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

平成　２９　年　　６　月　１９　日現在

機関番号：　３４３１５
研究種目：　基盤研究(C)（一般）
研究期間：　2014～2016
課題番号：　２６３３０１６７

研究課題名（和文）CPUの機能拡張とOS連携によるバッファオーバフロー攻撃検出方式

研究課題名（英文）A Method for detection of buffer overflow attacks by cooperating between extended CPU and OS

研究代表者

毛利　公一（Mouri, Koichi）

立命館大学・情報理工学部・教授

研究者番号：９０３１３２９６

交付決定額（研究期間全体）：（直接経費）　　3,500,000円

研究成果の概要（和文）：ソフトウェアの脆弱性には，(1)Webアプリケーション，(2)バッファチェックの不備，(3)仕様上の不備，(4)文字列チェックの不備，(5)アクセス制御の不備などがある．本研究課題は(2)を対象としてその解決を試みるものである．(2)は広く使われるアプリケーションやライブラリに含まれている脆弱性のため影響範囲は広い．また，十分充実した開発体制で臨んでも減少しておらず，新たな技術をもって解決する必要がある．以上より，本研究では，このバッファチェックの不備に起因するバッファオーバフロー攻撃を，CPUの機能拡張とそれを支援するOSを開発することで根本的に解決できることを示した．

研究成果の概要（英文）：Top 5 of typical software vulnerabilities are (1)Web applications, (2) insufficient buffer checking, (3)insufficient software design, (4)insufficient string checking, and (5)insufficient access control. In this research, we focused on (2) and tried to solve the problem. (2) is very large impact because it is included in not only applications but also libraries. Furthermore many challenges have been taken to solve the problem but they could not do that. So, in this research, we proposed extended CPU and OS for supporting it. And we have shown the problems are solved by our proposal.

研究分野：　オペレーティングシステム

キーワード：　マルウェア対策　コンピュータセキュリティ　サイバー攻撃対策　エンドポイントセキュリティ　バッファオーバフロー

１．研究開始当初の背景

現代社会では，行政，企業，教育機関，個人が，情報共有・情報交換だけでなく，契約，決済，申請等の重要な目的にインターネットを活用している．すなわち，インターネットはインフラストラクチャとして十分定着した．一方で，接続されたコンピュータを攻撃し，営利目的・政治的目的・個人的怨恨・欲望・私刑の達成を試みる者が出現した．これらの者による脅威は現在も増しており，情報漏洩・改ざん，侵入，サーバダウン等のインシデントが頻繁に報告されている．2012 年上半期の国内の個人情報漏洩に関する推定損害賠償総額だけで約 348 億円[†1]であり，世界のインシデント全体では更に巨額になると予想できる．更なるセキュリティ向上が急務である．

現在もセキュリティ向上のために，ソフトウェアの脆弱性対策，ネットワークの脆弱性対策，暗号や認証の導入，教育や法律の強化，情報セキュリティマネジメントの導入が講じられている．いずれも重要な対策であるが，ソフトウェアの脆弱性を軽減できれば根本的で直接的な対策となるため，技術的側面からの研究開発は大変重要である．

＜参考文献＞

†1 日本ネットワークセキュリティ協会：「2012 年情報セキュリティインシデントに関する調査報告書」

２．研究の目的

ソフトウェアの脆弱性には，(1)Web アプリケーション，(2)バッファチェックの不備，(3)仕様上の不備，(4)文字列チェックの不備，(5)アクセス制御の不備などがある[†2]．文献[†2]によると，脆弱性の報告件数は(1)が 58%と最も多い．ただし，個々の Web サイト用に開発されたものが多く，件数は多いが 1 件が及ぼす影響範囲は狭い．また，粗製濫造されたものが多いとの報告もあり，まずは開発体制の充実・技量向上が必要である．次に報告件数が多いのは(2)で 5%である．一見少なく思えるが，広く使われるアプリケーションやライブラリに含まれているため，件数が少なくても影響範囲は広い．また，十分充実した開発体制で臨んでも減少しておらず，新たな技術をもって解決する必要がある．以上より，本研究では，このバッファチェックの不備に起因するバッファオーバフロー攻撃を新たな方式を提案することで根本的に解決することを目的とする．

バッファオーバフロー攻撃は，自動変数とサブルーチンからの戻り先アドレスがいずれもスタック領域に保存されることを悪用する．自動変数の領域を超えてデータを書き込むこと(以下，領域超え書込み)により，戻り先アドレスを書き換え，任意のアドレスのプログラムを実行させる．この攻撃の歴史は古く，数多く対策が提案されている．ソフトウェアによる対策として，カナリアコードに代表される，ソフトウェアに領域超え書込み検出機能を付加する技術があるが，実行速度の低下を招く問題がある．また，アドレス空間配置のランダム化(ASLR)に代表される方式は，スタック領域のアドレスを予想困難にするため，攻撃成功の可能性を低下させるが，効果を保証するものではない．さらに，ソースコードどおりにプログラムが実行されているか実行時に確認し攻撃成功を検出する手法も提案されているが，スタック偽装攻撃に弱い．また，ソフトウェアの実行前に高度な準備作業が必要であったり，実行時のオーバヘッドが大きいなどの課題もある．ハードウェアによる対策として，スタック領域など特定のメモリ領域に配置されたプログラムを実行禁止とする機能を持つ CPU があるが，一般的に用いられている実行時コンパイル(JIT)技術やシグナル処理を実現しようとすると当該機能を無効化する必要があるため，活用されていない．

以上で述べた背景から，本研究では，以下の(1)～(3)の特徴を有するバッファオーバフロー攻撃検出方式の実現を目的とする．なお，(a)～(d)は特徴を実現する具体的手法を示している．

(1) CPU の機能を拡張し，バッファオーバフローを自動検出する機能を実現する．
  (a) ハードウェアエミュレータの QEMU を活用
  (b) call 命令と ret 命令の拡張
  (c) 実行中のプロセス ID とスレッド ID を格納するレジスタとそれを操作する命令の追加
  (d) 検出時の例外割込みによる通知機能の追加
(2) OS が上記機能と連携することにより効果を発揮する．
  (a) 実行中のプロセス ID とスレッド ID を CPU へセットする機能の追加
  (b) 例外割込みハンドラの追加
  (c) プロプライエタリな OS への対応
(3) アプリケーションやライブラリ等の従来のソフトウェアはそのまま利用可能とする．
  (a) オープンソース OS とアプリケーションを用いた機能評価と性能評価
  (b) プロプライエタリな OS とアプリケーションを用いた機能評価と性能評価

本研究の本質的なアプローチはハードウェアによる対策である．ただし，実際に CPU の設計・製造はしない．QEMU を改変することによって CPU の機能拡張の実現可能性を実証する(1-a)手法を採り研究推進を図る．本研究では，インテル社の x86 アーキテクチャを対象とする．

サブルーチン呼出しでは，call 命令により戻り先アドレスがスタックに積まれ，ret 命令によりスタックから取り出されたアドレ

スへプログラムの制御が戻る．このcallとretの対称性を利用し，call時に積まれた値がret時に変化していないことをCPUが確認することで攻撃を検出する(1-b)．ただし，一般的にはスタックがスレッド単位で生成されるため，それらを正しく区別した上で検出処理をする必要がある．そこで，実行中のスレッドを区別するための仕組みを導入する(1-c)．このレジスタは，OSのプロセス管理部やスケジューラによってセットされる(2-a)．戻り先アドレスの書換えが検出されると，検出をOSに通知する必要がある(1-d)．またOSではそれを受けて，攻撃を受けたプロセスを停止させるなどの処理をする必要がある(2-b)．

(2-a)と(2-b)の実現にはOSのソースコードの改変が必要となるため，OSがオープンソースである必要がある．よって，当初はLinuxで研究を進める予定とするが，本研究が対象とする攻撃はWindowsに代表されるプロプライエタリなOSが対象となる場合が多い．そこで，Windowsにおいても，OS内部のイベント通知機構を活用して(2-a)と(2-b)に相当する機能を実現する(2-c)．

以上で研究開発した環境を用い，脆弱性を有するライブラリやアプリケーションを動作させた上で，実際のマルウェア(悪意あるソフトウェア)やネットワーク攻撃ツール(metasploit等)を動作させ，本研究の有効性と実用性を示す(3-a)(3-b)．

＜参考文献＞

†2 IPA・JPCERT/CC：「ソフトウェア等の脆弱性関連情報に関する活動レポート（2013年第2四半期）」

3．研究の方法

平成26年度の計画

研究のスタートアップ段階として，次の3点について調査・研究を進める．

(1) QEMUの内部構成について調査する．具体的には，後述の(2)を実現するための手法を調査する．QEMUはハードウェアエミュレーションの高度な技術が使われており，それらを詳細に記した文献が存在しない．よって，調査には長めの約8ヶ月を見積もっている．

(2) 上記(1)に基づいて，プロセスIDとスレッドIDを格納する専用レジスタを追加し，それらの操作をするための命令群を拡張する(1-c, 2ページ研究目的②で示した項目番号)．(1)が十分達成できていれば2ヶ月程度で実装と動作確認が可能であると見積もっている．

(3) OSによるプロセス・スレッドの生成・消滅時，スケジューラによる切替え時に，OSが上記(2)のレジスタに値をセットするようOSを拡張する(2-a)．実装自体は1ヶ月程度，その後の動作確認に1ヶ月程度を見積もっている．

1年目は上記のように，バッファオーバフロー攻撃を検知する機能を実現するための礎となる部分についての研究を進める．なお，(1)については，三菱電機情報技術総合研究所にQEMUをよく知る研究者が在籍しているため，適時に協力を仰ぐ計画としている．

平成27年度の計画

本研究の中心となる技術の確立を行う．具体的には以下の5つのステップで進める．

(4) call命令とret命令の拡張を行う．この拡張では，call命令実行時には戻り先アドレスをスタックに積むと同時に，CPU内の保護されたメモリ領域(保護領域)に戻り先アドレスをコピーする．ret命令実行時には，保護領域内の戻り先アドレスとスタック内の戻りアドレスを比較する(1-b)．なお，このような処理をスレッド単位で保護領域を切り替えながら比較したり，OSのような信頼できるプログラムの場合は比較を省略するなどの機能なども同時に検討・開発する．よって，約6ヶ月を見積もっている．

(5) 上記(4)の処理でアドレスを比較して異なる値であった場合は，バッファオーバフロー攻撃を受けていることが検出できるため，割込を発生させる処理をエミュレートしなければならない(1-d)．本機能はQEMU内に既に実現されている機能をベースに作成できると考えられるため，約1ヶ月を見積もっている．

(6) 上記(5)で発生した割込を受けて，OSは当該プロセスを終了させる等の処理を行う必要がある．そのための割り込みハンドラの実装を行う(2-b)．ハンドラの実装そのものは比較的容易であるため，約1ヶ月を見積もっている．

(7) 上記(4)～(6)を一連の動作として実現する必要があるが，OSの起動前後にはCPUが16ビットモードになったり，ユーザレベルでは用いない特殊なプログラミングテクニックを用いていることが多いため，その対処が必要となると予想している．約2ヶ月を見積もる．

(8) 以上で基本的な実装が完了するため，オープンソースのOSを利用した評価実験を実施する(3-a)．様々なアプリケーションやライブラリにて実験を行うため，約2ヶ月と見積もる．

2年目で，バッファオーバフロー攻撃の検知に必要な機能を実現し，本研究の基本的知見を確立することを目標とする．

平成 28 年度の計画

最終年度では，より広い適用分野へ本研究が適用できることを証明するために，次の 2 点を実施する計画とする．

(9) Windows に代表されるプロプライエタリな OS において提案機能が利用可能であることを示すために，上記(3)・(6)に相当する機能を Windows 向けに開発する(2-c)．そのための調査を含め，約 8 ヶ月を見積もっている．

(10) 最後にプロプライエタリな OS において，様々なアプリケーションやライブラリを用い，機能評価・性能評価のための実験を行う(3-b)．約 2 ヶ月を見積もっている．

以上で，本研究で得られる知見を，新規性・実用性ともに確立する．

４．研究成果

「２．研究の目的」で示した（１）～（３）に対応付けて研究成果について述べる．

（１）の CPU の機能を拡張では，ハードウェア QEMU を用いて CPU の命令を拡張または追加した．具体的には，call 命令と ret 命令を拡張した．call 命令では戻り先アドレスをセキュア領域に保存し，ret 命令では戻り先アドレスがセキュア領域に保存されたものとスタックに保存されたものを比較して書き換えがあればそれを検出できるようにした．

また，提案方式ではスタックを区別する必要があるため，スタックを生成する単位であるスレッドを CPU が区別できるようにするための新たな命令の追加とそのためのレジスタを追加した．具体的には，走行中のスレッドについての，プロセス ID とスレッド ID を格納するためのレジスタとそれを読み書きする命令を追加した．これによって，スレッドを採用している OS においても，call/ret 命令の拡張によるアドレスの書き換え検出機能が動作するようにした．

さらに，ret 命令によって戻り先アドレスの書き換えが検出された場合に，例外を発生するように拡張した．これによって，ソフトウェア（一般には OS）が攻撃の発生を検出でき，さらにプロセスの削除などの対処が可能となった．

（２）上述（１）の機能は OS が適切に運用しなければならない．まず，スレッド毎に生成されるスタックを，CPU が確実に区別するために，OS のプロセス管理モジュールおよびスケジューラを拡張した．具体的には，スレッドの生成・削除・切り替え時に，CPU に拡張されたプロセス ID・スレッド ID を格納するレジスタに適切な値をセットするようにした．これによって，CPU が適切なスタックを認識できるようになった．

また，ret 命令による戻りアドレスの書き換え検出時に CPU が発行する例外を，OS が受信してプロセスを強制終了させるための拡張を実現した．さらに，シグナル機能を利用してプロセス自身が例外をハンドルして対処することが可能となるような拡張も実現した．

上記については，Linux オペレーティングシステムを用いて実現した．研究計画では，プロプライエタリな OS でも実現する目標としていたが，これについては実現ができなかった．これは，プロプライエタリな OS では，スレッド生成・削除・切り替え時におけるレジスタへのプロセス ID とスレッド ID セットのプログラムの実現が時間的に困難であったことによる．ただし，実現方針については検討ができており，スレッド生成・削除・切り替え時に OS 内で発生するイベントを獲得して実現することができることを確認している．

（３）については，オープンソースである Linux オペレーティングシステムについての評価を実施した．その結果，従来のユーザプログラムがそのまま利用可能であることが確認された．

ただし，この時点で新たな課題が明らかになった．具体的には，一般的なユーザプログラムで用いられる下記の 3 点については，誤検出・検出漏れが発生する．

- シグナル処理
- エラーハンドリング処理
- 動的リンクライブラリ

本研究課題では予定していなかったが，これらの 3 点についても，対処法を示した．さらに，その対処を行った上で従来のユーザプログラムが動作可能であることを確認した．さらに，攻撃が発生した際にそれを検出できることも確認した．

５．主な発表論文等

（研究代表者、研究分担者及び連携研究者には下線）

〔雑誌論文〕（計 5 件）

①奥野 航平, 内匠 真也, 大月 勇人, 瀧本 栄二, 毛利 公一, コンパイラを用いた情報フロー制御による情報漏洩防止機構, 査読有, 情報処理学会論文誌, Vol. 57, No. 12, pp. 2836-2848, 2016.

②大月 勇人, 瀧本 栄二, 齋藤 彰一, 毛利 公一, ブランチトレース機能を用いたシステムコール呼出し元識別手法, 査読有, 情報処理学会論文誌, Vol. 57, No. 2, pp. 756-768,

2016.

③内匠 真也, 奥野 航平 , 大月 勇人, 瀧本 栄二, 毛利 公一, コンパイラと OS の連携によるデータフロー追跡手法, 査読有, 情報処理学会論文誌, Vol. 56, No. 12, pp. 2313-2323, 2015.

④ Yuto Otsuki, Eiji Takimoto, Shoichi Saito, Eric W. Cooper, and Koichi Mouri, Identifying System Calls Invoked by Malware using Branch Trace Facilities, International MultiConference of Engineers and Computer Scientists 2015 (IMECS 2015), Vol. 1, pp. 145-151, 2015.

⑤大月 勇人, 瀧本 栄二, 齋藤 彰一, 毛利 公二, マルウェア観測のための仮想計算機モニタを用いたシステムコールトレース手法, 査読有, 情報処理学会論文誌, Vol. 55, No. 9, pp. 2034-2046, 2014.

〔学会発表〕(計 4 件)

①松本 隆志, 明田 修平, 瀧本 栄二, 齋藤 彰一, 毛利 公一, 動的テイント解析機能を利用した OS による細粒度データ出力制御手法, 情報処理学会研究報告 Vol. 2016-CSEC-75, No. 1, pp. 1-8, 2016.12.1, はこだて未来大学 (北海道函館市).

②久保田 曹嗣, 明田 修平, 瀧本 栄二, 齋藤 彰一, 毛利 公一, CPU によるリターンアドレス書換え攻撃検知とソフトウェア支援, コンピュータセキュリティシンポジウム 2016(CSS2016)論文集, pp. 718-725, 2016. 10.12, 秋田キャッスルホテル (秋田県秋田市).

③松本 隆志, 大月 勇人, 明田 修平, 瀧本 栄二, 齋藤 彰一, 毛利 公一, Argos の動的テイント解析機能を利用した OS による情報漏洩防止手法, 情報処理学会研究報告, Vol. 2016-CSEC-72, No. 33, pp. 1-8, 2016.3.4, 明治大学駿河台キャンパス (東京都千代田区).

④柴田 達也, 奥野 航平, 大月 勇人, 瀧本 栄二, 毛利 公一, QEMU を用いた命令拡張によるリターンアドレス書換え攻撃検知手法, 情報処理学会研究報告, Vol. 2015-CSEC-68, No. 33, pp. 1-8, 2015.3.5, 法政大学 (東京都小金井市).

〔その他〕
ホームページ等
http://www.asl.cs.ritsumei.ac.jp/

６．研究組織
(1)研究代表者
毛利 公一 (MOURI, Koichi)
立命館大学・情報理工学部・教授
研究者番号：90313296