SOPHOS

エクスプロイト解説:

包括的エクスプロイト対策

エクスプロイトとは、正規ソフトウェア製品 (Adobe Flash や Microsoft Office など) の弱点を狙ってコンピュータを感染させ、悪用しようとするものです。企業や組織に侵入しようとするサイバー犯罪者が好んで利用します。そのような犯罪者の目的は、データを盗み出して身代金を要求したり、偵察を行ったり、従来型のマルウェアに感染させる手段として用いたりなどさまざまです。

エクスプロイトはサイバー攻撃で多用される手口であり、報告されているデータ侵害のうち実に 90% 以上で、エクスプロイトが関与していることが分かっています。セキュリティ対策のラインナップにエクスプロイト防止機能を追加すれば、明らかな効果が得られます。

エクスプロイトが最初に登場してから 30 年以上が経過しており、すべての主要なセキュリティベンダーが一定レベルのエクスプロイト防止策を提案できるようにはなりました。ただし、保護対応が可能な範囲とその内容はベンダーごとに大きく異なっています。一部のベンダーは最低限の機能にとどまるのに対し、他のベンダーは主力機能として力を入れています。本ホワイトペーパーでは、エクスプロイトについて、また有名なセキュリティ製品に搭載されているさまざまなレベルのエクスプロイト防止機能について詳しく解説しています。

ソフォス ホワイトペーパー 2018年 3月

目次

エクスプロイト業界: サービスとしてのクライムウェア (Crimeware-as-a-Service)

エクスプロイトキットが登場したことで、マルウェア開発者は苦労してエクスプロイトを作成する必要がなくなりました。Java や Silverlight、Flash のバグを探したり、発見したバグをエクスプロイトに組み込んだり、エクスプロイトをホスティングする無防備な Web サーバーを見つけたり、不正な Web ページにユーザーをおびき寄せる方法を考えたりする手間が不要になります。

同様に、エクスプロイトキットの開発者は、破壊力のあるマルウェアを自ら作成する必要がありません。感染コンピュータを追跡するサーバーの運用や、個々の被害者からの金銭徴収、盗み出したデータの販売などの行為に関わらなくて済みます。

すでに数十億ドル規模の産業となったサイバー犯罪は、2019 年までに約 2 兆ドルの損害を引き起こすと予測されており、攻撃の 1つ 1つの側面が産業化されています。各分野のスペシャリストがセキュリティ脅威の一端を担っており、CaaS (Crimeware-as-a-Service: サービスとしてのクライムウェア) と冗談半分で呼ばれる攻撃代行サービスが成立しています。

業界自体の収益性が高まったため、エクスプロイトブローカーなる者まで現れました。彼らは、エクスプロイトの発見者からエクスプロイトを買い取って、利用したいという人々 (政府機関や悪意のあるハッカーのいずれであろうとも) に販売します。

買い手が購入目的を明らかにすることはありません。Mitnick's Absolute Zero Day Exploit Exchange の創設者であるケビン・ミトニック氏が Wired に説明しているように、「顧客が何らかの理由でゼロデイ脆弱性を必要としている場合、私たちは理由を尋ねませんし、尋ねたとしても彼らは言わないでしょう。リサーチャーが脆弱性を見つけ、私たちに売り込み、私たちからクライアントに売る - リサーチャーやクライアントに理由は問いません。その間に立つことによって利益を生み出しています。」

「ゼロデイの脆弱性を求めるクライアントに対して、その理由を尋ねることはありませんし、尋ねたしても、答えは返ってこないでしょう。リサーチャーが脆弱性を見つけ、私たちに売り込み、私たちからクライアントに売る - リサーチャーやクライアントに理由は問いません。その間に立つことによって利益を生み出しています」

Kevin Mitnick

エクスプロイトを抑止する方法

毎日固有のマルウェアサンプルが 40 万件以上作成され、毎年何千もの新しい脆弱性が発見されている現状では、悪意のある攻撃の防止は容易ではない課題です。サイバー犯罪者から身を守るには、このようなマルウェア亜種の爆発的増加に対してこれまでとは異なる革新的なアプローチが必要です。

サイバー犯罪業界の現状を入念に調査することで、非対称防御の機会が明らかになります。新たな攻撃方法は無限にあるかのように見えますが、実際にはソフトウェアの悪用に使用できる技術は 20数種類程度しかありません。したがって、(個々のエクスプロイトをターゲットにするのではなく、) この約 20数種類の攻撃方法に対抗できるアプローチは非常に強力です。

さらに、脆弱性の種類によっては、攻撃者はいくつかのエクスプロイトを次々に使用しなければ、マルウェアを拡散できないことも少なくありません。そうしたエクスプロイトは年ごとに大きく変化するものではなく、1 つか 2 つの新しい手法が追加される程度です。

主要なセキュリティ製品を評価する際、エクスプロイト手法を抑止する重要な機能が搭載されていないことに驚かされます。また、次世代テクノロジーを搭載しているとする新興ベンダーのなかには、エクスプロイト抑止のための広範なサポートを提供しているベンダーもありますが、そのカバレッジにはムラがあります。

以下は、脆弱性や脆弱性のクラス全体を排除して、サイバー犯罪者や国家が使用するエクスプロイト手法に対抗するための抑止策のリストです。各手法の抑止策は、ベンダーによって異なります。エクスプロイトを防止すると謳っていても、大半のベンダーは一般的に使用されているエクスプロイト手法のごく一部に対して防御しているだけであり、多くの場合、64bit アプリケーションにも対応していないということを知っておく必要があります。真に包括的なエクスプロイト対策を提供しているのはソフォスだけです。

データ実行防止 (DEP：Data Execution Prevention)

データ実行防止 (DEP: Data Execution Prevention) は、バッファオーバーフローの防止を目的として、メモリに対して追加チェックを実行するハードウェアおよびソフトウェアのテクノロジーセットです。DEP がなければ、攻撃者が自身の管理下にあるデータが存在するメモリ上の場所 (ヒープやスタックなど) で悪意のあるコード (シェルコード) にジャンプすることによって、ソフトウェアの脆弱性を悪用しようとする可能性があります。DEP がなければ、通常これらの領域は実行可能とマークされて、悪意のあるコードの実行が可能になります。

DEP は Windows XP 以上のオプトインオプションで、アプリケーションの構築時にソフトウェアベンダーが設定する必要があります。さらに、ビルトインの DEP 保護をバイパスするための攻撃が登場しており、オペレーティングシステム実装に依存することは推奨されません。

必須 ASLR

一部のエクスプロイトは、特定のプロセスに関連していることが分かっているメモリ位置を標的にします。古いバージョンの Windows (Windows XP など) では、システムの起動時にコアプロセスが予測可能なメモリ位置に読み込まれる傾向がありました。アドレス空間配置のランダム化 (ASLR) は、システムファイルやその他のプログラムで使用されるメモリの場所をランダム化します。そのため、攻撃者にとっては特定のプロセスの場所 (実行ファイルの基点、およびスタック、ヒープ、ライブラリの位置など) を正確に推測することが大幅に困難になります。

ASLR は Windows Vista 以上でのみ利用可能で、DEP と同様にアプリケーションの構築時にソフトウェアベンダーが設定する必要があります。また、DEP と同様、ビルトインの ASLR 保護をバイパスする攻撃が登場しており、オペレーティングシステム実装に依存することは推奨されません。

Bottom-up ASLR

ボトムアップのランダム化を有効にすると、ASLR のエントロピーまたはランダム性が向上します。

Sophos Intercept X におけるアドレス空間配置およびボトムアップのランダム化の最大のメリットは、再起動時だけでなく、保護アプリケーションの起動時にも、毎回アプリケーションのベースアドレスが変更されることです。

Null ページ (Null デリファレンス対策)

Windows 8 以降では、プログラムは「NULL ページ」(アドレス空間の仮想アドレス 0x00000000にあるメモリ) を割り当てたりマッピングしたりできないように設計されています。Microsoft はこうすることで、「NULL ポインターデリファレンス」と呼ばれる脆弱性全体の直接的な悪用を防止することに成功しています。

Windows XP、Windows Vista、および Windows 7 では、このような脆弱性を悪用する攻撃者は、カーネルのコンテキストで (ring0 の CPU 権限レベルで) コードを実行できるようになります。その結果、最高レベルへの権限昇格が行われます。このような脆弱性が存在していると、攻撃者はオペレーティングシステムのほぼすべての部分にアクセスできます。

ヒープスプレー プリアロケーション

ヒープスプレーは、実際には脆弱性を突かない手法ですが、脆弱性を悪用しやすくするために用いられる手法です。攻撃者は Heap Feng Shui (ヒープ風水)[1] と呼ばれる手法を用いることで、説く知恵のデータ構造やシェルコードをヒープ上に確実に配置できるため、ソフトウェア上の脆弱性をより確実に攻撃することができます。

一般的なヒープスプレー抑止策では、よく使われるメモリアドレスをあらかじめ確保したり、割り当てておくことにより、不正なペイロードの挿入を防止します。巧妙な攻撃者は、こういったアドレスの存在を把握しているため、実際の攻撃に対してあまり効果はありません。ヒープスプレー プリアロケーションは、Anti-HeapSpray Enforcement または Shellcode Preallocation とも呼ばれており、一般的に、テスト機関で使用される規定のエクスプロイトに対して効果を発揮します。

ダイナミックヒープスプレー

静的なヒープスプレー プリアロケーションに対し、ダイナミックヒープスプレーの抑止対策は、一般にメモリ消費量の急増をきっかけに作動します。

ダイナミックヒープスプレーの抑止対策では、最近のメモリ割り当ての内容が分析され、エクスプロイト攻撃を容易にするために配置された NOP スレッド、ポリモーフィック型 NOP スレッド、JavaScript 配列、およびその他の不審なシーケンスを含むヒープスプレーを示すパターンを検出します。

スタックピボット

アプリケーションのスタックは、メモリのアドレス位置 (「リターンアドレス」) のリストなどを含むメモリ領域です。これらの場所には、プロセッサが近い将来に実行する必要がある実際のコードが含まれています。

スタックピボットは、DEP などの保護機能をバイパスする目的で脆弱性を突くエクスプロイトによって広く使用される手法です。たとえば、Return-Oriented Programming (ROP) 攻撃で ROP ガジェットをチェーン化させます。スタックピボットを利用する攻撃者は、実際のスタックから新しい偽のスタックに固定化することができます。この偽のスタックは、攻撃者の制御下にあるバッファ (ヒープなど) であり、攻撃者はこのスタックからプログラム実行のこの後からのフローを制御できます。

1 https://cansecwest.com/slides/2014/The%20Art%20of%20Leaks%20-%20read%20version%20-%20Yoyo.pdf

スタック実行 (MemProt)

通常の状況では、スタックには近いプロセッサが実行するコードを指し示すデータとアドレスが含まれています。攻撃者はスタックバッファオーバーフロー[2]を利用すると、任意のコードでスタックを上書きできます。このコードをプロセッサ上で実行させるには、DEBを回避するようスタックメモリ領域を実行可能にしなければなりません。スタックメモリが実行可能になると、攻撃者は極めて簡単にプログラムコードを供給し、実行することができます。

スタックベースの ROP 抑止 (Caller)

インターネット接続アプリケーションの脆弱性を突いた制御フローの乗っ取りは、データ実行防止 (DEP) やアドレス空間配置のランダム化 (ASLR) などのセキュリティ技術を無効にする目的でよく使われる手法です。こういったインメモリ攻撃には悪意のあるファイルが利用されていないため、従来のアンチウイルスソフト、ほとんどの「次世代」セキュリティ製品、およびその他のサイバー防御システムでは検出されません。この攻撃は悪意のあるファイルを使用する代わりに、Internet Explorer や Adobe Flash Player などの既存アプリケーションの一部である無害なコードを組み合わせることで、実行時に構築されます。「コード再利用攻撃」や「Return-Oriented Programming (ROP) 攻撃」とも呼ばれます。

通常の制御フローでは、CALL 命令によって VirtualAlloc や CreateProcess などの重要な API 関数が呼び出されます。重要な API が呼び出されると、一般的な ROP 対策は、コードの実行を停止し、スタックの先頭に格納されている「リターンアドレス」から、API を呼び出したアドレスを特定します。API を呼び出したのが CALL 命令でない場合、そのプロセスは停止されます。

スタックは書き込み可能であるため、攻撃者は特定の値をスタックに書き込むことにより、スタックベースの ROP 対策による解析を摺り抜けることができます。スタックベースの ROP 対策では、スタックのデータが無害なものであるか、攻撃者によって操作されたものであるかを識別することはできません。

分岐ベースの ROP 抑止 (ハードウェア拡張 CFI)

上述したとおり、スタックベースの ROP (Return-Oriented Programming) は、抜け穴が多く攻撃側から操作されやすい対策です。この点を改善するには、防御側におけるコード実行時の分析を、すり抜ける隙間が少なく操作されにくいものにする必要があります。

Sophos Intercept X は、メインストリームの Intel® プロセッサ (2008年以降) に搭載されている未使用のハードウェア機能を利用し、ハードウェア拡張 CFI (Control-Flow Integrity: 制御フローの整合性) を確保します。プロセッサのハードウェアには、高度なエクスプロイト攻撃のランタイム検知に活用できる読み取り専用のデータがあります。ハードウェアトレースされた (分岐) レコードを使用することで、ソフトウェアスタックベースのアプローチよりも大きなセキュリティ上のメリットがもたらされます。これらのレコードから取得可能な分岐情報からは、分岐のターゲットだけでなくソースも識別できます。したがって、実際には、制御フローの変化がどこから生じたかが示されます。この情報をスタックベースのソリューション (Microsoft EMET、Palo Alto Networks Traps など) を使用して取得した場合には、同じレベルの信頼性はありません。

2 https://en.wikipedia.org/wiki/Stack_buffer_overflow

ハードウェアトレースされたレコードの分岐情報は操作できません。攻撃者の制御下にあるデータによって、分岐情報が上書きされることはありません。スタックベースのアプローチはスタックデータに依存しています。特に ROP 攻撃の場合、スタックデータは攻撃者の制御下にあり、防御側を欺くことができます。対照的に、Sophos Intercept X で診断されたハードウェアトレースデータは、より信頼性が高く、改ざん防止機能に優れています。

同様のテクノロジーである Endgame 社の HA-CFI (Hardware-Assisted Control-Flow Integrity) は、通常の制御フローをトレーニングすることにより、プログラム作成者の意図に反したコード処理の流れを検知します。ただし、保護対象アプリケーションの全機能とバージョンが反映されている、有効なポインタアドレスのホワイトリストを構築するには、継続的なトレーニングが必要になります。Sophos Intercept X は、トレーニングが不要なうえ、スレッドコンテキストの切り替えやダイナミック周波数スケーリングを実行している最中であっても正常に機能します。

Sophos Intercept X は、Intel® Core™ i3、i5、または i7 プロセッサ (CPU) を検出すると、自動的にハードウェア拡張した制御フローのトレースを実行します。サポート対象のプロセッサが検出されない場合には、Sophos Intercept X はソフトウェアのみのスタックベースの整合性チェックで自動的にフォールバックします。

Sophos Intercept X は、ハードウェアトレースされたレコードを利用することにより、ROP の検出を強化するだけではなく、IAF (Import Address Filtering) も実行し、保護対象アプリケーションのインポート アドレス テーブル (IAT) を保護します。

注：Intel CPU 内の分岐予測に関する脆弱性「Spectre」の修正パッチを適用しても、Sophos Intercept X の機能や性能に影響はありません。

SEHOP (Structured Exception Handler Overwrite Protection)

攻撃者はスタック上の例外レコードのハンドラーポインターを、自身の制御下にある値で上書きすることができます。例外が発生すると、オペレーティングシステムは例外レコードチェーンを確認し、各例外レコードのすべてのハンドラーを呼び出します。攻撃者がレコードの1つを制御しているため、オペレーティングシステムは攻撃者が意図した場所にジャンプします。結果、攻撃者は実行フローを制御できるようになります。

SEHOP は Windows Vista 以上のオプトインオプションで、アプリケーションの構築時にソフトウェアベンダーが設定する必要があります。ビルトインの SEHOP 保護をバイパスする攻撃が登場しており、オペレーティングシステム実装に依存することは推奨されません。

IAF (Import Address Table Access Filtering)

攻撃者が最終的に悪意のある行動を実行するためには、特定のシステム関数 (例: kernel32!VirtualProtect) のアドレスが必要になります。このようなアドレスは、さまざまなソースから取得が可能です。その 1 つが、読み込まれたモジュールのインポートアドレステーブル (IAT) です。IAT は、アプリケーションが別のモジュールの関数を呼び出す際に参照テーブルとして使用されます。コンパイル済みプログラムは利用するライブラリのメモリ位置を知ることができないため、API 呼び出しが実行されるたびに間接的なジャンプが必要になります。ダイナミックリンカーはモジュールを読み込んで結合する際に、対応するライブラリ関数のメモリ位置を示すように実際の関数のアドレスをIATスロットに書き込みます。

Sophos Intercept X は、メインストリームの Intel® プロセッサ (2008年以降) に搭載されている未使用のハードウェア機能を利用することにより、ハードウェア拡張 IAF (Import Address Table Access Filtering) を実行します。CFI (制御フローの整合性) を強制するハードウェアトレースされた (分岐) レコードに加え、ハードウェアの分岐予測も利用して重要なアドレステーブルの保護をさらに強化します。

注：Intel CPU 内の分岐予測に関する脆弱性「Spectre」を修正するパッチを適用した場合でも、Sophos Intercept X の動作に影響はありません。

ライブラリ読み込み

攻撃者は、UNC パス上に悪意のあるライブラリを配置することによって、そのライブラリの読み込みを試みる可能性があります。LoadLibrary API へのすべての呼び出しを監視することで、このタイプのライブラリ読み込みを防止できます。

Reflective DLL Injection (反射型 DLL インジェクション攻撃)

通常、Windows で DLL を読み込む場合には、API 関数 LoadLibrary を呼び出します。LoadLibrary は、DLL のファイルパスを入力として受け取り、メモリに読み込みます。

反射型 DLL ローディングとは、ディスクからではなくメモリから DLL を読み込むことを指します。Windows はこれをサポートする LoadLibrary 関数を持たないため、この機能を利用するには独自に関数を記述する必要があります。独自の関数を記述することのメリットの 1つが、Windows が通常行う処理の一部 (プロセス内の読み込み済みモジュールとして DLL を登録する、など) を省略できる点です。これにより、調査対象となっている反射型ローダーが検出され難くなります。Meterpreter は、リフレクティブローディングを使用して身を隠すツールの代表的なものです。対策として、DLL のリフレクティブローディングがメモリ内で実行されたかどうかを分析します。

シェルコード

シェルコードは、ソフトウェアの脆弱性を悪用するペイロードとして使用されるコードです。感染したマシンを攻撃者が制御するためのコマンドシェルを実行することが多いことから「シェルコード」と呼ばれますが、同様のタスクを実行するコード断片はすべてシェルコードと呼ぶことができます。

多くの場合、エクスプロイトは、脆弱性を悪用してプロセッサの命令ポインタ (EIP/RIP) を制御する前、もしくはそれと同時に標的のプロセスにシェルコードを注入します。この命令ポインタがシェルコードを指すように変更されると、シェルコードが実行され、タスクが行われます。

VBScript God Mode

Windowsでは、ブラウザまたはローカルシェルで VBScript を使用できます。ブラウザで使用する場合、VBScript の機能はセキュリティ上の理由から制限されています。この制限はセーフモードフラグによって制御されます。このフラグが改ざんされると、HTML の VBScript はローカルシェル内であるかのように任意の処理が可能になります。その結果、攻撃者は VBScript に悪意のあるコードを簡単に書き込むことができます。Web ブラウザで VBScript のセーフモードフラグを操作することは、God Mode[3] と呼ばれます[3]。

たとえば、攻撃者は CVE-2014-6332 の脆弱性[4] を悪用することで、セーフモードフラグの値を変更できます (この脆弱性は、Internet Explorer の VBScript エンジンで配列のサイズを変更する際に処理が不適切であったことが原因で発生したものです)。God Mode では、VBScript で記述された任意のコードがブラウザのサンドボックスから脱け出す可能性があります。God Mode が原因で、データ実行防止 (DEP)、アドレス空間配置のランダム化 (ASLR)、および制御フローガード (CFG) の保護は機能しません。

WoW64

Microsoft は、「Windows on Windows」(WoW) レイヤーを通じて 64 ビット版 Windows 上の 32 ビット用ソフトウェア向けに下位互換性を提供しています。WoW が実装されているため、攻撃者は動的分析やバイナリ解凍を複雑化したり、エクスプロイト対策をバイパスしたりすることが可能です。

WoW64 環境での 32 ビットアプリケーションの挙動は、実際の 32 ビットシステムとはさまざまな点で異なります。実行時に実行モードを切り替える機能が搭載されているため、攻撃者は以下のような方法を用いてエクスプロイト、難読化、およびアンチエミュレーションを達成できます。

- 32 ビットコードには含まれていない追加の ROP ガジェット
- 実行モードが混在しているペイロードエンコーダ
- セキュリティ対策の効果を低下させる可能性のある実行環境の機能
- セキュリティソフトウェアによって挿入されたフックのバイパス (32 ビットのユーザー領域においてのみ)

3 https://en.wikipedia.org/wiki/Glossary_of_video_game_terms#God_mode
4 https://www.rapid7.com/db/modules/exploit/windows/browser/ms14_064_ole_code_execution

大半のエンドポイント保護ソフトウェアは、プロセスが WoW64 で実行されている場合に限って、32 ビットのユーザーメモリ空間で重要な API 関数をフックします。攻撃者が 64 ビットモードに切り替えることができれば、32 ビットモードでフックされた重要な API 関数のフックされていない 64 ビットバージョンにアクセスできるようになります。

64 ビット版の Windows の場合、Sophos Intercept X は、WoW64 レイヤーによる 32 ビットモードから 64 ビットモードへの直接切り替えを許可しながらも、プログラムコードが (ROP などの手法を使うなどして) この移行を実行することを禁止します。

WoW64 を悪用する手法の詳細については、「Duo Security: 「WoW64 and So Can You」[5] および「Mitigating Wow64 Exploit Attacks」[6]の調査報告を参照してください。

Syscall

Syscall (またはシステムコール) とは、オペレーティングシステム (OS) のカーネルのサービスを要求するために使用されるプログラム手法のことです。これには、ローカルディスクへのアクセスや、新しいプロセスの作成・実行といったハードウェアに関連するサービスも含まれます。

オペレーティングシステムは通常、正常なプログラムと OS をつなぐ一般的な API (Application Programming Interface) を提供します。通常の状況では、アプリケーションは、常に API を呼び出してカーネルに特定のタスクを要求します。セキュリティ製品は、要求を提供することをカーネルに許可する前に、フックを使用して重要な API 関数の呼び出しを傍受し、ウイルススキャンなどのチェックを実行します。

攻撃者は以下の事実を悪用します。

- すべての API 関数がセキュリティソフトウェアによってフックされるわけではなく、重要な関数のみがフックされる。
- カーネル関数の呼び出しに使用されるスタブは非常に似ており、一意なのは関数インデックスのみ。

攻撃者は、最初の段階で、重要度が低く監視されていない関数のスタブを呼び出し、それを利用して重要なカーネルサービスのアドレスを指定することにより、大半のセキュリティソフトウェアやサンドボックスの解析を効果的に回避できます。

Sophos Intercept X では、保護されていない API 関数を通じて、重要なカーネル関数のアドレスが指定されるのを阻止する新しいアプローチを採用しています。

Syscall を悪用する方法の詳細については、BreakDev.org で「Defeating Antivirus Real-time Protection From The Inside」[7] というタイトルのブログ記事を参照してください。

コード書き換え (プロセスの空洞化)

プロセスの空洞化とは、explorer.exe や svchost.exe などの信頼できるアプリケーションを、悪意のあるコードのコンテナとして機能させる目的のためだけにシステムに読み込む手法です。空洞化されたプロセスは通常、サスペンド状態で作成されます。その後、メモリの割当が解除され、悪意のあるコードに置き換えられます。コードインジェクションと同様、正常なプロセスの中に悪意のあるプロセスが隠されるため、防御対策や検知を回避することが可能です。

5 https://duo.com/blog/wow64-and-so-can-you
6 https://hitmanpro.wordpress.com/2015/11/10/mitigating-wow64-exploit-attacks
7 https://breakdev.org/defeating-antivirus-real-time-protection-from-the-inside

Process Doppelgänging (プロセス ドッペルギャンギング)

Windows コンピュータで使用されるファイルシステムのほとんどは NTFS です。2007年、マイクロソフトは、トランザクション NTFS (Transactional NTFS、TxF) と呼ばれる新機能を公開しました。複数のファイル操作を 1つのトランザクションとして扱い、すべての操作が成功した場合のみトランザクションがコミットされ、一部でも失敗した場合は、一連の処理がロールバックされるといった機能です。この機能では、アプリケーションが、ディスク上の複数のファイルに変更を加え、途中でエラーが検知された場合、すべてのファイルを元の状態に戻す (ロールバック) ことが可能になります。

トランザクション NTFS のもっとも一般的な使用例は、Windows アップデートのインストールです。Process Doppelgänging では、マルウェアを隠すために、トランザクション NTFS のメカニズムが利用されます。正当なファイルを上書きし、信頼できるファイルを装うために、レベルの低い API を通じてマルウェアを実行します (この点は、プロセスの空洞化に類似しています)。マルウェアの実行直前に、すべての変更内容をリジェクトまたはロールバックするため痕跡が残らず、実際に実行されたファイルの内容をウイルス対策ソフトでスキャンすることができなくなってしまいます。ディスク上のファイルを開いても疑わしいコンテンツは含まれていません。有名なアプリケーションのデジタル署名付きファイルであることさえあります。

DLL ハイジャック

DLL ハイジャック、DLL スプーフィング、DLL プリローディング、バイナリプランティングなどと一般的に呼ばれる脆弱性が原因で、多くのプログラムは、当該プログラムによって開かれたデータファイルと同じフォルダに格納されている悪意のある DLL を読み込んで実行してしまいます。

DDE (Dynamic Data Exchange)

Windows の DDE (Dynamic Data Exchange、動的データ交換) は、アプリケーションが相互にプロセス間通信 (IPC) を行えるようにするためのクライアントサーバープロトコルです。攻撃者は、DDE を使用して任意のコマンドを実行します。たとえば、スピア型フィッシング メールや、Web サイトにホストされているコンテンツを通じて、DDEAUTO フィールドを埋め込んだ Microsoft Office ドキュメントを配信し、そのファイルからマクロを使わずに PowerShell のコードを実行することが可能です。また、返信や承諾をすると実行される DDEAUTO コマンドを Microsoft Outlook のメールや会議出席依頼の本文に埋め込むこともできます。

Sophos Intercept X のアプリケーションロックダウン機能は、DDE (動的データ交換) を使用した悪意のあるコードの実行を阻止できます。

アプリケーションロックダウン

攻撃者がメモリおよびコードの悪用に成功し、すべての対策をバイパスした場合、Sophos Intercept X が攻撃能力を制限します。Intercept X のこの機能は、アプリケーションロックダウンと呼ばれ、攻撃者による好ましくないコードの持ち込みを防止します。

アプリケーションのロックダウンによって、通常はアプリケーションのソフトウェアバグに依存していない攻撃を阻止します。こうした攻撃では、たとえば (標的型) フィッシングメールに添付された Office ドキュメントに含まれている不正なマクロが使用されます。

ドキュメントに埋め込まれたマクロは危険度が高いと見なされます。マクロは Visual Basic for Applications (VBA) プログラミング言語で作成されますが、VBA にはバイナリを Web からダウンロードして実行する機能があるだけでなく、PowerShell など信頼されたアプリケーションの実行も可能だからです。

このようなロジックの欠点を突けばコンピュータに侵入できるので、ソフトウェアバグを悪用したり、コード/メモリ保護機能をバイパスする方法を探す必要がなくなります。信頼できるアプリケーションとして広く普及しているソフトウェアの標準機能を悪用するだけで攻撃を仕掛けることができ、ソーシャルエンジニアリングを使ってユーザーに細工したドキュメントを開かせるよう仕向けるだけです。

Sophos Intercept X はフォルダのブラックリストを保持している必要がなく、保護対象アプリケーションの挙動に基づいて、そのアプリケーションを自動的に終了します。たとえば、PowerShell を起動したり、WMI にアクセスしたり、マクロを実行して任意のコードをインストールしたり、重要なシステム領域を操作したりする目的で Office アプリケーションが使用された場合、攻撃による子プロセスが誕生していなかったとしても、Sophos Intercept X がその悪意のあるアクションをブロックします。

Java ロックダウン

ドライブ・バイ・ダウンロード攻撃に利用される主要ツールといえば、これまではエクスプロイトキットでした。エクスプロイトキットを用いて Java Runtime Environment (JRE) に存在する脆弱性を利用し、Windows 実行ファイルのペイロードをドロップするといった具合です。JRE は、一般的なブラウザではプラグインやアドオンとしてロードされます。

Sophos Intercept X は、JRE による Java 以外のアプリケーションの実行を阻止します。たとえば、Java が Windows の PE 形式のバイナリを実行しようとすると、Sophos Intercept X がそのバイナリを強制的に終了します。加えて、Java を悪用して、自動起動の設定項目 (「スタートアップ」フォルダ、Run、RunOnce をはじめとしたレジストリキーなど) を操作することもできなくなります。

注：2014年に公開された Java 8 Update 20 以降、Java アプリケーションのデフォルトのセキュリティ設定レベルは「高」になり、エンドポイントを感染させるのに十分な権限で Java エクスプロイトを実行するのは非常に難しくなっています。その結果、以前ほど Java の脆弱性を狙った攻撃が行われなくなったため、現在 Java ロックダウンの抑止はあまり推奨されていません。

Code Cave

Code Cave とは、正規のソフトウェアを改ざんして、別のアプリケーションを埋め込み、悪用するテクニックを指します。具体的には、正規ソフトウェアのプログラム内の「Code Cave」と呼ばれる普段使わないセクションに別のアプリケーションが埋め込まれます。Code Cave は、大半のアプリケーションに存在し、この部分にコードを埋め込んでも、元のアプリケーションは問題なく動作します。

Code Cave に埋め込まれる実行コードは、単なるリモートシェルの起動コードやバックドアであることが多く、攻撃者は、この短いコードを足掛かりにエンドポイントで他の操作を実行します。このタイプの攻撃を行うには、エンドポイントに不正侵入してバックドアを持つアプリケーションをインストールするか、Code Cave に悪意のあるコードが埋め込まれているアプリケーションをユーザーがインストールするように仕向ける必要があります。

Code Cave が悪用される大きな理由の 1つは、ユーザーや管理者の目から逃れられやすいという点です。コードが埋め込まれてもアプリケーションは問題なく動作するので、埋め込まれた別のアプリケーションが実行されていることに気づきにくいのです。

たとえば、改ざんされたアプリケーションが、通常デバイスにインストールされている標準的な業務ツールであった場合、従来型のウイルス対策製品で問題が検出されたとしても、その業務ツールがマルウェアであるとは考え難いでしょう。管理者は、ウイルス対策製品の誤検出と見なして、問題のアプリケーションを除外リストに追加してしまうかもしれません。このようにして、エンドポイントにマルウェアを潜伏させるだけでなく、埋め込んだアプリケーションの実行を管理者が許可するように仕向けることさえもできてしまいます。

いわゆる「サプライチェーン攻撃」では、ソフトウェア製品のアップデートサーバーに不正侵入し、アップデートプログラムに悪意のあるコードを密かに埋め込むことが可能です。そのプログラムを何も知らない製品ユーザーが実行すると、ランサムウェアやワイパー型マルウェアなどに感染してしまいます。

Sophos Intercept X は、バックドアが埋め込まれたアプリケーションの実行を自動的にブロックします。Code Cave や、感染した実行形式ファイルに挿入されたセクションに辿り着かず、コードが実行されない場合でも、埋め込まれているシェルコードを検知することが可能です。Shellter や Backdoor Factory などのシェルコード インジェクション ツールに対して、幅広い防御を提供するのも大きな特長です。

プロセスマイグレーション – リモートから実行される Reflective DLL Injection

プロセスマイグレーションは、デバイスへ最初に常駐化しようとするときのほか、権限昇格を実行したり、アクセスを継続する目的で、別のプロセスへ移動するためによく使われるテクニックです。ブラウザを閉じる、感染プロセスを終了するといったエンドユーザーの操作に関わることなく、コントロールを維持したがります。攻撃者はシステムに移動できるのがもっとも望ましいです。

リモートからの Reflective DLL Injection も類似の攻撃です。既に感染させているプロセスから、別のプロセスを操作して DLL を読み込み、任意のコードを実行するため、より対応が難しくなります。

ローカル権限昇格

Sophos Intercept X は、高い権限を持つプロセスから盗み出したトークンを利用することにより、低い権限のプロセスがより高い権限へ昇格するのを防止します。この手法は、別の脆弱性を組み合わせて、システム権限を使って悪意のあるコードを送り込み、実行する手段としてよく使われます。

DoublePulsar コードインジェクション

DoublePulsar は元々、アメリカ国家安全保障局 (NSA: National Security Agency) が開発し、2017年初頭にハッカー集団「Shadow Brokers」が盗み出したとされるバックドアツールです。まったく新しいインジェクションのテクニックを備えており、その部分は、「EternalBlue」や「EternalRomance」など、NSA が制作した複数のエクスプロイトと共通します。これらのエクスプロイトは、世界的に大流行した「WannaCry」や「NotPetya」が自己拡散に使ったワーム機能にも利用されました。

DoublePulsar コードインジェクションでは、非同期プロシージャコール (APC：Asynchronous Procedure Call) を悪用して、信頼できる正規のプロセス内で任意のコード (シェルコード) が実行されます。Sophos Intercept X は、DoublePulsar が利用する基本テクニックを本質的に阻止するため、同じテクニックを使用するコードインジェクション攻撃もブロックすることが可能です。

AtomBombing コードインジェクション

非同期プロシージャ コール (APC) のコードインジェクションでは、プロセススレッドの APC キューに悪意のあるコードが埋め込まれます。このスレッドが可変状態になると、キューに追加された APC 関数が実行されます。AtomBombing は、グローバル アトム テーブルに悪意のあるコードを書き込み、非同期プロシージャ コールを利用してそのコードを実行させる、コードインジェクション手法の 1つです。

DoubleAgent コードインジェクション

DoubleAgent は、「Microsoft Application Verifier」という正規の Windows ツールを悪用する攻撃です。このツールは、全バージョンの Windows に搭載されており、アプリケーションのバグを検出し、修正するための実行時検証ツールとして使われます。Application Verifier を使用することにより、ディスクから任意のライブラリを読み込みこむことができるため、悪質なライブラリを標的のプロセスに読み込ませ、そのプロセスの権限を奪うことが可能になります。

DoubleAgent は、ウイルス対策製品の脆弱性を突いたゼロデイ攻撃ですが、Application Verifier の本来の意図は、信頼できる製品や Windows のプロセスなど、開発者が選択したアプリケーションに任意のコードをロードさせるためのものです。

Sophos Intercept X は、Application Verifier の悪用によるコードインジェクションを阻止します。

Intercept X の機能

機能	
エクスプロイト防止	
データ実行防止 (DEP：Data Execution Prevention)	✓
必須 ASLR	✓
Bottom-up ASLR	✓
Null ページ (Null デリファレンス対策)	✓
ヒープスプレーアロケーション	✓
ダイナミックヒープスプレー	✓
スタックピボット	✓
スタック実行 (MemProt)	✓
スタックベースの ROP 抑止 (Caller)	✓
分岐ベースの ROP 抑止 (ハードウェア拡張)	✓
SEHOP (Structured Exception Handler Overwrite)	✓
IAF (Import Address Table Filtering)	✓
ライブラリ読み込み	✓
Reflective DLL Injection (反射型 DLL インジェクション攻撃)	✓
シェルコード	✓
VBScript God Mode	✓
Wow64	✓
Syscall	✓
コード書き換え	✓
DLL ハイジャック	✓
Squiblydoo Applocker Bypass	✓
APC プロテクション (Double Pulsar / AtomBombing)	✓
プロセスの権限昇格	✓
敵対行為に対するアクティブな抑止	
認証情報盗難防止	✓
Code Cave 抑止	✓
MITB 攻撃対策 (セーフブラウジング)	✓
Malicious Traffic Detection (MTD)	✓
Meterpreter Shell Detection (Meterpreter シェル検出)	✓

機能	
ランサムウェア対策	
ランサムウェアからのファイル保護 (CryptoGuard)	✓
ファイルの自動修復 (CryptoGuard)	✓
ディスクとブートレコードの保護 (WipeGuard)	✓
アプリケーションロックダウン	
Web ブラウジング (HTA を含む)	✓
Web ブラウザのプラグイン	✓
Java	✓
メディアアプリケーション	✓
Office アプリケーション	✓
ディープラーニング	
ディープラーニングによるマルウェア検出	✓
ディープラーニングによる業務外アプリケーションのブロック	✓
誤検知削減	✓
Live Protection	✓
レスポンス・調査・クリーンアップ	
根本原因解析 (RCA)	✓
Sophos Clean	✓
Synchronized Security Heartbeat	✓
導入形態	
単体エージェントとして実行	✓
既存の他社製ウイルス対策製品との共存	✓
既存の Sophos Endpoint のエージェントのコンポーネントとして実行	✓
Windows 7	✓
Windows 8	✓
Windows 8.1	✓
Windows 10	✓
macOS*	✓

* CryptoGuard、MTD (Malicious Traffic Detection)、Synchronized Security Heartbeat、根本原因解析が利用できます

Sophos Intercept X
無償評価 (30日間)

www.sophos.com/intercept-x

ソフォス株式会社営業部
Email: sales@sophos.co.jp

18-03-06 WP-JA (DD)

SOPHOS