

IDS での攻撃の識別方法

イントロダクション

A. ネットワークの侵入について、多くのベンダーが Intrusion Detection System (IDS; 侵入検知システム) を開発していますが、そこにはさまざまな要素が関連しています。そのため、最良の方法や解決策をめぐって、各 IDS ベンダーの主張も大きく分かれています。ここでは、各種侵入検知方法の長所および短所について検討し、IDS 製品に関するシスコのアプローチを説明します。ここで扱う検知方法は、シンプル パターン照合、ステートフル パターン照合、プロトコル デコードベース シグニチャ、学習ベース シグニチャ、異常検出です。ここではそれぞれの分析手法について詳細に検討することはできないため、基本概念と各アプローチの違いに重点を置いて説明します。

ここでは、「シグニチャ」という用語は、侵入イベントのタイプを示す一連の条件という意味で使用します。シグニチャで使用されるアルゴリズムは、ここで説明する 5 種類の方法のいずれか（「異常検出シグニチャ」など）に基づいています。シグニチャという用語は、とりわけパターン照合に密接に関連しているため、この定義を明確に理解しておく必要があります。シグニチャベースの IDS はパターン照合にしか使用されないと誤解されることがありますが、この定義を理解しておけば、こうした誤解がなくなります。

パターン照合

パターン照合では基本的に、単一パケット内で、固定シーケンスのバイトを検出します。その名の示すとおり、柔軟性に欠けることはありますが、展開の容易な方法です。多くの場合、パターンの照合が行われるのは、比較対象となるパケットが特定のサービスに関連付けられている（つまり、特定のポートで送受信される）場合のみです。そのため、各パケットに対して行われる検査の量は軽減されます。ただし、特定のポートを使用しないプロトコル、特にトロイの木馬およびそれに関連するトラフィックは、通常任意にポートを変更することが可能で、処理が難しくなる傾向があります。

シンプル パターン照合におけるシグニチャの構造は、次のようになります。

パケットが IPv4 の TCP で、宛先ポートが 2222、ペイロードに文字列「foo」が含まれる場合アラームが発行される、とします。

この例のパターン照合はもちろん非常に単純なものですが、ここから派生する照合方法も単純なものです。パケット内で検査する開始点および終了点を規定したり、検査対象のパケットの TCP フラグを指定したりすることもできます。ただし、この方法は侵入検知の最も単純で原始的な基本要素です。



長所

- 侵入を検知するための最も単純な方法。
- これまでの経験に基づいて直接関連付けを行うことができる。
- 指定したパターンについて、確実に警報が発行される。
- プロトコルに関係なく利用できる。

短所

- シグニチャ作成者が固有のパターンを規定できない場合、誤検知（フォールス ポジティブ）の発生率が高くなる。
- 攻撃に修正が加えられると、イベントの不検知（フォールス ネガティブ）が発生することがある。
- 攻撃に利用される単一の脆弱性に対して複数のシグニチャが必要となることがある。複数のツールを利用する場合、複数のシグニチャが必要となる。
- 一般的に、単一パケットの検査しか行えないため、HTTP トラフィックなどのネットワーク トラフィックについて、ストリームベースの特性が十分検査できない。そのため、検査をくぐり抜ける侵入方法を簡単に編み出されてしまう。

ステートフル パターン照合

より高度な方法は、ステートフル パターン照合ベースの分析です。このシグニチャ開発方法では、「ネットワーク ストリームは単一パケットだけで構成されているわけではないので、ストリームのステート内のコンテキストに合わせて照合を行う必要がある」という概念が追加されています。そのため、このようなシグニチャ分析を行うシステムでは、TCP ストリーム内のパケットの到着順序を考慮し、パケットの境界を越えてパターン照合を行う必要があります。

このシナリオを、シンプル パターン照合で紹介した例に当てはめるとどうなるでしょうか。この場合、パケットごとにパターンを探すのではなく、まず監視対象の TCP ストリームでステート情報を維持する必要があります。この違いを理解するため、次のシナリオを考えてみてください。検出対象の攻撃がサーバに接続されたクライアントから実行され、パターン照合メソッドが IDS に設定されているとします。攻撃により、ポート 2222 を宛先とする特定の単一 TCP パケットに文字列「foo」が設定されている場合、アラームが発行されます。しかし、攻撃者が文字列の設定方法を変更し、最初のパケットで「f」がサーバに送信され、次のパケットで「o」が送信されるようにした場合、アラームは発行されません。これに対して、ステートフルパターン照合アルゴリズムを使用した場合、センサに文字列の「fo」の部分がまず格納され、「.o」の部分がクライアントから転送された時点で照合を完了することができます。

長所

- 展開に要する労力は、シンプルパターン照合の場合と大差がない。
- これまでの経験に基づいて直接関連付けを行うことができる。
- 指定したパターンについて、確実に警報が発行される。
- プロトコルに関係なく利用できる。
- シンプルパターン照合に比べて、検査のくぐり抜けを難しくできる。



短所

- シグニチャ作成者が固有のパターンを規定できない場合、フォールス ポジティブの発生率が高くなる。
- 攻撃に修正が加えられると、イベントの不検知（フォールス ネガティブ）が発生することがある。
- 攻撃に利用される単一の脆弱性に対して複数のシグニチャが必要となることがある。複数のツールを利用する場合、複数のシグニチャが必要となる。

プロトコル デコードベースの分析

プロトコル デコードベース シグニチャは、ステートフル パターン照合をさまざまな点で高度に拡張したものです。このクラスのシグニチャは、クライアントやサーバでのやりとりと同じ方法で各種エレメントをデコードして実装されます。プロトコルのエレメントを識別する際、IDS では、RFC によって定義された規則を適用して違反を検出します。こうした違反は、特定のプロトコル フィールド内でのパターン照合によって検出される場合もあれば、フィールドの長さや引数の多さなどの変数を利用した高度な手法が必要となる場合もあります。誤解されがちですが、パターン照合とプロトコル デコーディングは、相互に排他的なものではありません。

理解を深めるため、もう一度 abc 攻撃の例を考えてみましょう。攻撃の行われる基本プロトコルを架空の BGS プロトコルとし、攻撃では不正な引数「foo」が BGS Type フィールドに設定されるとします。さらに複雑にするため、Type フィールドの前には BGS Option という可変長のフィールドがあるとします。オプションの有効な一覧は、fooh、mooh、tormer、buildo とします。この場合にシンプル パターン照合またはステートフル パターン照合アルゴリズムを使用すると、オプション「fooh」に検索対象のパターンが含まれているため、フォールス ポジティブが発生します。また、フィールドが可変長であるため、検索の開始位置と終了位置の指定によってフォールス ポジティブを制限することはできません。BGS Type 引数として「foo」が設定されていることを確認するには、プロトコルを完全にデコードするしかありません。

プロトコルを完全にデコードしていない場合、パターン照合アルゴリズムでの対処が難しい動作をプロトコルで許可すると、フォールス ネガティブが発生することがあります。たとえば、BGS ヘッダーへの値の設定時に、BGS プロトコルで1バイトおきにNULLの設定を許可している場合、パターン照合では、fx00ox00ox00 は検出されません。プロトコル デコード対応の分析エンジンでは、NULL が無視されるため、Type フィールドに「foo」が設定されているとみなされ、アラームが発行されます。

長所

- プロトコルの定義が適切で、十分機能している場合、フォールス ポジティブが発生する可能性が低い。
- これまでの経験に基づいて直接関連付けを行うことができる。
- 基準を緩めることにより、テーマに基づく変種の検出に対応することができる。
- 定義に従って、プロトコル規則の違反に対して確実に警報が発行される。

短所

- RFC があいまいで、開発者が自由に解釈および実装できる余地がある場合、フォールス ポジティブの発生率が高くなる。このグレーゾーンに属するプロトコル違反は多数存在する。
- プロトコル パーサーを適切に実装するには、開発に時間がかかる。



学習ベースの分析

学習ベース シグニチャでは、アラーム決定の基礎となるアルゴリズム ロジックを使用します。このアルゴリズムには、送信されるトラフィック タイプの統計評価がよく使用されます。このタイプのシグニチャの例としては、ポートのスweepの検出に使用されるシグニチャがあります。このシグニチャは、特定のマシンに接続される固有ポートのスレッシュホールド値があるかどうかを探します。対象となるパケットのタイプ (SYN パケットなど) を指定することにより、このシグニチャをさらに絞り込むことができます。また、プローブがすべて単一の送信元で生成されている必要があるという要件を設けることもできます。このタイプのシグニチャの場合、監視するネットワークの利用パターンに合わせるため、スレッシュホールドを一部変更する必要があります。このタイプのシグニチャを使用すると、単純な統計例だけでなく、非常に複雑な関係も検出することができます。

長所

- 他の手段では検出できない不正アクティビティを検出できる。

短所

- ネットワーク トラフィックに合わせ、フォールス ポジティブの発生を抑えるには、アルゴリズムのチューニングおよび変更が必要な場合もある。

異常ベースの分析

異常ベース シグニチャは基本的に、「正常」とみなされるものから逸脱したネットワーク トラフィックを検出します。この方法の最大の問題は、「正常」とするものを最初に定義しなければならないことです。システムによっては、「正常」の定義がハードコーディングされているものもありますが、この場合、学習ベースのシステムと考えることができます。「正常」を学習するよう構築されているシステムもありますが、こうしたシステムでの課題は、「異常」動作を誤って「正常」として分類してしまう可能性を低減することです。また、学習するトラフィック パターンを「正常」とみなすには、許容可能な逸脱と、許容できない逸脱や攻撃ベースのトラフィックを示す逸脱とをシステムでどう区別するかが課題となります。異常ベースの検出メソッドの使用を謳った製品も一部市販されていますが、この領域の開発は主に学術的な世界を中心に進められてきました。このタイプの検出のサブカテゴリとして、プロファイルベースの検出メソッドがあります。このようなシステムでは、ネットワーク上でのユーザやシステムのやりとりの変化をもとに警報が発行されます。ここでも、メイン カテゴリで動作の変更の意味を推測する上で障害となる制限や問題が同様に存在します。

データを調べることにより、興味深い事実を明らかにできます。それに基づいて、このアルゴリズムを使用し、実行中の攻撃を検出することができます。ただし、システムによって提供される情報は、一般的に特異性が非常に低く、コンテキストを適切に判断するには詳細な調査が必要になります。

異常ベースの検出には、プロトコルベース異常検出という検出方法もあります。この方法は、前述のプロトコル デコード メソッドと密接に関連しています。プロトコル定義でプロトコル検出が適切に定義されているため、異常の学習は不要です。プロトコル異常の一例は、プロトコル内のフィールドに予期しない値が設定されている場合です。多くのプロトコル デコード分析エンジンでは、既知の攻撃に直接関連するものではないが、「異常」(長さに基づくバッファのオーバーフロー検出など) とみなされるプロトコル異常があった場合に警報が発行されるため、どの方法で検出が行われたかはっきり分類できない場合もあります。そのため、この例では、エンジンは異常ベース システムの属性を持ちます。

統計異常についても、トラフィックの種類ごとに定められた統計基準を学習または通知することにより、ネットワーク上で特定することができます。たとえば、UDP フラッド、TCP フラッド、ICMP フラッドなどのトラフィック フラッドの検出が可能です。これらのアルゴリズムでは、現在のトラフィック受信レートと履歴参照値が比較され、それに基づいて、履歴の平均から統計的に大きく逸脱したものが警報されます。警報の統計スレッショールドをユーザが設定できるものもあります。

こうしたシステムは、一般的に IDS のソリューションとして製品化されていますが、同様の手法を利用したシステムの学術的な研究も盛んに行われています。ある程度の成果は出ていますが、製品として実用化に耐えるものとはなっていません。学術的研究分野における成果の少なさは、「夢のような話は所詮夢でしかない」という言葉を裏付けるものとなっています。

長所

- このメソッドが適切に実装できれば、未知の攻撃を検出することが可能になる。
- シグニチャを新しく開発する必要がないため、オーバーヘッドが低減できる。

短所

- 一般的に、侵入データを正確に把握することができない。何か問題が起こったらしいということはわかっても、はっきりしたことはわからない。
- ほとんどの場合、信号対雑音比が非常に低い。
- システムが何を「正常」とみなすかという環境によって結果が大きく異なる。

結論

では、最良の方法はどれでしょうか。答えは、実現しようとしている機能によって異なるとしか言えません。Network IDS (NIDS) に対するシスコの方針は、パターン照合、ステートフル パターン照合、プロトコル デコード、学習ベース シグニチャを組み合わせることで使用することです。解決しようとしている問題、つまり、検出しようとしている攻撃に合わせて、適切なツールを使用することをお勧めします。組み合わせの比重としては、プロトコル デコードの占める割合が大きくなります。シスコのシグニチャの大部分がこの方法で実装されているためです。その次によく使用されるのは学習ベース シグニチャ、次がパターン照合となり、最後が、プロトコルおよび統計の各種異常ベース メソッドとなります。シスコは、今後も IDS 分野の研究と開発の監視を継続し、新しい手法の効率性、実践性、コスト効率、商業上の実現可能性が実証された時点で、製品に組み込んでいきます。

©2004 Cisco Systems, Inc. All rights reserved.

Cisco、Cisco Systems、および Cisco ロゴは米国およびその他の国における Cisco Systems, Inc. の商標または登録商標です。この文書で説明した商品、サービスはすべて、それぞれの所有者の商標、サービスマーク、登録商標、登録サービスマークです。この資料に記載された仕様は予告なく変更する場合があります。



シスコシステムズ株式会社

URL: <http://www.cisco.com/jp/>

問合せ URL: <http://www.cisco.com/jp/service/contactcenter/>

〒 107-0052 東京都港区赤坂 2-14-27 国際新赤坂ビル東館

TEL: 03-6655-4433

電話でのお問合せは、以下の時間帯で受付けております。

平日 10:00 ~ 12:00 および 13:00 ~ 17:00

お問合せ先