A-019

16 進数コードの出現状況に着目したバッファオーバーフロー攻撃の特徴抽出
Feature extraction of Buffer Overflow Attacks based on distribution of hexadecimal codes

南後 吉秀 [1]　松田 健 [2]　園田 道夫 [1]　趙 晋輝 [1,3]
Yoshihide Nango　Takeshi Matsuda　Michio Sonoda　Jinhui Chao

1. はじめに

近年, インターネットの急速な普及により, 老若男女問わずインターネットが使われる時代となった. しかしながらその一方で, ネットワーク上でのサイバー攻撃も年々増加し続ける状況下にある. 特に, システムに被害を与える不正アクセスの原因の一つとされているのが, 古典的な攻撃であるバッファオーバーフロー攻撃と呼ばれるものである[1]. バッファオーバーフロー攻撃対策における既存手法としてシグネチャマッチング方式がよく知られている. この方式は侵入検知システム(IDS)において用いられていることが多く, 例えば IDS としてよく知られている Snort の場合では, セキュリティホールに対する攻撃パケットの固有な部分と攻撃の根幹を成すシェルコードと呼ばれる部分の特徴的な部分に関わるものをシグネチャと呼ばれるデータベースに登録しておき, シグネチャの内容とネットワーク上を流れるパケットの内容とを比較することによって攻撃検出を行っている. しかしこの方式においては, 未知のセキュリティホールの検出ができず, さらにシェルコードのパターンが無限に作成可能であるがゆえにすべての攻撃の検出が原理的に不可能である, ということが欠点として存在する[1].

そこで本論文では, インテル x86 アーキテクチャ環境下でバッファオーバーフロー攻撃を引き起こすシェルコードにおける 16 進数コードの出現状況に着目しながら, アセンブラ命令レベルのシェルコードにおけるアセンブラ命令の出現ペアを出現頻度の観点から解析することによって, バッファオーバーフロー攻撃に用いられるシェルコードの特徴を見出す手法について検討及び考察する.

2. シェルコードとバッファオーバーフロー

2.1 シェルコード

シェルコードはシェルを起動してコマンドを受け付けるようにするプログラムである. コンピューターセキュリティの分野におけるシェルコードは, ソフトウェアの脆弱性を利用するペイロード(パケットの中でヘッダーを除いた部分のこと)として使われるコード断片であり, バッファオーバーフローを効果的に利用するためにシステムに送られるものでもある. シェルコードをシステムに応じてカスタマイズすることによって, 脆弱性を抱えたプログラムから制御を奪い取ることができるようになっている. これによって行われる操作の一例として, コンピューターに管理者アカウントを追加したり, ログファイルから記録を抹消したりすることが挙げられる[2]. シェルコードは機械語の命令列で構成されたコードであることから, 16 進数で記述されていることが多い.

1　中央大学大学院理工学研究科情報工学専攻
Department of Information and Systems Engineering, Graduate School of Science and Engineering, Chuo University
2　静岡理工科大学総合情報学部コンピュータシステム学科
Department of Computer Science, Faculty of Comprehensive Informatics, Shizuoka Institute of Science and Technology
3　中央大学理工学部情報工学科
Department of Information and Systems Engineering, Faculty of Science and Engineering, Chuo University

2.2 バッファオーバーフロー

バッファオーバーフロー(バッファオーバーラン)とはよく知られているセキュリティホールの一種である. バッファオーバーフローによる脆弱性はコンピューターの黎明期から現在に至るまで見かけることができ, Internet Explorer をはじめとするプログラムで発見されたゼロデイ脆弱性でもバッファオーバーフローが用いられている[2]. バッファオーバーフロー攻撃はバッファオーバーフローを利用した攻撃であり, 一例としてサービス妨害攻撃と呼ばれる DoS 攻撃(Denial of Service attack)や, DoS 攻撃にとってかわった DDoS 攻撃(Distributed Denial of Service attack)が存在する. バッファオーバーフロー攻撃による被害の事例として, 2000 年 1 月から 2 月にかけて続発した中央省庁 Web サイトの改ざんが知られている[3].

3. 攻撃コードで頻繁に用いられるアセンブラ命令

本研究を進めるに当たり, Linux におけるバッファオーバーフロー攻撃で用いられるシェルコードを対象として, 攻撃コードで頻繁に用いられているインテル x86 アーキテクチャのアセンブラ命令について, 実際のシェルコードデータを閲覧することによって調査した. この調査で用いたシェルコードは, シェルコードデータベースとして知られている shell-storm に存在するシェルコードデータ 100 個である. これらのシェルコードデータに対して調査を行った結果, 攻撃コードで頻繁に用いられるアセンブラ命令として, 次の 6 つが得られた.

- INT 命令 (Call to Interrupt Procedure)
  割り込み処理を呼び出すためのアセンブラ命令で, 代表的な 16 進数のオペコードは CD である.
- LEA 命令 (Load Effective Address)
  実効アドレス(制御装置が命令を実行する際に実際に使用するアドレス)をロードするためのアセンブラ命令で, 代表的な 16 進数のオペコードは 8D である.
- MOV 命令 (Move)
  データを転送するためのアセンブラ命令で, 代表的な 16 進数のオペコードは 89 と B0 である.
- POP 命令 (Pop a Value from the Stack)
  スタック領域からデータを取り出すためのアセンブラ命令で, 代表的な 16 進数のオペコードは 58, 59, 5A, 5B, 5C, 5D, 5E, 5F である.

・PUSH 命令 (Push Word of Doubleword onto the Stack)
スタック領域へデータを積むためのアセンブラ命令で, 代表的な 16 進数のオペコードは 50, 51, 52, 53, 54, 55, 56, 57, 6A, 68 である.
・XOR 命令 (Logical Exclusive OR)
排他的論理和をとるためのアセンブラ命令で, 代表的な 16 進数のオペコードは 31 である.

ここで, オペコードとはマイクロプロセッサーなどに与える機械語の命令の識別番号のことである.

4. アセンブラ命令ペアの出現頻度解析

本研究においては, シェルコードデータ中に存在するアセンブラ命令ペアの出現頻度を次の方法で解析している. シェルコードデータの先頭を始点として, 着目しているアセンブラ命令(遷移元命令)の次に出現しているアセンブラ命令(遷移先命令)を確認する. 遷移先命令を遷移元命令とみなして新たな遷移先命令の確認を繰り返しながら, 遷移元命令と遷移先命令のペアの出現頻度を数えていく. ここで, 前節で挙げた 6 つのアセンブラ命令のみで構成されるペアを主な解析対象としたが, これ以外のアセンブラ命令が含まれるペアの出現頻度も解析した. このようにして, シェルコードデータベースの shell-storm に存在する攻撃データ 50 個に対して解析したところ, アセンブラ命令ペアの出現頻度は次の表 1 と表 2 の通りであった. なお, スペースの都合により 2 つの表に分割しており, いずれの表も行方向が遷移元命令を表しており, 列方向が遷移先命令を表している.

表 1　攻撃データ 50 個におけるアセンブラ命令ペアの出現頻度(1)

	INT	LEA	MOV	POP
INT	0	0	28	3
LEA	4	3	7	0
MOV	71	10	70	1
POP	23	0	10	2
PUSH	33	6	137	53
XOR	7	0	34	0
その他	6	1	32	12

表 2　攻撃データ 50 個におけるアセンブラ命令ペアの出現頻度(2)

	PUSH	XOR	その他
INT	17	23	36
LEA	3	1	0
MOV	118	30	19
POP	15	14	10
PUSH	286	6	10
XOR	55	20	21
その他	27	13	63

上記の 2 つの表で示した通り, 前述の攻撃データ 50 個の中で出現しているアセンブラ命令のペアにおいては, PUSH 命令同士のペアが 286 個見つかり, その一方で MOV 命令→POP 命令のペアが 1 個しか見つからなかったことから, 出現頻度に大きな差が生じていることがわかった. その一方で, 攻撃データ 50 個の中で 1 個も出現しなかったアセンブラ命令のペアも 7 組存在していた.

5. 考察

前節の表 1 と表 2 から読み取れることとして, PUSH 命令同士のペアが 300 個弱見つかっており, 2 つの表の中で突出して多くなっているということが第一に挙げられる. 実際に 50 個の攻撃データを解析していくところで, PUSH 命令同士のペアが 1 つの攻撃データ内において多数存在しており, 約半数が PUSH 命令同士のペアであった攻撃データも存在していた. スタックにおけるバッファオーバーフローのように, すぐには処理しきれないほどの大量のデータをスタックに積むことによってスタックの中を溢れさせ, これによりプログラムに意図しない動作を実行させるために, PUSH 命令同士のペアが大量に存在しているものと考えられる.

また, 50 個の攻撃データの中で 1 個も出現しなかったアセンブラ命令のペアが 7 組存在していることも表 1 と表 2 から読み取れるが, これらの 7 組のペアが攻撃データ中に存在しないことについても何らかの特徴が存在する可能性があると考えられる.

そして, 第 4 節で挙げた 6 つのアセンブラ命令以外のアセンブラ命令が含まれるペアについて考えてみると, LEA 命令が関係しているアセンブラ命令のペアよりも出現頻度が多くなっている傾向を表 1 と表 2 から読み取ることができる. 着目対象とするアセンブラ命令を追加および変更することによって, バッファオーバーフロー攻撃に用いられるシェルコードの特徴を現状よりも的確に見出せるのではないかと考えられる.

6. まとめと今後の課題

以上より, 解析対象とした 50 個の攻撃データ中に存在するアセンブラ命令のペアとして, PUSH 命令同士が多数を占めていることがわかった. その一方で, 1 個しか存在していないアセンブラ命令のペアや全く存在していないアセンブラ命令のペアも確認でき, バッファオーバーフロー攻撃特有のシェルコードの書き方が特徴として表れてくるのではないかと考えられる.

今後の課題として, 正常データにおいても第 4 節のようにアセンブラ命令ペアの出現頻度を解析し, 攻撃データと正常データのぞれぞれの解析結果を比較することで, バッファオーバーフロー攻撃に用いられるシェルコードの特徴を見出すこと, 次に特徴を見出してから未知のデータが与えられた際に, そのデータが攻撃データか正常データかを正しく判定できるかを考察すること, そしてアセンブラ命令ペアの出現頻度の解析において着目するアセンブラ命令の追加および変更を検討すること, が挙げられる.

参考文献

[1]北條 孝佳, 佐久間 英夫, 種茂 文之, “シェルコード解析による不正アクセス検出手法”, 情報処理学会研究報告 2003-CSEC-23, 2003
[2]Jon Erickson 著, 村上 雅章 訳, “Hacking 美しき策謀 脆弱性攻撃の理論と実際 第 2 版”, オライリー・ジャパン, 2011
[3]WEB マーケティング研究会, “トラブル事例に学ぶ Web サイトのセキュリティ 第 5 回・バッファオーバーフロー攻撃”, http://www.webdbm.jp/column2009/column2009-03/2330/, 2015 年 6 月 24 日閲覧
[4]インテル株式会社, Intel Corporation, “IA-32 インテル アーキテクチャ・ソフトウェア・デベロッパーズ・マニュアル”, 2004