2007－DSM－47 (8)
2007／9／21

IDSの誤検知除去に対するソフトウェア工学的アプローチ

淡路 淳
千葉大学自然科学研究科

今泉 貴史
千葉大学総合メディア基盤センター

本研究では侵入検知システム (IDS:Intrusion Detection System) の誤検知の判断基準を明確にし、さらに誤検知を除去するために、ソフトウェア工学的なアプローチを試みる。誤検知の判断は管理者が行うものであり、明確な基準はない。検知手法によって誤検知の定義を変えてしまえば、IDS の性能を示す際に用いられる誤検知の数に統一性はなく、評価や比較を行うことは困難である。そこで本研究では、検知したいものを検知手法によらず表現する方法として UML を用いた記述方法を提案する。この記述と照らし合わせて誤検知の判断をすることで誤検知の定義が明確になり、評価や比較が行えるようになる。

Software Engineering Approach to False Detection Removal of IDS

Jun AWAJI
Graduate School of Science and Technology,
Chiba University

Takashi IMAIZUMI
Institute of Media and Information Technology,
Chiba University

We applied some techniques used in the software engineering area to clarify the criterion, and to remove the false detection of IDS. When an alert occurs, system administrators decide whether the alert is a false detection. In the judgment of the false detection, there are no clear criteria. The definition of the false detection changes depending on detection techniques. Therefore, it is difficult to evaluate and compare the detection techniques. In this paper, we propose a description method of using UML. By using this technique, we can describe the one wanting to detect it without depending on the detection techniques. We can judge the false detection by comparing events with this description. And, we can evaluate and compare the detection techniques.

1 はじめに

インターネットの普及にともない、現在では多くの計算機がネットワークに接続されている。しかし、セキュリティホールを突くための攻撃コードが多数インターネット上に公開されており、ウイルスなどの不正なプログラムも増加している。高度なスキルを持っていなくても攻撃を行うことができる現状では、計算機がネットワークに接続されている以上、いつ攻撃にさらされてもおかしくはない。そのため、不正侵入を発見し被害を食い止めるためのシステムである侵入検知システム (IDS:Intrusion Detection System) が普及してきている。

侵入検知を行うための手法は様々なものが提案されている。IDS には誤検知が多いという問題点があるため、IDS として提案された手法の性能を論じる時には誤検知の数や割合を示すことになる。しかし、何を検知すべきなのか、何を検知すべきではないのか、誤検知とは何を指すのか、といったことを明確に定めぬままに誤検知の数や誤検知率が論じられることが多い。

誤検知の定義として、検知したいものをそのシステムの管理者に記述させたり、学習の期間から得られた情報を用いる手法もある。しかし、検知手法に基づく誤検知の定義では手法ごとに定義が異なってしまい、統一的な評価や比較を行うことは困難である。また、表現できることが検知手法によって制限されることになり、管理者が検知すべきと考えていることを正確に表現できないこともある。また具体的な攻撃をすべて記述させるような定義の方法は、現実的には不可能である。詳細な記述を管理者に求める定義の方法では、管理者の負担が大きくなり、人為的なミスが含まれる可能性も高くなる。このようなミスによって生じた余計な検知や検知漏れを誤検知とするかどうかの判断も誤検知の定義における難しい問題である。

本研究では、誤検知を判断するためにソフトウェア工学的なアプローチをとる。ソフトウェア工学において、設計や実装の正当性を確かめるためには要求や仕様と比較を行う。そこで本研究では、誤検知を定義するため、実装方法ではなくシステムが何をすべきかを記述した仕様、つまりは検知手法によらず管理者が何を検知したいかを記述した要求仕様を用いる。この要求仕様と実際に検知したものを照らし合わせることで誤検知を判断する。この要求仕様を記述するにあたって、次の要件を満たす記述法を提案する。

(1) 検知手法によらない

(2) 管理者の興味を抽象的に表現できる

(3) 人為的な記述ミスが含まれる可能性が少ない

この記述と照らし合わせて管理者が誤検知の判断をすることで誤検知が明確になり、評価や比較が統一的に行えるようになる。

2 誤検知の定義とあいまいさ

IDS がアラートを発した時に管理者はそのアラートが不正侵入かどうかを判断する必要がある。不正侵入と判断したならば、詳細な調査を行い対応することになる。不正侵入ではないと判断したならば、同様の誤検知が生じないように可能であれば設定などの変更を行うことになる。また、IDS がアラートを発しないにもかかわらず不正侵入があった場合も、同様の誤検知が生じないように設定の変更が必要である。

誤検知には攻撃ではないのにもかかわらずアラートを発する false positive と、攻撃を見逃してしまう false negative がある。そのどちらの誤検知についても、誤検知かどうかを判断するのは、管理者の主観によるところが大きい。攻撃の予備動作とされるポートスキャンやパッチを適用済みであり実際には影響のない攻撃などは、ネットワークの管理者によって検知すべきかどうかの判断は分かれるだろう。このように誤検知の判断基準が管理者の興味というあいまいなものであるにもかかわらず、IDS の性能を評価する際の誤検知数は、判断基準を明確にせず論じられることが多い。

誤検知数で性能評価を行うならば、誤検知の判断基準は明示されるべきであるが、管理者の主観をすべて具体的かつ詳細に記述するというのは実際問題として不可能である。そもそもここで必要な判断基準とは、何を評価したいかによって異なる。例えば、ワームを検知する能力を評価したければ、ワームを検知するということ明示すれよい。ある特徴の動作をするワームを検知する能力を評価したければ、その動作の特徴を明示すればよい。全てのワームを具体的に記述することは不可能であるため、ワームとは何かを検証が容易な形で完全に記述することは困難である。全てのパケットとマシンの状態を詳細に調べて、管理者がワームかどうかを判断できるだけの定義が記述されればよい。むしろ判断基準として明示すべきなのは、予備動作を検知したいか、影響がなくても検知したいか、といった興味の対象がどこまでなのかである。つまり、管理者の興味の対象の明確化が判断基準となる。

3 ソフトウェア工学的アプローチ

ソフトウェア工学は、高度で安全なソフトウェアを短期間で設計するための学問である。システムを開発しようと思いついた時点から、実際に動くソフトウェアが完成し運用されるまでをいくつかのフェーズに分ける。そして、フェーズごとに典型的な課題があり、それを解決するための方法を明確にしようというものである。

開発対象のシステムを思いつき、あいまいな顧客の要求を実際のシステムにするために具体的な設計にしていくという過程は、IDS における管理者の主観というあいまいな誤検知の定義を、実際の IDS が検知するものとして具体的に定めていく過程に利用できる。

また、システムが正しいかどうかは、システム単体では判断できない。実装方法ではなくシステムが何をすべきかを記述した仕様を与え、対象システムが仕様に照らして正しいかどうかを判断するという考え方は、IDS が発したアラートが正しいかどうかを判断する際に利用できる。

このように、あいまいさを含む誤検知の定義において、ソフトウェア工学の考え方や手法を利用することは問題解決に適していると考えられる。本研究では誤検知の定義を管理者の判断と捉え、管理者の興味を表現することで誤検知の定義すなわち要求仕様とする。また、要求仕様は、システムを作る際に要求、分析、設計に用いられる UML[1] を用いて表現する。あいまいな要求から具体的な設計にしていく過程で用いられるモデルは、あいまいな管理者の興味の対象を明確に表現するのに適していると考える。

4 要求仕様

4.1 要求仕様を記述する目的

誤検知の定義は管理者の主観によることは 2 章で述べた。そして、誤検知かどうかを議論するならば、管理者が検知して欲しいものを表した要求仕様と、実際に検知したものを比べる必要があることを 3 章で述べた。

管理者の興味を柔軟に表現した要求仕様を記述する目的は、明示されなかったり手法ごとに異なっていた誤検知の定義を統一的に表現し、誤検知を明確にすることである。誤検知の定義を定めなければ、誤検知を数えることはできない。検知手法ごとに異なる誤検知の定義をしていたのでは、誤検知の数は同様の手法に対する相対的な比較しかできず、性能の評価とはいい難い。しかし、統一的な誤検知の定義の表現方法を定めれば、誤検知の数という指標で性能を評価することが可能になる。

要求仕様を記述することの目的はもう一つある。それは、管理者の主観が誤検知の定義であることを明確にすることである。正しいアラートかどうかを判断するためにはアラートを出す根拠とは別に、どんなアラー

トが欲しいかを示す記述が必要である。この2つを分けずに一緒にしてしまうと正しいかどうかの判断ができない。例えば、手法を基に誤検知の定義を定めてしまうと、管理者が必要ないと感じたアラートがあったとしてもシステムとして正しいアラートであればfalse positiveではないとされ、修正の必要がないことになってしまう。これでは管理者はアラートに対してどんな対応をするべきなのか分からなくなってしまう。そうならないためにも、誤検知の定義を管理者の主観であると定め、その表現方法として管理者の興味の記述をすることは必要である。

4.2 要求仕様の概要

要求仕様はUMLを用いて表現する。UMLは分析、設計、実装などに用いられる、システムのモデルを表現するための言語であり、わかりやすい表記法が定義されているため、理解が容易である。用いる図のモデル要素だけでは表現できない情報は、UMLの共通の要素であるノートを用いて、非形式的に記述を追加できる。

要求仕様にはIDSの性能を誤検知の数で評価するという目的を踏まえ、以下の3つの情報を含む必要がある。

- 問題領域
- 検知対象
- 誤検知単位

問題領域　問題領域とは管理者が期待する評価対象のIDSが検知すべきものの範囲である。問題領域によって検知対象や検知単位の表現方法は異なる。攻撃全般を検知するIDSとして評価したいのか、特定の攻撃の検知手法として評価するため特定の攻撃だけを検知するべきなのかを明確にすることで、管理者の興味の範囲を明確にする。問題領域の区分けとしては、IDS、攻撃の種類、攻撃の手法の3つとし、いずれかを指定する。

検知対象　検知対象とは管理者が期待する評価対象のIDSが検知すべきものを表現したものである。問題領域によって表現方法が異なる場合がある。どんな種類の攻撃を検知したいか、何を攻撃とみなすか、また攻撃の予備動作は検知対象に含まれるか、などの情報である。

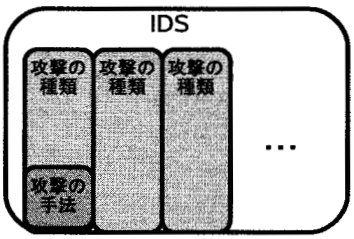

図 1: 問題領域

誤検知単位　手法により攻撃を検知する単位が異なることがある。そのため発生するアラートの数や、誤検知の数が大きく異なる場合がある。誤検知の数を評価するのであれば、誤検知の数え方を明示する必要がある。

4.3 問題領域がIDSの要求仕様

問題領域がIDSの要求仕様は、UMLのユースケース図をもとに表1の対応表に基づいて管理者の興味の対象を表現したものと、ユースケース記述をもとに表2の対応表に基づいて管理者の詳細な興味を記述したもので表現する。ユースケース図とはUMLにおいて、ユーザー視点でシステムの機能を把握するためのダイアグラムである。ユースケースとはシステムが提供する機能のことであり、ユースケース記述とはユースケースを詳細に表した記述のことである。

ユースケース図におけるユースケースをそのままIDSの機能と考え、アクターは最終的なアラートを受けとる人間つまり管理者と考える。関連はアラートの流れとなり、他のシステムか管理者とつながる。これにより、システムがどんな構成になっているか、管理者がIDSに対して何を検知して欲しいと要求しているかが一目で理解できる。またネットワークに最適化するためなどの管理者の興味を示す個別具体的な条件はシナリオとして記述する。ユースケース図におけるシナリオとは、あるユースケースの具体例のことである。

管理者の興味の対象は、管理者と直接関連のある機能のユースケース記述となる。そのため必ず記述が必要になるのは管理者と直接関連のあるユースケース記述のみである。対象を検知する機能には、ユースケース記述にその対象の定義を含む。対象を検知する機能からアラートを受け取るその他の機能には、ユースケース記述にアラートを出した機能の検知対象の定義を記述する必要がある。

対象を検知する機能から直接管理者に対してアラートが発せられない場合は、対象を検知する機能のユースケース記述を書かなくてもよい。しかし、このユー

スケース記述を書くことは、管理者が誤検知と判断した際に、システムのどの部分を修正すればよいのかを調べる手がかりとなる。この場合ユースケース記述は、管理者ではなくアラートを受け取るシステムが要求している内容を書く。

表 1: ユースケース図との対応表

要求仕様の要素	ユースケース図の要素
管理者	アクター
機能	ユースケース
アラートの流れ	関連
システムの範囲	境界枠
個別具体的条件	シナリオ

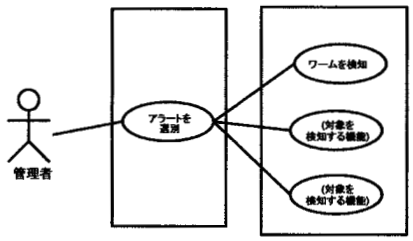

図 2: IDS の要求仕様 1

表 2: ユースケース記述との対応表

要求仕様の要素	ユースケース記述の要素
機能	ユースケース名
定義および概要	概要
予備動作の検知	事前条件
影響の有無	事後条件
誤検知の単位	ビジネスルール
注	注

4.4 問題領域が攻撃の種類の要求仕様

問題領域が攻撃の種類の要求仕様は、問題領域がIDSである場合の要求仕様において、対象を検知する機能が一つだけの場合である。

4.5 問題領域が攻撃の手法の要求仕様

攻撃の手法を問題領域とする場合の要求仕様はUMLのシーケンス図とステートマシン図をもとに表3と表4の対応表に基づいて表現する。UMLにおけるシーケンス図とはオブジェクト間の相互作用を時系列に沿って表現するための図であり、ステートマシン図はオブジェクトの状態の遷移とそれに応じた振る舞いを表現

機能　　：アラートを選別
定義および概要　：ワームを検知したアラートから要求仕様に応じてアラートを選別し管理者に伝える（ワームの定義）
予備動作の検知　：（予備動作を検知するかどうかを記述）
影響の有無による検知：（影響がないワームの挙動を検知するかどうかを記述）
誤検知の単位　：（誤検知の単位を記述）
注　　：（個別の条件）

図 3: IDS の要求仕様 2

するための図である。この問題領域の要求仕様は検知したい攻撃の手法の特徴を記述する。包含関係にある攻撃の種類の要求仕様もあわせて書き、その内のこの記述の特徴を満たすもののみを検知すべきである。

どんな攻撃であれ、マシン間のやり取りか、マシンの状態の変化という現象に過ぎない。つまり、IDSが検知対象とする事象をイベントと定めることで、イベントのやり取り、またはイベントによる状態の変化によって攻撃の手法は表現が可能である。具体的な手法を表現するに当たって、イベントのやり取りに着目した手法 [2] であれば、シーケンス図を利用して表現する。イベントによるマシンの状態の変化に着目した手法 [3] であればステートマシン図を利用して表現する。要求仕様において記述する必要があるのは、管理者が判断する根拠になる手法の特徴であるため、具体的なパラメータを記述する必要はない。シーケンス図であればどんな目的のイベントがやり取りされているのかが記述されていればよく、ステートマシン図であれば、判断の根拠になる状態かイベントのどちらかが記述されていればよい。

表 3: シーケンス図との対応表

要求仕様の要素	シーケンス図の要素
攻撃の種類	相互作用名
イベントのやり取りをする対象	ライフライン
イベントのやり取り	同期メッセージ

表 4: ステートマシン図との対応表

要求仕様の要素	ステートマシン図の要素
攻撃の種類	コンポジット状態の状態名
マシンの状態	状態
イベント	遷移

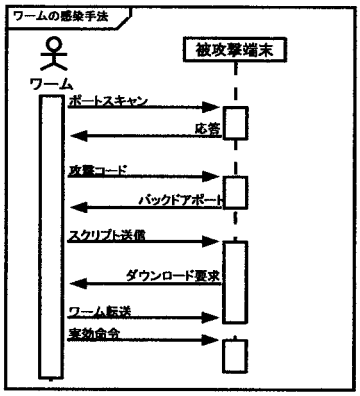

図 4: 攻撃の手法の要求仕様 1

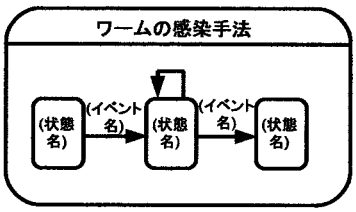

図 5: 攻撃の手法の要求仕様 2

4.6 要求仕様の有効性

要求仕様を記述することで、管理者の興味を明示することができる。本研究で提案した要求仕様では、問題領域という考え方と UML を用いたことで管理者の興味を柔軟に表現している。これにより、この記述から判断できることと、管理者の主観による判断が異なることをなくすことができる。

しかし、提案した要求仕様では管理者の興味を柔軟に表現できるよう、非形式的な記述を多く用いている。非形式的な記述は検証が容易なものだけとは限らず、複数の記述において整合性がとれているかどうかも保証しない。また、具体的な検証方法を示さないため、その部分にあいまいさは残る。

管理者は性能を評価する際に用いるデータについて、その内容を詳しく把握している必要がある。さもなければ、何を検知すべきだったのかを誰も示すことができない。要求仕様に記述され具体的な検証方法が示されないあいまいさは、内容の詳しい理解がなされていれば解消されるものと考える。

5 議論

IDS において誤検知の定義と誤検知かどうかの判断はきわめて重要である。アラートを発したこと、もしくは発しなかったことが誤検知と判断できたなら、設定や手法を修正していくことが可能である。しかし、誤検知であると判断できなければ、誤りでない設定や手法を変更する理由を運用から得ることはできず、修正することはできない。手法ごとに検証に都合の良い定義を用いてしまうと、管理者が本来望んでいる検知したいもの、検知したくないものと誤検知が異なってしまう場合がある。この章では典型的な IDS のタイプごとに、一般的な検知手法と誤検知の定義の方法、そして検知手法によらない要求仕様を定義することによるメリットを考察する。

5.1 シグネチャベース IDS

一般にシグネチャベースと呼ばれる IDS では、攻撃の特徴を記したシグネチャと呼ばれるパターンとパケットをマッチングすることで攻撃を検出する。このタイプの IDS では false positive を削減するために、シグネチャをネットワークに合わせ修正したり、複数のアラートをまとめるといった手法がとられている [4]。誤検知の定義は明示されないか、修正後のシグネチャ自身とされることが多い。

この方法では、明示されないのでは言うまでもなく、誤検知かどうかの判断を行うことができない。さらに誤検知の数を評価しようにも、何を数えれば良いのかさえ不明である。修正後のシグネチャ自身を誤検知の定義とした場合、記述の表現力の限界、シグネチャの不足、記述ミスなどによって見逃しや過剰なアラートを発したものに対して誤検知であると判断することができない。よってこのような誤検知の定義では、運用によって改善される余地が一切ないシステムということになってしまう。

本研究で提案したように管理者の本来の要求を要求仕様として記述し、それを定義とすれば、シグネチャを修正した後でも検知したいものが検知されていないとき誤検知であると判断でき、運用によって表現能力の不足や記述ミスを修正していくことができる。

5.2 統計ベース IDS

一般に統計ベースと呼ばれる IDS では、過去の動作の学習により正常な動作を判別する規則を作り、その規則沿って異常を検知し攻撃と判断する [5]。そのため、このタイプの IDS では攻撃などがない状態で学習を行

い、攻撃のある状態の動作と比較させ false positive と false negative を評価したり、再度攻撃のない他の状態で動作させ false positive を評価したりする。

このタイプの IDS では、学習時の状態を正常な状態としているため、攻撃とは何かを明示しないで評価を行う場合がある。攻撃を明示しなくても、相対的な評価を行うことはできる。しかし、何を攻撃として捉え、検知できるかは不明である。また、攻撃の定義を学習によって作られた規則とすると、検知すべきものは、学習によって作られた規則に反するものということになり、この規則に沿っているものは検知すべきではないことになる。つまり、この定義において誤検知は発生しないことになり、運用による修正はできない。

そもそも学習を攻撃のない状態で行わなければいけないのなら、攻撃とは何を想定しているのかを明示することは必要である。本研究で提案した要求仕様を記述しておけば、攻撃とは何を想定しているかが明確になるだけでなく、規則をすり抜けるような攻撃があった場合でも誤検知と判断することが可能である。

5.3 仕様ベース IDS

一般に仕様ベースと呼ばれる IDS では、プログラムの正常な動作を記述させ、プログラムがその仕様に沿って動作しているかどうかを検査することで攻撃を検知する。そのため仕様を記述するのが管理者の場合、負担が大きいことが知られており、人為的なミスが発生しえる。未知の攻撃を検知できたり、false positive の発生が少ないなどの利点があるが、仕様の書き方によっては false negative を多発させることもありえるだけでなく、オーバーヘッドが大きいという問題もある。

攻撃の定義を仕様以外の状態にさせるものと定義してしまうと、人為的なミスや仕様の表現方法の限界によって、取り逃したり検知してしまったものを誤検知と判断することができない。また、統計ベースと同様、検知方法が正常な状態との比較であるため、管理者の興味の範囲を明確にしなければ、何を攻撃と捉えているのかが不明である。本研究で提案した要求仕様を記述しておけば、攻撃とは何かを明示でき、人為的なミスや仕様の書き方により規則をすり抜けるような攻撃があった場合でも、誤検知と判断することが可能である。

6 おわりに

本研究では誤検知の定義として、管理者の興味を検知手法によらない方法で表現することの重要性を述べ、管理者の興味を IDS の要求仕様と考えた。そしてこの要求仕様を柔軟に表現をするために UML を用いて記述する方法を提案した。UML を用いることで理解しやすく、誤認することを少なくすることができる。また、非形式的な記述を用いており、管理者の興味の柔軟な表現が可能になった。これにより、誤検知数や誤検知率を論じる際に何を検知すべきで、何を検知すべきでないかを統一的に示すことができる。

今後の課題としては、この要求仕様の表現方法の改善が挙げられる。提案した手法は、柔軟な表現を可能にするために非形式的な記述を多く含んでいる。非形式的な記述は柔軟な表現を可能にするためにも必要であるが、複数の記述の整合性が保証されておらず、記述によってはあいまいな基準、不十分な基準になってしまう可能性がある。十分な基準として記述される必要がある要素をさらに洗い出し、表現方法を改善していく必要がある。

参考文献

[1] "Object Management Group -UML-" , http://www.uml.org/

[2] 前田秀介、馬場達也、大谷尚通、角将高、稲田勉、"感染プロセスに着目したワーム拡散防止システムの実装と評価", 情報処理学会研究報告,2006-DPS-126 2006-CSEC-32, pp287-292(2006)

[3] 鈴木功一、松本隆明、高見知寛、馬場達也、前田秀介、西垣正勝、"自己ファイル READ の検出による未知ワーム・変異型ワームの検知方式の提案", 情報処理学会研究報告,2006-DPS-126 2006-CSEC-32, pp275-280(2006)

[4] 田辺光昭、勅使河原可海、"ネットワーク型侵入検知システムにおけるアラートベースシグネチャの実現方式", 情報処理学会研究報告,2004-CSEC-27, pp.63-68(2004)

[5] 宮地玲奈、小宅宏明、川口信隆、重野寛、岡田謙一、"機械学習によるネットワーク型 IDS の false positive 削減手法の提案", 情報処理学会研究報告, 2003-CSEC-21,pp.53-58(2003)