2005－CSEC－31（2）
2005／12／9

Redundancy-control strategy を用いた自己改変型ウィルス検出の高速化

安藤類央　武藤佳恭
慶應義塾大学政策メディア研究科
〒２５２０８１６　神奈川県藤沢市遠藤５３２２
{ruo,takefuji}@sfc.keio.ac.jp

あらまし　本論文では、ウィルス・ワームの自己改変による検出回避手法への対応策として、形式的検証をベースにした導出モデルに、ＲＥＤＵＮＤＡＮＣＹ－ＣＯＮＴＲＯＬ戦略を適用することで、検出の高速化を行う。提案する導出型ウィルス検出では、従来の逐次処理で対象プログラムを追跡するコンパイラ型のデバッグ手法では発見が困難な自己改変型コンピュータウィルスを、インタプリタ型の処理が可能な形式に再表現を行うことで、数十行の定理群によって感染処理を構成する操作を検出することが可能であることを示す。評価実験では、ＦＯＬ（一階述語論理）定理証明系の上で導出モデルを構築し、ＲＥＤＵＮＤＡＮＣＹ－ＣＯＮＴＲＯＬ戦略を適用することで、高速化を行い、現実的な計算コストで同ウィルス検出が可能になることを示した。

Faster metamorphic computer virus detection using redundancy control strategy

Ruo Ando, Yoshiyasu Takefuji
Graduate School of Media and Governance,Keio University,
5322 Endo Fujisawa, Kanagawa, 252 Japan
{ruo,takefuji}@sfc.keio.ac.jp

Abstract: In this paper we propose a resolution based detection method for detecting metamorphic computer virus. Our method is the application of formal verification using theorem proving, which deduce parts of viral code from the large number of obfuscated operations and re-assemble those in order to reveal the signature of virus. To make our detection method more feasible and effective, redundancy-control strategies are applied for the resolution process. In this paper the strategies of demodulation and subsumption are applied for eliminating the redundant path of resolution. Experiment shows that without these strategies, resolving metamorphic code into several simplified operations is almost impossible, at least is not feasible in reasonable computing time.

１はじめに

インターネットの普及は、情報通信システムは、企業や公的機関の業務インフラとして活用される現状をもたらした。これに伴い、セキュリティインシデントの被害が増加している。特に、電子メールやＷＥＢサイトの閲覧などによって感染するウィルスの被害は、ＩＰＡの報告によると、２００３年度には１７４２５件、２００４年度には５２１５１件となっており、前例にないペースで増加している。また、攻撃手法も高度化しており、自己を暗号化あるいは一部を改変することで、感染のたびに実行ファイルの形態を変えるウィルスは、従来のシグニチャベースの方式では対応が難しいという点で、今後感染が瞬時に拡大する可能性がある。本論文では、形式的検証の手法を用いて、自己改変型のウィルスの検出を提案する。具体的には、対象となるコードのウィルスとしての性質を、定理証明系を用いて導出する方法を示す。

２関連研究

不正なコードの検出には、シグニチャ／ストリングマッチングが一般的に用いられ、有効である。実行ファイルや入出力に対するシグニチャマッチングを用いないコンピュータウィルスの検出は、広義にはソフトウェアの検証と捉えことができる。ソフトウェアの検証

には、モデル検査と定理証明の２つがある。これらの先行研究は基本的にはＣ／ＪＡＶＡなどの高級言語で書かれたプログラムや、アセンブリコードの構文解析を行う[5][6]。モデル検査を利用した不正コード検出には、バッファオーバーフローを扱ったものがある[7]。その他には、スタックを抽象化したグラフを扱う研究[8]がある。

３検出対象の定義

コンピュータウィルスは、感染エンジンの機能によって、エントリポイント感染型、圧縮／ステルス型、ポリモーフィック／メタモーフィック型の３つに分類される。本論文では、自己改変型ともいわれるメタモーフィックウィルスの検出を扱う。

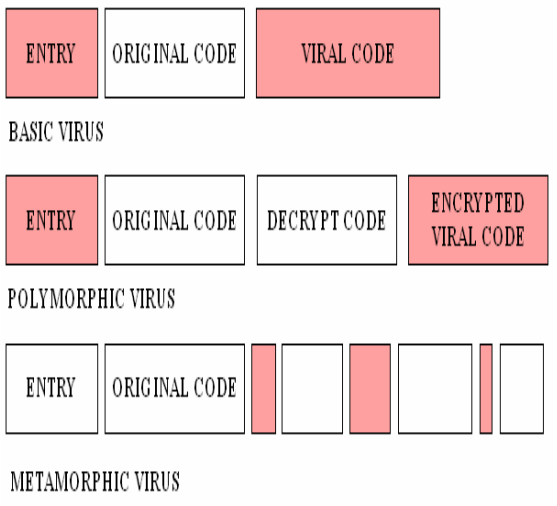

図 １ ３種類のウィルスの構造

図１は、従来（追加感染型）のウィルスと、ポリモーフィクウィルス、メタモーフィックウィルスを図示したものである。他の２つのウィルスに比べ、メタモーフィックウィルスはエントリポイントの変更を隠蔽し、感染ごとに異なる命令パターンを用いて自己を変更するため、非常に検出が難しくなる。本論文では、この自己改変された命令群に対して、述語論理による導出を行い、冗長化のための命令を除去し、複数の命令パターンを１つの操作に変換する方法を提案する。

４提案手法

提案システムは、レジスタへの代入やスタック操作、その他の命令を節形式で表現し、これを定理群とする。その後、自己改変型のウィルスのアセンブリコードを節形式で表現し、これに定理群を適用することで、ＡＰＩの呼び出しや復号ルーチンなどのウィルスの一部を構成する操作の命令シーケンスを導出することで検出完了とする。

４．１　検出方法

処理対象となるコードが、検出の対象となる性質を持つかどうかチェックするためには、性質を表現した節の否定形を用意し、定理の適用によってその節表現が導出されることで検出の終了とする。

具体的に、論理プログラムによる検出プロセスが終了するためには、検索の過程で、単位矛盾を生じさせる必要がある。単位矛盾とは、厳密には改変前と改変後のアセンブラコードを形式化した節同士が共通するリテラルを持ち、双方のリテラルが逆の符号を持ち、単一化可能な状態をさす。換言すると、メタモーフィック化される前のコードを否定した負の節を設定し、導出によって同じリテラルを持つ節が生成された時点で等価性が証明されたとし、検出を終了する。

４．２　Redundancy control strategy

冗長性制御戦略（Redundancy control strategy）は、主に等価代入による節の簡略と、より保持された節のうち、より導出に適した節の選択という、２つの操作に分類される。

４．２．１　包摂

述語論理を用いた定理証明では、目標とする節を導出する過程で、いくつかの節が保持され、新しい節が生成された時点で、過去に保持された節との間で、定理が適用される。この保持されている節のうち、より一般的な節を残す処理を包摂(subsumption)という。

OLDER(mother(x),x).
は、
OLDER(mother(hanako),hanako).

を包摂する。

また、導出を含めた包摂として、

-WIFE(hakako,tao) | FEMALE(hanako)
は、
FEMALE(hanako)
に包摂される。

このように包摂は、重複している節と、利用可能（保持されている）節のうち、より一般的でない節を除去する処理である。本論文では、後述する評価実験において、この戦略の有効性を示す。

４．２．２　デモジュレーション

デモジュレーションとは、あらかじめ等価代入（書き換え）を行うための節を定理証明系に与えて、処理節群の簡略化あるいは正準化を行う処理である。

T　←　EQUAL(R,S)

デモジュレータ EQUAL(R,S)を項 T に適用する際には、T が R か S いずれかの具体例であることが条件となる。デモジュレーションによる操作は等価代入により対象となる情報を簡略化または正準化することが目的である。等価代入を引き起こす節は、デモジュレータと呼ばれ、以下のように記述する。等価代入を行う検証の特徴は、対象とする情報の性質を反映したデモジュレータを追加することにより、より効率的な情報の検索が行えるところにある。本論文では、この手法を用いて自己改変されたアセンブラコードから、特定の感染操作を行うコードを導出する。

５　評価実験と検出対象の定式化

自己改変型ウィルスが用いる手法としては、レジスタ置換、マジックナンバー改変、そして冗長なループ挿入の３つに大きく分類される。

図 2 は、この３種類の手法が適用される場所を示したものである。自己改変型ウィルスでは、エントリポイント関数についてレジスタ置換や冗長な命令の挿入を行って改変の隠蔽を行い、本体部分（ペイロード）では冗長なループや命令を挿入し、分岐命令を利用してこれらが感染処理に影響を与えないようにコードで生成される。

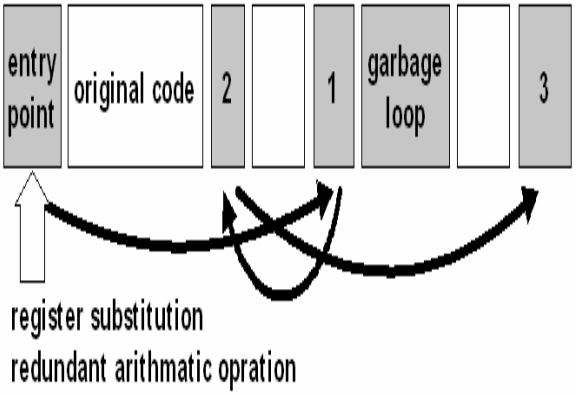

図 2　自己改変型ウィルスの構造とその手法

今回、本論文では、評価実験の対象として、プロセス操作関数の１つである GetModuleHandleA を扱った。このＡＰＩは５つの命令から構成される。表は、このアセンブラコードを定式化したものである。

アセンブラコード	定式化後
mov dword_1,A	**state(VAR(dword_1), const(A),v,Tim(1))**
mov dword_2,B	**state(VAR(dword_2), const(B),w,Tim(1))**
mov dword_3,0	**state(VAR(dword_3), const(0),x,Tim(1))**
push offset dword_3	**state_push(var(dword_3), y,Tim(1))**
call ds: GetModule Handle	**state(call (GetModuleHandle), z,Tim(1))**

表 1 アセンブリコードの定式化

提案手法では、各節ごとに状態を生成し、この状態を表現した節群に対して一回述語論理の定理の適用を行う。

表２と表３は、レジスタ置換による自己改変コードに対する導出結果である。このケースでは、レジスタを置き換えることで、操作後の各レジスタの状態は同じになるが、経由するレジスタが異なるため、シグニチャが感染の度に変化する。

オリジナルの行数	2 clauses	
自己改変後の行数	14 clauses	
	高速化なし	提案手法
処理した節数	59	45
生成された節数	93	60
等価代入節数	29	19
包摂された節数	0	16
SoS による包摂回数	0	5
保持された節の数	92	43

表 2　方法１：レジスタ置換型Ｉの導出結果

オリジナルの行数	2 clauses	
自己改変後の行数	8 clauses	
	戦略なし	提案手法
処理した節数	>1000	25
生成された節数	>1000	24
等価代入節数	>1000	8
包摂された節数	>1000	6
SoS による包摂回数	>1000	1
保持された節の数	>1000	17

表 3　方法２：レジスタ置換型ＩＩの導出結果

表３は、ＷＩＮＤＯＷＳオペレーティングシステム内での特定のＡＰＩのアドレスや、攻撃が成功した際に特定のアプリケーションを実行する機械語をシグニチャとして、検出する方法に対して、冗長な算術演算命令の挿入によりマジックナンバーを変更して、検出を回避する方法に対する導出結果を示したものである。

オリジナルの行数	1 clauses	
自己改変後の行数	5 clauses	
	戦略なし	提案手法
処理した節数	>1000	19
生成された節数	>1000	25
等価代入節数	>1000	12
包摂された節数	>1000	9
SoS による包摂回数	>1000	2
保持された節の数	>1000	15

表 4　方法３：マジックナンバー改変型

表５は、ジャンプ命令を利用することで、各オペレーションコードの順序を変えると語同時に、ジャンプ命令直後に実行されないコードやループを挿入し、パターンマッチを回避する方法の検出結果である。

オリジナルの行数	0 clauses	
自己改変後の行数	3 clauses	
	戦略なし	提案手法
処理した節数	24	14
生成された節数	43	24
等価代入節数	18	9
包摂された節数	0	2
SoS による包摂回数	0	2
保持された節の数	43	22

表 5　方法４：冗長な命令とループの挿入

以上、３種類の手法を総合して、GetModuleHandleA の呼び出しの複雑化（隠蔽）を検出し、表１の元のコードに簡略化する導出処理の結果を表６に示す。

オリジナルの行数	5 clauses	
自己改変後の行数	26 clauses	
	戦略なし	提案手法
処理した節数	>1000	203
生成された節数	>1000	293
等価代入節数	>1000	143
包摂された節数	>1000	112
SoS による包摂回数	>1000	25
保持された節の数	>1000	180

表 6　方法５：方法１から４への総合

６まとめと今後の課題

本論文では自己改変型のウィルスのアセンブラコードの解析に、導出ベースの定理証明系を用いた検出法の提案と評価実験を行った。複雑化した感染コードの簡略化あるいは正準化の過程で、冗長性制御戦略（redundancy-control strategies）を適用することで、検出の高速化が行えることを示した。今後の課題としては、処理対象としてはより規模の大きく複雑なウィルスコードを扱うこと、また、適用アルゴリズムとしてはパラモ

ジュレーション（等号調整代入）などの他の推論規則を応用する余地が残っている。

謝辞

本研究は文部科学省２１世紀ＣＯＥプログラム「次世代メディア・知的社会基盤」の支援を受けている。

参考文献

[1]Peter Szor and Peter Ferrie, "Hunting for Metamorphic," Virus Bulletin Conference, September 2001, pp. 123-144.
[2] Stephen Pearce, “Viral Polymorphism”, paper submitted for GSEC version 1.4b,2003
[3] Diomidis Spinellis. Reliable identification of bounded-length viruses is NP-complete. IEEE Transactions on Information Theory, 49(1):280-284, January 2003.
[4] Szor, Peter. "Attacks on Win32 - part 2." Virus Bulletin Conference, September 2000.
[5] Mihai Christodorescu, Somesh Jha, Sanjit A. Seshia, D.Song, and R.E. Bryant, ”Semantics-Aware Malware Detection”, in Proc IEEE Symposium on Security and Privacy, Oakland, May 2005
[6] Young H. Cho and William H. Mangione-Smith,” High-Performance Context-Free Parser for Polymorphic Malware Detection”, in Proc Advanced Networking and Communications Hardware Workshop,2005
[7] Chaki, Sagar & Hissam, Scott,”Precise Buffer Overflow Detection via Model Checking”, white paper of PACC.
[8]Arun Lakhotia, eric Uday Kumar , “Abstract Stack Graph to Detect Obfuscated Calls in Binaries”, in Proc Fourth IEEE International workshop on Source Code Analysis and Manipulation, 2005.
[10]独立行政法人:情報処理推進機構:未知ウィルス検出技術に関する調査,
１５情経第１６７５号(2004)