シンボリック実行を活用したマルウェア解析作業の効率化の研究

Improving Malware Analysis By Using Symbolic Execution

窪　優司・ネットワーク分科会・情報セキュリティ大学院大学

Understanding and circumventing anti-analysis techniques embedded in malware binaries are crucial for malware analysis because recent malware often reveals the intended actions after checking analysis environment. In addition, revealing all the behaviors of malware is also important to understand the purpose of the attack using the malware. With symbolic execution, the constraints for hindering the analysis could be listed and revealed, and all the behaviors could be found. The goal of this research is circumventing the anti-analysis techniques and revealing all the behaviors of malware.

研究目的

マルウェア解析において、コンピュータの演算能力を活用したシンボリック実行技術の導入により、解析時間の短縮や解析作業の効率化を図ることを目的としている。

提案手法

シンボリック実行技術が備える
①分岐条件の収集と解の演算能力を解析妨害機能の回避
②実行経路探索機能を単純な実行では発現しない動作の解明に適用する。

寄与

- 解析妨害機能を回避するための条件の自動抽出の可能性を指摘
- 外部からの指令に基づき動作するマルウェアが持つ機能の全容解明の補助機能
- 高度な知見や経験がない解析者でも容易かつ短時間にマルウェアを解析できる可能性を指摘

1. シンボリック実行技術の概要

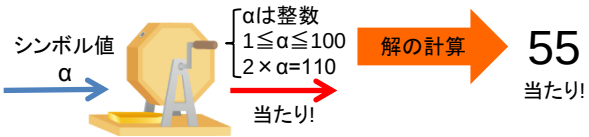
シンボリック実行は、プログラムが実行しうる経路を網羅的に抽出する技術であり、ソフトウェアの試験で試験用データの自動生成等に活用されている。
プログラムの入力として具体値を使用せず、その代わりに任意の値を表現するシンボル値と呼ばれる値を使用する。プログラムを模擬的に実行し、出力されるシンボル値に関する計算式を評価して、入力値が満たすべき条件と値を求める。

例）くじ引きプログラムの当選判定試験

- 総当たりによる試験

- 総当たりでは解の計算に非常に時間がかかる場合がある
  例）64ビットの整数値:2^{64}通り
- シンボリック実行では条件の抽出・収集が目的のため、解の計算の困難さは条件の複雑さに依存する

- シンボリック実行による試験

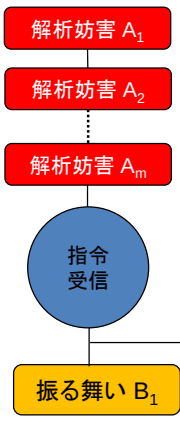


2.シンボリック実行技術の主要な問題点と対策例

問題点	問題の概要	対策例
動的ライブラリのシンボリック実行	マルウェアが呼び出す動的ライブラリの関数の内部までシンボリック実行を行うと処理が複雑になる	• APIのフック及び擬似応答 • 具体値での実行によるAPIの応答の近似
パス爆発とパス選択のヒューリスティックス	分岐やループの存在によって探索すべき実行経路の数が指数関数的に増加する	• ループ回数の制限 • 類似した経路の統合等による経路の削減
すべての分岐条件を満たす解の演算	収集した分岐条件が複雑なため、すべての条件を満たす解を求めることが困難	• 一部の条件式の近似または具体値での置換による条件の簡素化

3. シンボリック実行の適用を前提としたマルウェアのモデル

- 外部からの指令を受けて動作するマルウェアを対象とすると、解析のポイントは解析妨害機能の回避と未知の振る舞いの発見
- 解析妨害機能は、個々の独立な解析妨害機能が直列に結合したものと仮定
- 振る舞いは、個々の独立な振る舞いが並列に結合したものと仮定



- 解析妨害機能
  - 解析妨害機能(全体)A_{all}は個々の解析妨害機能A_i(i=1,2,…,m)から構成される
- 未知の振る舞いの発見
  - 振る舞い(全体)B_{all}は個々の振る舞いB_i(i=1,2,…,n)から構成される

$$A_{all} = A_1 \cap A_2 \cap \cdots \cap A_m,\ A_i \cap A_j = \phi$$
$$B_{all} = B_1 \cup B_2 \cup \cdots \cup B_n,\ B_i \cap B_j = \phi$$

4. 提案手法の評価方法

評価項目	評価方法
解析妨害機能	• 回避できた妨害の個数 • 回避条件の抽出
未知の振る舞いの発見	• 単純に実行した場合とは異なる振る舞いの個数 • 指令受信後に各振る舞いへ至る実行経路について、単純に実行した場合の経路とは異なる経路の数

5. 今後の方針

- シンボリック実行フレームワーク"angr"を使用し、その実行経路探索機能と分岐条件の収集および解の演算能力を活用して評価
- 動的ライブラリの処理およびパス爆発についてヒューリスティックな対策の適用