L-010

バイト値列の類似性に基づく亜種ウィルス検出
Detecting Subspecific Virus based on Similarity of Byte-Sequences

上園 智士[1], 小林 和朝 [2] , 高田 寛之 [2]
Akihito Kamizono, Kazutomo Kobayashi, Hiroyuki Takada

1 はじめに

インターネットで蔓延するコンピュータウィルスやワームの被害を抑制するためには高度な検出手法が必要となる。これまで類似性に基づいたワームの検出手法として、ワームフローのペイロードに共通な文字列の集合をシグネチャとして用いる手法 [1] や、フローペイロード中のコードの出現確率を用いた検出手法等 [2][3] が提案されている。しかし今回我々は特に亜種ウィルスに注目してその検出手法を提案する。その理由はセキュリティベンダーの米サンベルトソフトウエアは 2008 年 1 月 24 日に 2007 年に出現したウィルスは過去最高の 549 万 960 種類だと発表し、この原因をサンベルトソフトウェアではオリジナルのウイルスをわずかに変えた亜種（変種）が多数作られるようになっているためと分析した [5]。549 万 960 種の中にはこの亜種ウィルスが多数を占めていると言われていて、現代亜種ウィルスの存在がとても脅威であるからである。本研究では、亜種ウィルスは元々あるウィルスを改変して作るウィルスなので、同じシグニチャをもつことが予測され、2 つのウィルスを比較し、共通のシグニチャを見つけ出し、それがウィルス全体のどれくらいの割合を占めるかを、そのウィルス同士の類似度とし亜種を検出する。検出を行う際に時間短縮のためにハッシュマップを用いて検出を行う。

2 バイト値列の類似性を用いた検出手法

ウィルスには特有のシグニチャをもつと可能性が高いという観点から、バイト値列の類似性に基づく亜種のウィルス検出について提唱する。今回の手法では 2 つのプログラムが全体のうちどの程度共通するシグニチャを含んでいるかをもとに類似度を計り亜種ウィルスを検出する。ウィルスが特有のシグニチャをもつのであれば、そのシグニチャはある程度の長さをもつはずである。よってファイル同士の共通のシグニチャを見つけ出すためにリファレンスとなるファイルを 150 バイトごとにマッピングしてハッシュマップをつくる。ここでは簡単のためにファイル α をリファレンスとし、3 バイトごとにマッピングした例を以下に示す。

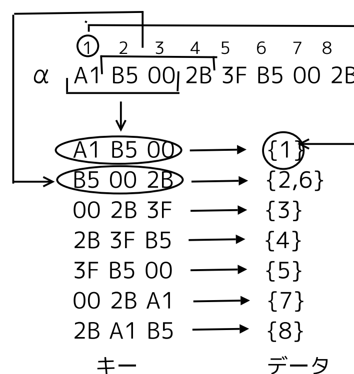

図 1: α を 3 バイトごとにマッピングしたハッシュマップ

[1]長崎大学大学院生産科学研究科 電気情報工学専攻
[2]長崎大学工学部情報システム工学科

まず図 1 のようなハッシュマップを作る。ハッシュマップはプログラム α の 1 バイト目から 3 バイトずつをキーに渡してデータの部分にはそのキーが α の何バイト目にあたるかを格納する。3 バイトとも同じキーの場合はデータを後ろに追加する。実際のプログラムでは 3 バイトではなく 150 バイトごとにマッピングしていく。

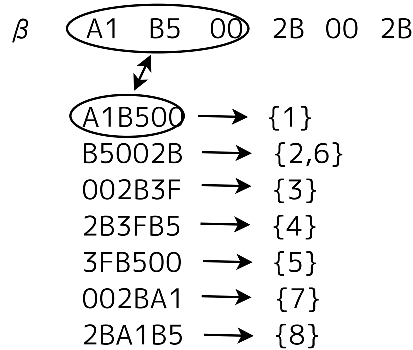

図 2:検査したいファイル β をハッシュマップにかけキーを取得する

次に検査したいファイル β を 3 バイトごとにハッシュにかけてデータを取得する。図 2 では A1B500 からデータ 1 を取得できる。

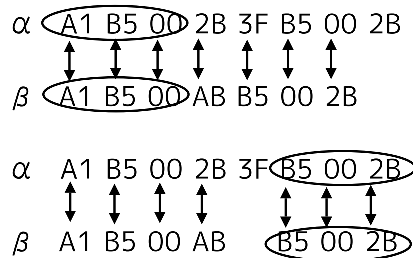

図 3:ハッシュマップからデータを取得し 2 つのファイルのバイト値列を比較

データを取得すれば α と β の 3 バイトごとの一致箇所を特定できるので図 3 のようにその位置からバイトを比較していく。なお、一致箇所を特定し α と β を比較していく際に初めて α と β で異なったバイト値がでてくる場所で比較を止め、その位置を記憶しておく。詳しくは図 4 に示す。

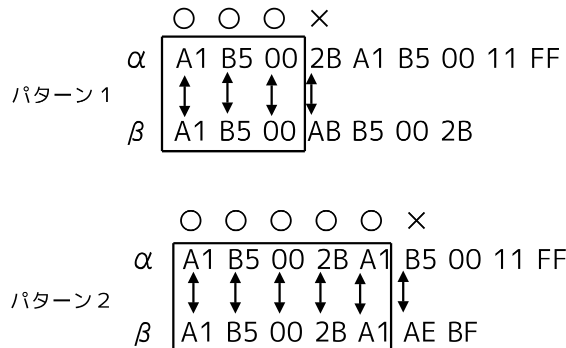

図 4:最初に異なったバイト値がでてくる位置の特定

図 4 のパターン 1 の場合、ハッシュマップから A1B500 の一致を見つけだし 1 バイトずつ比較していった際に 4 バイト目で初めて異なったバイト値がでてくる。同様にパターンにでは 6 バイト目に初めて異なったバイト値がでてくる。これより図 5 の四角で囲んだ部分は一致していると断定できるので、次に β をハッシュマップにかける際は一致部分の次のバイトからでよい。つまりパターン 1 では 4 バイト目、パターン 2 では 6 バイト目からの 3 バイトをハッシュにかけていき、これを繰り返す。実際のプログラムでは 3 バイトではなく 150 バイトごとにマッピングしていく。但し、マッピングしていく際に 150 バイトの内 135 バイト以上が 00 の場合はマッピングしない。これはバイナリコードで書かれたファイルは 00 を非常に多く含んでいるため、誤検出と実行時間の削減のためである。

本論文では 2 つのプログラムのバイト値列の類似性指標として、次の δ を考える。

$$\delta = \sum_{i=1}^{n}(g_i \div d_1 \times 100) \tag{1}$$

n は β をハッシュマップにかけたときにハッシュマップからデータを取ってこれた回数 (つまり 150 バイトごとの一致を見つけることができた回数)、g_i はそのときの始めの 150 バイトを含めバイト値列が連続して完全に一致している数 (図 4 の〇の部分) である。d_1 は 2 つのプログラムでファイルサイズが小さい方のファイルのサイズとする。δ はファイルサイズの小さい方のファイルに対し、150 バイト以上のバイト値が連続して一致している部分が、全体のどの程度の割合を占めているかを表したものである。

3 提案手法の評価

今回の実験にはインターネット上のサイトや P2P ソフト Winny から取得したウィルスを使用する。また False Positive を検査する際には Linux の bin や usrbin の中にあるバイナリファイル (225 個) や Windows の実行ファイル (322 個) を使用した。今回既知のウィルス (89 個) と検出対象を比較し、その δ が閾値 t を越えた場合に検出対象を亜種ウィルスであると判定した。なお閾値は文献 [4] で亜種だと保証されているものを検出できるぎりぎりの値 0.72%とした。

表 1: 評価

Virus	TPR(%)	FPR(%)	WPN(個数)	LPN(個数)	NSVPN(個数)
Backdoor.Agobot.gen	100	0	0	0	9
Backdoor.Agobot.ok	100	0.18	1	0	0
Backdoor.Agobot.pp	100	0	0	0	0
Backdoor.Agobot.sy	100	0.18	1	0	1
Backdoor.Gobot.p	100	0.36	2	0	8
WORM_ANTINY.F	100	0	0	0	0
Win32.Pinom.c	100	0	0	0	25
Worm.Win32.Dedler.g	100	0.18	1	0	0

実験から表 1 のような結果が得られた。TPR は亜種ウィルスの検出率で、FPR は WPN と LPN の個数をたして全体のファイル数 (547 個) で割った誤検出率、WPN は Windows の実行ファイルにおいての誤検出を生んだファイルの個数、LPN は Linux の実行ファイルにおいての誤検出を生んだファイルの個数、NSVPN は亜種とは保証されていないウィルスを亜種だと検出したファイルの個数である。今回閾値は亜種だと保証されているものは検出できるぎりぎりの値を使用したので TPR は 100%となる。今回誤検出にひっかかったファイルは全て Windows のファイルで、システム情報を知るためのプログラムやディレクトリを作るプログラムであった。NSVPN が示すように亜種だと保証されていないウィルスに関しても検出に多くひっかかることがわかった。

4 まとめ

Windows の実行ファイルでひっかかたファイルはシステム情報を知るためのプログラムやディレクトリを作るプログラムであったため、ウィルスが感染活動を行う際にこれらのプログラムと似た動きをするのではないかと推測できる。亜種だと保証されていないウィルスも検出にひっかかるウィルスがあったことは、ウィルスが亜種でなくても共通のシグニチャを持つことを示し、この検出手法は亜種ウィルス検出に有効であるだけでなく、新種のウィルスも検出できる可能性があることを示唆している。

参考文献

[1] S. Singth, C. Estan, G. Varghese and S. Savage, “Automated worm fingerprointing”, In Proceedings of the 6th ACM/USENIX Symposium on Operating System Design and Implementation (OSDI), Dec, 2004
[2] 鈴木洋平, 和泉勇治, 角田裕, 根元義章: パケットペイロードの類似性に基づくワーム検出に関する一検討, 電子情報通信学会通信ソサイエティ大会, B-7-25, 2007.
[3] 鈴木洋平, 和泉勇治, 角田裕, 根元義章:フローペイロードのクラスタリングを用いた低負荷なワーム検出方式, CS2006-33, pp.67-72, 2006
[4] トレンドマイクロ: ウイルスデータベース, http://www.trendmicro.co.jp/vinfo/virusencyclo/default.asp
[5] ITpro: セキュリティニュース, http://itpro.nikkeibp.co.jp/article/NEWS/20080125/291973/