様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９　（共通）

科学研究費助成事業　　研究成果報告書

令和　５ 年　 ６ 月 ２９ 日現在

機関番号：５７４０３
研究種目：基盤研究(C)（一般）
研究期間：2019～2022
課題番号：１９Ｋ１１９１６

研究課題名（和文）自動解析を用いたMan-At-The-End攻撃に対するソフトウェアの保護

研究課題名（英文）Protecting Software against Man-At-The-End Attacks Using Automated Code Analysis

研究代表者
神崎　雄一郎（Kanzaki, Yuichiro）

熊本高等専門学校・電子情報システム工学系ＨＩグループ・准教授

研究者番号：９０４３５４８８

交付決定額（研究期間全体）：（直接経費）　　2,500,000円

研究成果の概要（和文）：本研究では，自動解析をともなうMan-At-The-End攻撃（端末を制御する権限を持つユーザによる，ソフトウェア等の解析・改ざん行為）からソフトウェアを保護する方法や，保護機構の有効性を評価する方法の開発に取り組んだ．主な成果は，SMTソルバを用いた命令列の自動生成機構に基づくコード難読化方法を開発したことである．提案方法は，LLVM IRのコードを対象に難読化を行う．提案方法が適用されたプログラムは，シンボリック実行を用いた自動解析攻撃への耐性が難読化前よりも高まる傾向があることを実験を通して確認した．

研究成果の学術的意義や社会的意義
Man-At-The-End攻撃からソフトウェアを保護する方法として，命令列の自動生成機構を用いたコード難読化方法を新たに提案した．また，難読化されたコードの機械学習を用いたステルス評価の方法など，ソフトウェア保護機構の有効性評価に関する方法についても検討した．これらの成果は，ソフトウェア保護の研究分野の発展に役立つものと期待される．また，LLVM IRのコード断片をSMTソルバによって自動生成する方法は，コードの自動生成技術を用いる他分野の研究への応用が期待できると考える．

研究成果の概要（英文）：The purpose of this study is to develop a method for protecting software against Man-At-The-End attacks that use automated code analysis techniques, and a method for evaluating the effectiveness of protection mechanisms. One of our achievements is to develop a method for obfuscating a program based on an automatic code fragment generation mechanism that uses an SMT solver. A system based on the proposed method is implemented as an LLVM pass. The experimental results show that the obfuscated programs tend to have more resistance to symbolic execution attacks than the original ones.

研究分野：ソフトウェア保護

キーワード：ソフトウェア保護　Man-At-The-End攻撃　難読化

様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９（共通）

１．研究開始当初の背景

Man-At-The-End 攻撃（MATE 攻撃）とは，あるハードウェア端末を制御する権限を持つエンドユーザが，何らかの利益を得るために，端末上で動作するソフトウェアや端末そのものを解析・改ざんする行為のことである[1]．ソフトウェアを構成するプログラムコードに機密性の高い情報が含まれる場合，その情報を MATE 攻撃からどのように保護するかは，ソフトウェアの開発者にとって重要な課題となる．

ソフトウェアに対する MATE 攻撃は，逆アセンブラやデバッガなどのバイナリを解析するツールによって得られる情報をもとに解析すべきコードを絞り込み，攻撃者自身がコードの意味を理解した上で，コードの解析や改ざんなどを行う，という手順で行われるのが典型的である．一方，近年のソフトウェア解析技術や計算機環境の向上にともない，シンボリック実行を応用した攻撃[2]（以下，シンボリック実行攻撃と呼ぶ）など，ソフトウェアの自動解析技術を用いた新たな MATE 攻撃の方法が出現している．このような MATE 攻撃は，攻撃の効率を飛躍的に向上させ得るものとして脅威となっており，自動解析を用いた MATE 攻撃からソフトウェアを保護する方法が求められる．また，ソフトウェア保護を目的とした数多くのコード難読化方法やツールが提案されている現在，ソフトウェア保護機構の有効性を実証的に評価する方法を検討することも重要であるといえる．

２．研究の目的

本研究の目的は，自動解析を用いた MATE 攻撃を困難にすることができるソフトウェア保護方法を開発すること，および，ソフトウェア保護機構の有効性を評価する方法について検討することである．本研究では，ソフトウェアを保護するための手段として，コード難読化（code obfuscation)が用いられることを前提とする．ここでコード難読化は，ソフトウェアを構成するプログラムコードを，コードの機能を保ったまま解析が困難なものに変換する方法のことを指す．

３．研究の方法

自動解析を用いた MATE 攻撃に対するソフトウェア保護方法として，プログラムコードの命令列の自動生成機構を用いたコード難読化方法を提案する．ここで命令列の自動生成とは，命令列の入出力例などの条件から，その条件を満たす命令列を SMT ソルバを用いて生成することを意味する．提案方法に基づく難読化システムを実装し，そのシステムによって難読化されたプログラムについて，難読化前のプログラムと同じ機能が保たれているか，また，シンボリック実行攻撃への耐性が向上したかといった点を評価する．

また，MATE 攻撃に対するソフトウェア保護機構の有効性評価の一環として，難読化されたコードのステルス(stealth)を評価する方法について検討する．難読化されたコードのステルスは，難読化されていないコードとの区別のつきにくさ（攻撃者による発見の困難さ）を意味するもので，ここでは，難読化されたコードのステルスを機械学習を用いて評価することを考える．

４．研究成果

研究期間内に得られた主な成果として，(1) 命令列の自動生成機構を用いたコード難読化方法の提案と難読化システムの実装，(2) 難読化されたコードの機械学習を用いたステルス評価方法の提案，の 2 点が挙げられる．それぞれについて以下に説明する．

(1) 命令列の自動生成機構を用いたコード難読化システムの提案と実装

プログラムコードの命令列の自動生成機構を用いた難読化方法を提案し，それに基づく難読化システムを実装した．提案システムの概略を，図 1 に示す．提案システムは，アセンブリや中間表現（LLVM IR など）のレベルで記述されたプログラムコードを対象に難読化を行う．「コード変形モジュール」は，入力として与えられたコード中の特定の命令を複雑な命令列に置き換えることで，難読化されたコードを出力する．ここで「複雑な命令列」は，元来の命令と同様の意味を持つ，2 命令以上で構成される命令列（コードの断片）を指す．複雑な命令列は，「命令列生成モジュール」が，SMT ソルバを用いて自動生成する．具体的には，元来の命令の入出力例，命令列を構成する候補となる命令の挙動の情報，命令列長などを SMT ソルバに制約として与え，得られた解をもとに命令列を生成する．このような構成を持つ，LLVM IR（コンパイラ基盤 LLVM の中間表現）のコードを対象にした難読化システムを実装した．

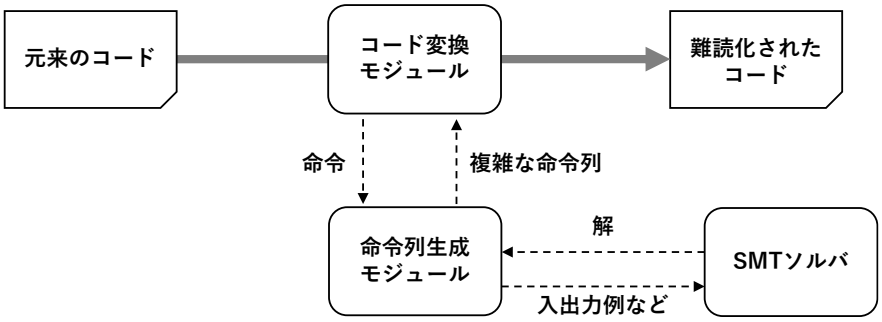

図 1　提案システムの構成

図 2 に示すのは，提案システムの命令列生成モジュールで実際に得られた，LLVM IR の加算命令「`add i32 %0, %1`」を置換できる長さ 12 の命令列である．このように，単純な命令から，同様の機能を持つ複雑な命令列を生成する．難読化対象のコードに含まれる特定の命令に対して，それぞれ複雑な命令列に置換することで，プログラムの解析を困難にすることを試みる．

```
%2  = xor i32 %1, %1
%3  = and i32 %0, %2
%4  = sub i32 %3, %1
%5  = add i32 %0, %4
%6  = xor i32 %5, %2
%7  = add i32 %0, %6
%8  = sub i32 %7, %0
%9  = sub i32 %8, %0
%10 = and i32 %2, %9
%11 = or  i32 %10, %4
%12 = add i32 %2, %11
%13 = sub i32 %0, %12
```

図 2　生成されたコードの例

提案システムによって複数のプログラムを対象に難読化を行い，難読化された各プログラムの動作の正しさや，シンボリック実行攻撃への耐性を確認する実験を行った．実験対象の各プログラムについて，難読化前の状態で命令網羅を達成するテストケースを用いてテストを行ったところ，いずれのプログラムもすべてのテストが成功した．このテストの範囲内では，難読化前と同じ動作（入出力関係）になることが確認できたといえる．また，難読化されたプログラムに対して，特定の出力が得られる入力値をシンボリック実行によって求めるのに要する時間を計測した．シンボリック実行ツールとしては，angr[3]および KLEE[4]を用いた．結果から，特に複数回繰り返して難読化を適用した場合において，シンボリック実行攻撃への耐性が高くなる傾向があることがわかった．

加えて，シンボリック実行による自動解析攻撃を失敗させることができる，計算量(難読化による実行時間のオーバーヘッド)が小さい難読化方法について，パスワードチェックを題材にした実験を通して議論した．

(2) 難読化されたコードの機械学習を用いたステルス評価方法の提案

機械学習のアルゴリズムの 1 つであるランダムフォレストを用いて，難読化されたコードのステルスを評価する方法について検討した．提案方法では，アセンブリコードが難読化によって変形されているかどうかを判定（2 クラス分類）するランダムフォレストのモデルを構築し，構築したモデルの予測結果に基づいて，任意のアセンブリコードのステルスを評価する．訓練データとして与える特徴は，コードを構成する各命令の出現頻度(コードの命令数で正規化したもの)とした．演算表現の複雑化や制御構造の平滑化などの既存の難読化方法を対象にした実験を行ったところ，実験対象とした難読化方法によって難読化されたコードの多くが，難読化されている（ステルスが低い）と判定されるという結果が得られた．

以上のように，MATE 攻撃からソフトウェアを保護する方法として，命令列の自動生成機構を用いたコード難読化方法を新たに提案し，それに基づくシステムの実装を行った．また，難読化されたコードの機械学習を用いたステルス評価など，ソフトウェア保護機構の有効性を評価する方法についても検討した．研究成果の多くは論文等で発表済であり，本研究課題の成果は，ソフトウェア保護の研究分野の発展に役立つものと期待される．

＜引用文献＞

[1] C. Collberg, “Code obfuscation: Why is this still a thing?”, Proceedings of the Eighth ACM Conference on Data and Application Security and Privacy, pp.173-174, 2018.
[2] S. Banescu, C. Collberg, V. Ganesh, Z. Newsham, and A. Pretschner, “Code obfuscation against symbolic execution attacks”, Proceedings of the 32nd Annual Conference on Computer Security Applications, pp.189-200, 2016.
[3] angr: https://angr.io/
[4] KLEE Symbolic Execution Engine: http://klee.github.io/

５．主な発表論文等

〔雑誌論文〕　計0件

〔学会発表〕　計12件（うち招待講演　0件／うち国際学会　2件）

１．発表者名 光本智洋，神崎雄一郎
２．発表標題 命令列の自動生成機構を用いたLLVM IRコードの難読化の試み
３．学会等名 情報処理学会 第84回全国大会
４．発表年 2022年

１．発表者名 光本智洋，神崎雄一郎
２．発表標題 SMTソルバを用いたコード変形に基づくプログラム難読化システムの実装
３．学会等名 ソフトウェアエンジニアリングシンポジウム2021
４．発表年 2021年

１．発表者名 光本智洋，神崎雄一郎
２．発表標題 SMTソルバによる命令列生成を用いたアセンブリプログラムの難読化
３．学会等名 情報処理学会 第83回全国大会
４．発表年 2021年

１．発表者名 北岡哲哉，神崎雄一郎，森川みどり，門田暁人
２．発表標題 難読化変形の機械学習による判別の困難さに関する実験的評価
３．学会等名 情報処理学会 第82回全国大会
４．発表年 2020年

１．発表者名
熊井優樹，神崎雄一郎

２．発表標題
SMTソルバを用いた演算表現の難読化方法の検討

３．学会等名
情報処理学会 第82回全国大会

４．発表年
2020年

１．発表者名
大槻成輝，玉田春昭，神崎雄一郎

２．発表標題
JVM環境におけるオペコード列と名前に着目した適用難読化ツールの特定

３．学会等名
2020年暗号と情報セキュリティシンポジウム（SCIS2020）

４．発表年
2020年

１．発表者名
玉田春昭，神崎雄一郎

２．発表標題
Javaバイトコードを対象とした命令列の頻度解析による適用難読化手法の特定

３．学会等名
コンピュータセキュリティシンポジウム2019

４．発表年
2019年

１．発表者名
北岡哲哉，神崎雄一郎，森川みどり，門田暁人

２．発表標題
ランダムフォレストを用いた難読化されたコードのステルス評価の検討

３．学会等名
第18回情報科学技術フォーラム（FIT2019）

４．発表年
2019年

１．発表者名 Toshiki Seto, Akito Monden, Zeynep Yucel, Yuichiro Kanzaki
２．発表標題 On Preventing Symbolic Execution Attacks by Low Cost Obfuscation
３．学会等名 20th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)（国際学会）
４．発表年 2019年

〔図書〕　計0件

〔産業財産権〕

〔その他〕

―

６．研究組織

	氏名 （ローマ字氏名） （研究者番号）	所属研究機関・部局・職 （機関番号）	備考

７．科研費を使用して開催した国際研究集会

〔国際研究集会〕　計0件

８．本研究に関連して実施した国際共同研究の実施状況

共同研究相手国	相手方研究機関