/data/refs/refmap/source-EXPLOIT-DB.html
[2] https://www.exploit-db.com/about-exploit-db/
[3] https://www.reuters.com/article/us-equifax-cyber/equifax-says-15-2-million-uk-records-exposed-in-cyber-breach-idUSKBN1CF2JU
[4] https://cve.mitre.org/cgi-bin/cvename.cgi?name=cve-2015-5477
[5] https://f5.com/labs/articles/cisotociso/trends/cisos-striving-toward-proactive-security-strategies
[6] http://carnegieendowment.org/specialprojects/ProtectingFinancialStability
[7] https://www.us-cert.gov/bsi/articles/best-practices/architectural-risk-analysis/architectural-risk-analysis
[8] https://f5.com/labs/articles/threat-intelligence/malware/from-ddos-to-server-ransomware-apache-struts-2-cve-2017-5638-campaign-25922
[9] https://www.exploit-db.com/
[10] https://www.helpnetsecurity.com/2017/07/12/magecart-monetize-stolen-payment-card-info/
https://blog.sucuri.net/2015/06/magento-platform-targeted-by-credit-card-scrapers.html
[11] https://en.wikipedia.org/wiki/Point-of-sale_malware
[12] https://www.owasp.org/images/7/72/OWASP_Top_10-2017_%28en%29.pdf.pdf
[13] https://www.owasp.org/index.php/Top_10_2017-A9-Using_Components_with_Known_Vulnerabilities
[14] https://www.owasp.org/index.php/PHP_Security_Cheat_Sheet
[15] http://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=CVE-2017-5638
[16] https://www.owasp.org/index.php/Top_10_2017-A5-Security_Misconfiguration
[17] https://nvd.nist.gov/vuln/detail/CVE-2017-9805
[18] https://www.owasp.org/index.php/Deserialization_of_untrusted_data
[19] https://www.owasp.org/index.php/Top_10-2017_A8-Insecure_Deserialization
[20] http://www.nytimes.com/2011/06/02/technology/02google.html
[21] https://moxie.org/software/sslstrip/
[22] http://webpolicy.org/2015/08/25/att-hotspots-now-with-advertising-injection/
[23] https://medium.com/@nicklum/my-hotel-wifi-injects-ads-does-yours-6356710fa180
[24] https://www.infoworld.com/article/2925839/net-neutrality/code-injection-new-low-isps.html
[25] https://csrc.nist.gov/Projects/Cryptographic-Module-Validation-Program/Standards
[26] https://www.keylength.com
[27] https://www.reuters.com/article/us-community-health-cybersecurity/u-s-hospital-breach-biggest-yet-to-exploit-heartbleed-bug-expert-idUSKBN0GK0H420140820
[28] https://www.wired.com/2014/11/countdown-to-zero-day-stuxnet/
[29] https://jblevins.org/log/ssh-vulnkey
[30] https://www.usenix.org/system/files/conference/usenixsecurity12/sec12-final228.pdf
[31] https://www.schneier.com/blog/archives/2011/03/comodo_group_is.html
[32] https://www.pcworld.com/article/223760/article.html
[33] https://techcrunch.com/2015/04/01/google-cnnic/
[34] https://sslbl.abuse.ch/
[35] https://arstechnica.com/information-technology/2017/03/google-takes-symantec-to-the-woodshed-for-mis-issuing-30000-https-certs/
[36] https://www.digicert.com/blog/digicert-statement-trustico-certificate-revocation/
[37] https://www.forbes.com/sites/thomasbrewster/2018/04/24/a-160000-ether-theft-just-exploited-a-massive-blind-spot-in-internet-security/#28dce3a45e26
[38] https://thehackernews.com/2018/02/pos-malware-dns.html?m=1
[39] https://krebsonsecurity.com/2018/02/domain-theft-strands-thousands-of-web-sites/
[40] https://www.wired.com/2017/04/hackers-hijacked-banks-entire-online-operation/
[41] https://www.fox-it.com/en/insights/blogs/blog/fox-hit-cyber-attack/
[42] https://krebsonsecurity.com/2015/05/st-louis-federal-reserve-suffers-dns-breach/
[43] http://www.theregister.co.uk/2014/12/17/icann_hacked_admin_access_to_zone_files/
[44] http://www.kiro7.com/news/catholic-hospitals-suffer-second-data-breach-year/81976907
[45] http://www.zdnet.com/article/dutch-dns-server-hack-thousands-of-sites-serve-up-malware/
[46] https://www.scmagazineuk.com/first-true-native-ipv6-ddos-attack-spotted-in-wild/article/747217/
[47] https://www.google.com/intl/en/ipv6/statistics.html
[48] https://f5.com/labs/articles/threat-intelligence/ddos/the-global-playing-field-is-leveling-out-as-europe-and-asia-take-on-more-ddos-attacks
[49] https://tools.ietf.org/html/rfc768
[50] https://en.wikipedia.org/wiki/Smurf_attack
[51] https://www.wired.com/story/github-ddos-memcached/
[52] https://devcentral.f5.com/articles/mitigating-sslsqueeze-and-other-no-crypto-brute-force-ssl-handshake-attacks
[53] https://tools.ietf.org/html/draft-nir-tls-puzzles-00
[54] https://devcentral.f5.com/articles/ssl-renegotiation-dos-attack-ndash-an-irule-countermeasure
[55] https://www.ietf.org/mail-archive/web/tls/current/msg07553.html
[56] https://news.netcraft.com/archives/2017/02/17/hackers-still-exploiting-ebays-stored-xss-vulnerabilities-in-2017.html
[57] http://www.zdnet.com/article/equifax-freeze-your-account-site-is-also-vulnerable-to-hacking/
[58] https://f5.com/labs/articles/threat-intelligence/malware/trickbot-focuses-on-wealth-management-services-from-its-dyre-core?tag=TrickBot

# 巻末の注（続き）

[59] https://f5.com/labs/articles/cisotociso/trends/wait-dont-throw-out-your-firewalls-25982
[60] https://adam.shostack.org/blog/2009/10/are-security-best-practices-unethical/
[61] https://en.wikipedia.org/wiki/Availability_heuristic
[62] https://www.owasp.org/index.php/OWASP_Dependency_Check
[63] https://www.owasp.org/index.php/Cryptographic_Storage_Cheat_Sheet
[64] https://en.wikipedia.org/wiki/Extended_Validation_Certificate
[65] https://www.owasp.org/index.php/HttpOnly
[66] https://f5.com/labs/articles/cisotociso/trends/cisos-striving-toward-proactive-security-strategies
[67] https://f5.com/labs/articles/threat-intelligence/ssl-tls/what-happens-to-encryption-in-a-post-quantum-computing-world
[68] https://caniuse.com/tls1-3
[69] https://blog.pcisecuritystandards.org/are-you-ready-for-30-june-2018-sayin-goodbye-to-ssl-early-tls
[70] https://f5.com/labs/articles?tag=quantum+computing
[71] https://support.f5.com/csp/article/K07359270


APPLICATION THREAT INTELLIGENCE

**F5ネットワークスジャパン合同会社**

**東京本社**
〒107-0052 東京都港区赤坂4-15-1 赤坂ガーデンシティ19階
TEL 03-5114-3210 FAX 03-5114-3201

**西日本支社**
〒530-0012 大阪府大阪市北区芝田 1-1-4 阪急ターミナルビル 16 階
TEL 06-7222-3731 FAX 06-7222-3838

**お問合わせ先：**https://f5.com/jp/fc/



2018 年 9 月