1Z-7

多段階マッチング回路を用いたウイルス検出エンジンの提案

†安田 圭佑　†小柳 滋
†立命館大学情報理工学部

1　はじめに

近年, マルウェアは増加傾向にある.McAfee 脅威レポート [1] によると, 昨年に比べてマルウェアは増加しており, 今後も増加すると考えられる. マルウェアは感染経路が多種多様であり, 電子メール, ウェブサイト, 記憶媒体などから感染する. マルウェアに感染された計算機はデータの改竄・破壊, 不正な遠隔操作, 情報の盗難が行われており社会問題となっている. 既存の対策方法としてウイルス対策ソフトの導入が挙げられ, ウェブブラウザや電子メールクライアントなどで送受信されるデータを動的にスキャンすることで感染を防ぐことができる. しかし, ネットワークや USB などの高速化に伴い, ソフトウェアでの処理は困難である. このため, ハードウェアを用いた高速なウイルス検出エンジンが必要である.

本研究では,USB メモリへの応用を想定し, 比較回路を複数用いることで多段階マッチングを行い, スループットが高く, 回路規模を抑えたウイルス検出エンジンの設計を目標としている. Verilog HDL を用いて実装し, 回路の周波数, 回路規模について評価する.

2　研究背景

ハードウェアを用いた高速なウイルス検出エンジンに関する既存研究として, 中原らの研究 [2] がある. この研究では,ClamAV のハードウェアアクセラレータを提案しており, 文字列の一部をハードウェア, 残りの文字列をソフトウェアを用いてパターンマッチングを行うことで, 高スループット, 低消費電力・安価, ウィルスパターンの更新可能なウイルス検出エンジンを実現している.

ウイスルの検出方法として, ウイルスのシグネチャとパターンマッチングを行う. ハードウェアを用いたパターンマッチングにはさまざまな方法が存在しており,CAM,Bloom Filter などが存在する. 連想メモリは, ネットワーク機器にも使われており高速な文字列検索が可能であるが, 回路規模が膨大になる欠点がある. このため, ウイルス検出エンジンには Bloom Filter を用いる.

Bloom Filter は, 空間効率の良い情報検索手法であり, ハードウェアで実現することで回路規模を抑え, 高速に文字列を検索することが可能である. しかし,Bloom Filter には偽陽性による誤検出が発生することがあり,

Proposal of a virus scanning engine using multi-stage pattern matching
†Keisuke YASUDA　†Shigeru OYANAGI
†College of Information Science and Engennering Ritsumeikan University

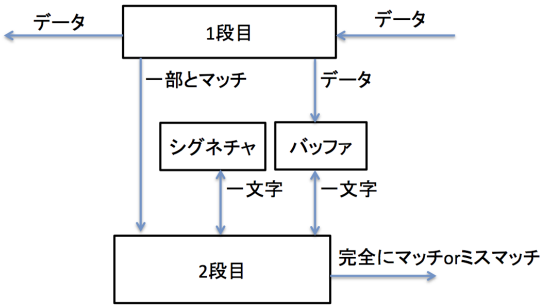

図 1: ウイルス検出エンジン

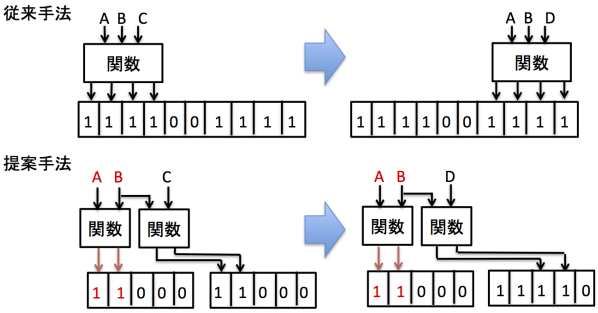

図 2: 多段階マッチング

また 3 以上のハッシュ関数を用いてビット配列に同時アクセスすることを考えると FPGA 上の Block RAM は使用できず, レジスタを用いることになり回路規模が膨大になる問題がある.

本研究では,2 段階マッチングというヒントを元に多段階マッチングを行う回路の作成を行い, スループットが高く, 回路規模を抑えたウイルス検出エンジンの作成を行う.

3　提案手法

3.1　概要

ウイルス検出エンジンでは, 図 1 のように 2 段階に分けてパターンマッチングを行う.1 段目ではパターンの一部と比較を行い,2 段目では 1 段目でウイルスの可能性が高いと判断したものに対し, パターンと完全に一致するか調べることでウイルスの検出を行う.

提案手法では,1 段目に Bloom Filter を複数用いた多段階マッチング回路の作成を行う. これにより, 偽陽性の発生を減らし, 回路規模を抑えることができると考えられる. 例として,ABC と ABD と言う文字列を Bloom

表 1: Bloom Filter 誤検出数

	理論値	実験値
4word(Bloom Filter × 3)	11488	14320
6word(Bloom Filter × 5)	126	150
8word(Bloom Filter × 7)	1	1
10word(Bloom Filter × 9)	0	0

表 2: 周波数

提案手法	283.286MHz
Bloom Filter	237.417MHz
CAM	162.840MHz
CAM エミュレータ	270.929MHz

Filter へ登録するときを考える. 図 2 から分かるように従来手法に比べ提案手法ではビット配列の 1 の数が少なくなり, 偽陽性を抑えることができる. また, ビット配列へは 2 アクセスであり,FPGA 上の 2 ポート Block RAM を使用することで回路規模を抑えることができる. 実装では,1 段目の Bloom Filter でウイルスだと判定したとき,2 段目の Bloom Filter を動かすようにしている. これにより,n 段目でウイルスでないと判定したとき,n+1 段目以降の Bloom Filter を動かす必要がなくなり, 消費電力を抑えることが可能である.

4 実験と評価

4.1 実験内容

ウイルスと仮定したコードを埋め込んだ 10MB のデータに対し,4 から 10 文字のパターンと比較を行う提案手法,Bloom Filter,CAM,CAM エミュレータを用意しスキャンを行った.

実験 1 では,(1) 式を用いて求めた Bloom FIlter の誤検出の推定値と実験値の比較を行う. また, 実験値から 2 段目が対応できるかについて評価を行う.

$$\text{誤検出の理論値} = \left(1 - \left(1 - \frac{1}{m}\right)^{kn}\right)^{ks} \times size \qquad (1)$$

[m:ビット配列の大きさ,k:ハッシュ関数の数,s:Bloom Filter の数,size:データの大きさ]

実験 2 では, 上記の 4 種類のパターンマッチング回路について周波数, 回路規模について評価を行う.

4.2 評価

実験 1 の結果を表 1 に示す. 実験では理論値と実験値がほとんど同じ値となり, 誤検出率は約 0.001 となった. そのため,1 クロックで 1 文字の比較を行い, 最大 100 文字のパターンと比較を行う 2 段目でも対応できる.

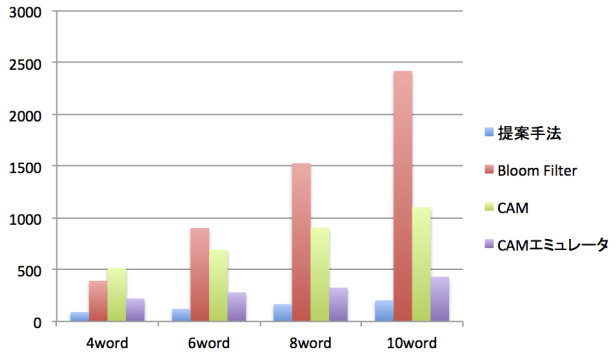

図 3: 回路規模

実験 2 の結果を表 2 と図 3 に示す. 周波数は提案手法が一番早くなっており, また回路規模は他の手法に比べて明らかに小さくなるという結果となった.

これら実験 1, 実験 2 の結果より, 本研究の目標であるスループットが高く, 回路規模の抑えたウイルス検出エンジンの設計を達成したといえる.

5 まとめ

本研究は,USB メモリへの応用を想定しおり, 回路規模を抑えた設計を目指した. 提案手法を用いた, ウイルス検出エンジンでは,USB2.0 規格に十分対応することができるが USB3.0 規格には対応できていないため今後の課題となっている. また,Bloom Filter を用いているため, 誤検出は避ける事ができないが, 誤検出率が充分低いため全パターンマッチングを行う回路またはソフトウェアを用いることで誤検出を取り除くことが可能である.

参考文献

[1] マカフィー株式会社：McAfee 脅威レポート：2013 年第 3 四半期, 入手先 <http://b2b-download.mcafee.com/products/japan/pdf/threatreport/threatreport13q3.pdf> (参照 2013-12-20)

[2] 中原 啓貴, 笹尾 勤, 松浦 宗寛 : 4IGU エミュレータと MPU を用いたウイルス検出エンジンについて, 電子情報通信学会技術研究報告, Vol. 111, No 31, pp. 55-60 (2011.5)

[3] 今泉貴史, 水野恵祐 :IDS に特化した文字列探索アルゴリズム, 電子情報通信学会技術研究報告,Vol.108,No460,pp.107-112 (2009.3)