Vol.2017-CSEC-77 No.12
Vol.2017-IOT-37 No.12
2017/5/26

DNS クエリの分析によるサーバと非サーバの識別

渡部 耕大[1]　佐藤 聡[2,3]　新城 靖[3]

概要：近年、DNS の通信を対象としたデータマイニングの研究が行われ、それによって有益な情報を得る研究が行われている。その中には DNS を利用してサイバー攻撃を試みるマルウェアに対して DNS クエリを解析し検知を試みるものがある。そこで本研究では DNS 問い合わせからサーバと非サーバの識別を機械学習を用いて行うことが可能であるかを確認することを目的として、調査を行った。調査には筑波大学に設置されているフルサービスリゾルバへの DNS 問い合わせのログを使用し、WEKA を用いた 13 種類の学習アルゴリズム毎の精度の結果の比較と C5.0 を用いた教師データのデータ数の変化による精度の変化の調査を行った。分析の結果、本研究での識別において不向きなアルゴリズムはいくつかあるが、突出して精度の高いアルゴリズムは存在しなかった。また、調査に用いるクエリのデータを 1 時間分から 2 ヶ月分まで大きくしていくと識別の精度が上がることがわかった。

キーワード：DNS, 機械学習, WEKA, C5.0, サーバ, 識別

Watanabe Kodai[1]　Sato Akira[2,3]　Shinjo Yasushi[3]

1. はじめに

DNS(Domain Name System) はドメイン名を IP アドレスに変換するサービスで、ウェブブラウジングやメール転送等の様々な場面で使われ、現在のネット社会において必要不可欠なものとなっている。DNS は様々なレコードタイプを保持可能であり、その用途に合わせたレコードタイプを指定して検索される。このように DNS の重要性が増すに連れ、近年、DNS の通信を対象としたデータマイニングの研究を行い、それによって有益な情報を得る研究が行われている。例えばその中には DNS を利用してサイバー攻撃を試みるマルウェアに対して DNS クエリを解析し検知を試みるものがある。マルウェアによる DNS の利用法にもいくつかあり、最近発見された手法として、検索結果の中身を自由に記述できる TXT レコードを利用したマルウェアの遠隔操作手法 [13] がある。これはマルウェアへの指令を保存するための DNS サーバを用意しその中にマルウェアへの指令を記載した TXT レコードを保存することで、マルウェアが定期的にこの TXT レコードを検索し記載されている指令を実行するという手法となっている。また、DNS の通信データからユーザトラッキングを行うものや回線を圧迫するヘビーユーザの識別を行うものもある。このように DNS の通信を対象としたデータマイニングの研究が数多く行われている。

そこで本研究では、DNS 問い合わせを機械学習による分析を行うことで、ホストをサーバとサーバ以外に分類することが出来るかどうかの確認を目的とした。これによりマルウェアの検知や組織内の LAN 環境の構成、または機器の設定ミスを知るのに役立つ可能性があると考えられる。分析に用いる DNS 問い合わせのデータは筑波大学に設置されているフルサービスリゾルバのログデータを使用し、各ホストからフルサービスリゾルバへの問い合わせデータのみを使用する。このログデータからホストごとにクエリ数、各レコードタイプの割合、そして通信時間という 3 種類の属性を分析する。このように本研究ではプライバシー等の観点から、問い合わせ内容を調査せず、統計データからホスト毎の識別を行うことが特徴としてあげられる。それとは別にフルサービスリゾルバへの問い合わせを行っているホストに対して、サーバであるかどうかの調査をし正解リストを用意した。この 2 つのデータを教師データセッ

[1] 筑波大学 システム情報工学研究科 コンピュータサイエンス専攻
Department of Computer Science, University of Tsukuba Graduate School of Systems and Imformation Engineering
[2] 筑波大学 学術情報メディアセンター
Academic Computing and Communications Center, University of Tsukuba
[3] 筑波大学 システム情報系
Faculty of Engineering, Information and Systems

トとして WEKA[9] を用いた 13 種類の学習アルゴリズム毎による結果の比較、及び C5.0[8] によるデータ数の違いによる識別精度の検証を行った。学習アルゴリズム毎による比較の結果では C5.0 等のいくつかのアルゴリズムで同程度の精度での識別を行うことが出来ることが分かった。そして C5.0 によるデータ数での精度の比較では、調査に用いるクエリのデータを 1 時間分から 2 ヶ月分まで大きくしていくと識別の精度が上がり、1 週間分程度のデータ数から比較的高い精度での識別が行えることがわかった。

2. 既存研究

武蔵らの研究 [18] では、熊本大学における大量の DNS クエリパケットのうち主に A レコードタイプのクエリを調査した。その結果大量メール送信型ワームの活動やスパムメール送信活動での A レコードタイプクエリのコンテンツにはいくつかの特定のキーワードが含まれているということが発見された。またその他のマルウェアに感染している端末からの A レコードタイプクエリのコンテンツには検索ドメインに IP アドレスが直接記載されているということが発見された。これらの結果から A レコードタイプ DNS クエリのコンテンツを調査することで大量メール送信型ワームやその他のマルウェアの検知が可能であるということが判明した。

Dietrich らの研究 [1] では、マルウェアサンプルの通信データのうち、主に TXT レコードタイプのコンテンツを k-means クラスタリングとユークリッド距離によるクラスタリングにより分類を行った。そしてマルウェアによる利用法やその検知のメカニズムを示している。

津田らの研究 [17] では機械学習を用いた DNS 通信の分析によるマルウェア検知手法を提案している。DNS 応答パケットを RandomForest を用いた分析を行い、津田らの論文内で記述されている他の手法と変わらない検知率と 1 時間に 1 度の検知速度を実現した。

石橋らの研究 [16] では、DNS サーバを圧迫する少数のヘビーユーザを識別するため、エントロピーを用いて正常なユーザとの差別化を行っている。

Dominik らの研究 [3] ではクッキー等の技術を用いず、DNS クエリを分析することによって動的 IP アドレスを持つユーザのトラッキングを行っている。

これらの研究に対する本研究の大きな特徴としては使用するデータにある。これらの研究ではクエリに対するレスポンスや問い合わせ内容とその答えなどの調査も行っているが、本研究ではそれらの調査は行わず複数のレコードタイプに対するクエリ量や通信の間隔を調査することによってプライバシーにも配慮している。

3. 識別概要

本章では識別の流れとそれに伴って必要となるデータの作成、及び評価方法について述べる。

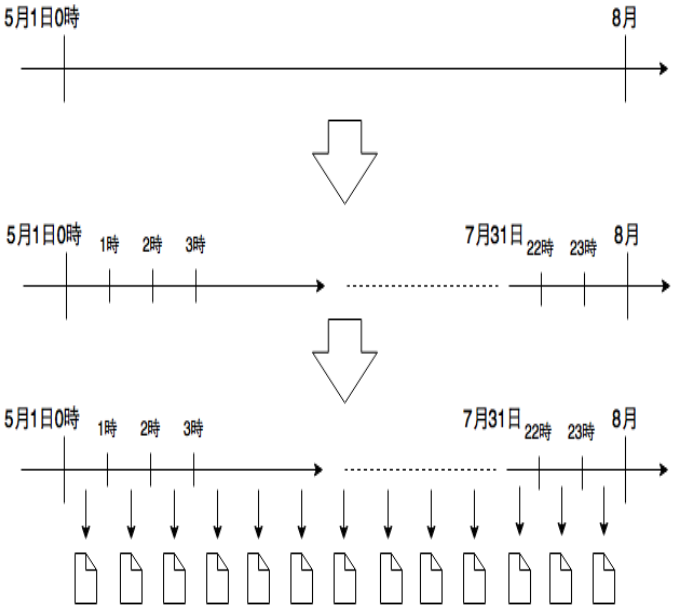

図 1 クエリデータのデータ間隔

3.1 識別の流れ

本研究では DNS 通信からサーバを識別することを目的とした教師あり機械学習による識別の実験を行い、その精度等についての考察を行う。

教師あり機械学習アルゴリズムを用いるため、数値等のデータセットとそのデータセットがどのクラスであるかという正解リストが必要となる。また、作成したルールを検証するためのデータが必要となる。そこで本研究では以下の 2 つのデータを作成する。

- 各ホストの DNS クエリの統計データ
- サーバであるホストのリスト

これらのデータを用いて、教師あり機械学習アルゴリズムの比較実験と教師データのデータ数による比較実験の 2 種類を行う。教師あり学習アルゴリズムの比較実験では、教師データやテストデータ等の条件を同じにした時の機械学習アルゴリズムごとの精度の比較を行う。これは WEKA を用いて行う。また教師データのデータ数による比較実験では、教師データに用いるクエリの統計データのデータ期間を増減させることでの精度の比較を行う。

3.2 クエリの統計データ

扱う DNS クエリのデータは、筑波大学に設置されている DNS フルサービスリゾルバのうちの 1 台で保存されているログデータのうち、2016/05/01〜2016/07/31 の期間のものから調査を行う。筑波大学内 LAN 環境の一部では DHCP が動作しているため正確なホスト数ではないが、調査期間では 18,214 のホストが計 2,571,014,808 のクエリを DNS フルサービスリゾルバに送信していた。また本研究

では各ホストからフルサービスリゾルバへの DNS クエリのみの集計を行い、再帰問い合わせやレスポンスの調査は行わない。このログデータには各ホストからフルサービスリゾルバへ問い合わせが 1 件ずつ記載されている。本研究では、このクエリの連続データをウインドウ毎に区切ってメタデータを抽出し、そのウインドウ毎に識別を行う。

データの識別を行う最終的な目的はマルウェアの検知である。したがって、このウインドウはなるべく短い時間である必要がある。またウインドウを短くしすぎると、データ数が少なく識別自体が困難になってしまうおそれがある。これらの理由から本研究でのウインドウの間隔は 1 時間とした。

1 時間毎に、ログデータから以下の統計データに変換する処理をする。

- ホスト毎の総クエリ数
- ホスト毎の総クエリ数に対する各レコードタイプの割合
- ホスト毎の通信時間の割合

用いるデータにはプライバシーの観点から、上記以外の検索ドメイン名等の情報は本研究では使用しない。IP アドレスは各ホストの区別とサーバリストの作成のみで扱う。

レコードタイプは主に使用されることの多いものを対象として A, AAAA, ANY, CNAME, MX, NS, PTR, SOA, SRV, SSHFP, TXT に分類する。これに当てはまらないものは全て unknown とする。

通信時間の割合とは**図 2** に示す例のように、1 時間を 1 分ごとの 60 のの区間に分け、そのうち通信を行っている区間の割合を計測する。計測区間を 1 分ごととしたのは、1 分ごとの 60 の区間に分割して計測すると、区間 1 つにつき $\frac{1}{60} = 1.6\%$ の差が出るため識別を行うのに十分な差だと考え、本研究では区間を 1 分ごとにして計測する。今後、本研究ではこの値を**通信時間の割合**と表記する。**図 3** は第 3.3 節で作成するサーバリストによって 2016/5/1 の 0 時から 1 時間分のクエリの統計データをサーバとそれ以外に分けた時の通信時間の割合による度数グラフを示した図である。図 3 より、サーバに関しては多くのホストにおいてこの数値が高いと言える。このように、稼働率の高いサーバに関してはこの数値が高いと考えられるため本研究において重要な数値になると言える。

3.2.1 クエリ数の下限値

サーバの中にはほとんど稼働せず DNS をクエリを送らないサーバがあることが考えられるが、クエリ数があまりにも少ないホストではデータ数が少なくなってしまい識別が困難だと考える。そこで全ホストのクエリ数を降順にした時、総クエリ数の 80%を占める上位のホストのみを調査対象とし、クエリ数の下限値を設定した。調査の結果 1 時間に 60 件以上のものを本研究での調査対象とし、以下での実験では全てこの条件に当てはまるホストのみを扱う。

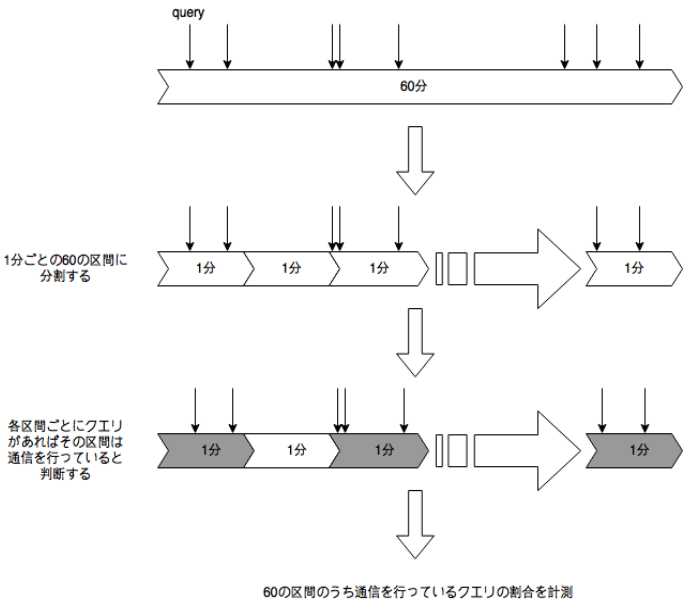

図 2 通信時間の割合の例

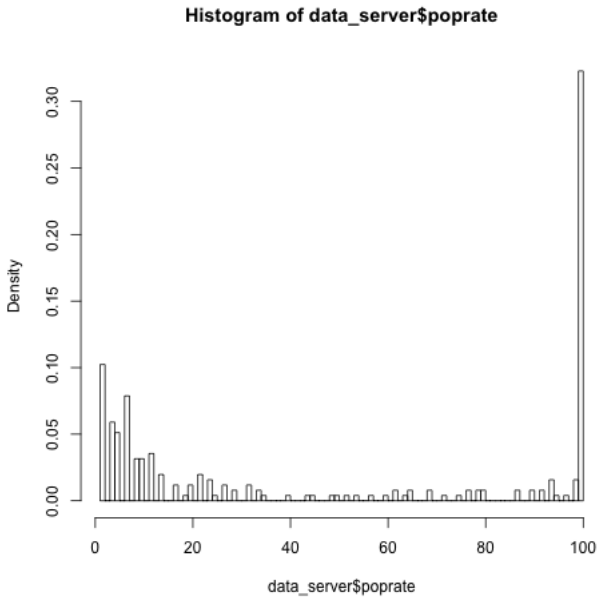

図 3 通信時間の割合によるサーバのホストの度数グラフ

調査期間のホストが延べ 9,906,791 ホストある中、条件に当てはまるホストは延べ 5,341,030 ホストとなっている。

3.3 サーバリスト

本研究では、学内に特に多く存在するであろうサーバとして、以下の 3 つのサーバのいずれかとして動作しているものをサーバとして調査を行う。

- メールサーバ
- ウェブサーバ
- SSH サーバ

各ホストがサーバであるかどうかのリスト作成方法として、調査対象となるプロトコルの標準ポートが開放されて

表 1 調査対象のポート番号と動作プロトコル

ポート番号	プロトコル	サーバ種
22	SSH	SSH サーバ
25	SMTP	メールサーバ
80	HTTP	ウェブサーバ
109	POP2	メールサーバ
110	POP3	メールサーバ
143	IMAP	メールサーバ
220	IMAP3	メールサーバ
443	HTTPS	ウェブサーバ
465	SMTPS	メールサーバ
587	サブミッションポート	メールサーバ
993	IMAPS	メールサーバ
995	POP3S	メールサーバ

```
#tcp の特定ポートに接続できるリストを作成
#実行例:$ruby port_check.rb 80 ip_list.txt

require "net/ping"

port=ARGV[0].to_i #ポート番号
ip_list_file=ARGV[1] #IP アドレスリスト

#IP アドレスリストを開く
File.open(ip_list_file, "r") do |file|
  file.each_line do |line|
    ip = line.chomp
    #ポートのチェック
    if Net::Ping::TCP.new(ip, port).ping?
      print ip,"\n"
    end
  end
end
```

図 4 ポートチェックのプログラム

いるかを調査する。調査対象として確認するポート番号は**表 1** に示す。

本研究では 2016/05/01〜2016/07/31 までの期間に筑波大学のフルサービスリゾルバへ DNS 問い合わせを行った送信元 IP アドレスのリストを作成した。これを調査対象の IP アドレスとした。次に、本研究では調査対象の IP アドレスに対してポートが開いているかどうかの調査を行った。この調査を 2016/11/16 に行った。上記に関して実装したものを以下に示す。このプログラムは Ruby の net-ping ライブラリを用いて実装している。実行の際に調査対象の IP アドレスを 1 行ごとに記述したリストのファイルを用意する。この IP アドレスリストを ip_list.txt、80 番のポートチェックを行うとすると**図 4** の 2 行目の実行例のように実行することで、読み込んだ IP アドレスのうち 80 番のポートがあいている IP アドレスのみが出力される。これを先程述べた表 1 の全てのポートに対して実行する。

3.4 評価方法

サーバリストに登録されているホストのうちサーバと識別された割合を感度、サーバリストに登録されていないホストのうちサーバでないと識別された割合を特異度とし、この 2 つの指標から精度評価を行う。

4. 様々な学習アルゴリズムによる判定の結果

本章では WEKA を用いた様々な学習アルゴリズムによる判定の比較実験とその結果について述べる。WEKA は Waikato 大学によって開発されたフリーソフトウェアでいくつかの機械学習アルゴリズムによる機械学習とその視覚化が行える。インストールや使用法に関しては [9] を参考にした。

また、C5.0 は RuleQuest Research[8] によって開発された教師あり機械学習アルゴリズムであり計算が高速であり使用法が容易である。次章のデータ期間による比較を行う際に大量のデータからルール作成を行う必要があるため本研究では C5.0 での比較実験を行った。インストールや使用法に関しては [8] を参考にした。

4.1 比較方法

アルゴリズムによる識別精度の比較を行うため、教師データとテストデータは全てのアルゴリズムで同様のものを用いる。2016/05/01〜2016/05/07 の 1 時間毎の統計データを 1 つの教師データとして作成する。この時、複数の時間帯に同じホストのデータがあった場合には別データとして扱う。この精度の評価には 2016/7/1〜2016/7/31 のクエリ統計データを 1 時間毎に作成したものをテストデータとして感度と特異度を計測し、1 ヶ月分、すなわち 31×24 個のテストデータに対する感度と特異度の平均値を出した。

比較する 13 種類のアルゴリズムを以下に示す。なお、比較するアルゴリズムとしては WEKA で使用できる決定木作成のアルゴリズムとルール作成のアルゴリズムを全て採択した。C5.0 以外の 12 種類のアルゴリズムが WEKA で使用出来るアルゴリズムである。C5.0 に関しては WEKA では選択出来ないため、WEKA の機能とは別に実験した。

- J48
- DecisionStump
- HoeffdingTree
- LMT
- RandomForest
- RandomTree
- REPTree
- DecisionTable
- JRip
- OneR
- PART

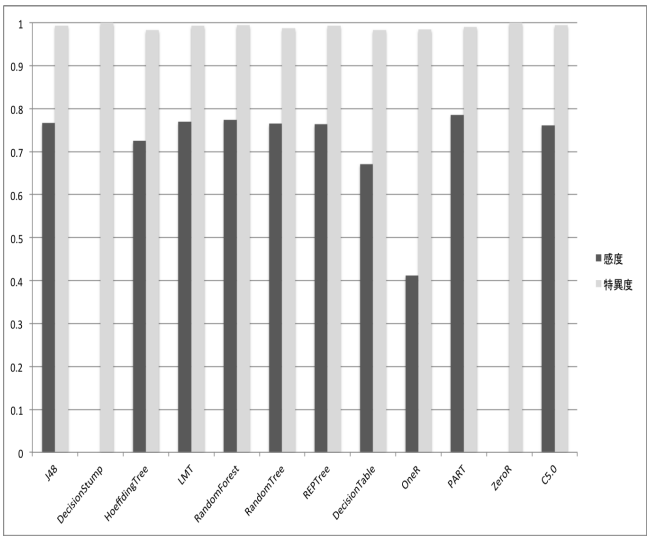

図 5 アルゴリズムによる比較の結果

- ZeroR
- C5.0

なお、実行時のメモリは最大 2GB 確保している。

4.2 結果

結果を**図 5** に示す。JRip はメモリ不足が原因で実行できなかった。また DecisionStump と ZeroR を用いた時はサーバのルールが作成できなかったため、すべて“サーバでない”と識別されてしまい感度が 0%というようになっている。DecisionTable と OneR は他のアルゴリズムと比べて感度と特異度が共にが少し低くなっていると言える。また HoeffdingTree, RandomForest, RandomTree は他のアルゴリズムとくらべて特異度のみが少し低くなっていると言える。それ以外の J48, LMT, REPTree, PART, C5.0 に関しては感度が 0.760 以上、特異度が 0.990 以上の精度で識別を行えていることがわかる。

13 種類のうち 5 種類のアルゴリズムでほぼ同程度かつ一番高い精度の数値が出ている。本研究の手法と今回使用したデータの場合では密度と特異度の上限は 0.786 と 0.994 となった。

5. データ期間による比較

本章では教師データに用いるデータの期間を変化させた時の比較実験とその結果について述べる。

5.1 比較方法

教師データに用いるデータの期間による識別精度の比較を行うため、複数の教師データに対して特定のアルゴリズムを用いて比較実験を行う。教師あり機械学習アルゴリズムとしては C5.0 を用いた。

表 2 に、教師データに用いるデータの期間とその教師データから C5.0 で作成されたルール数を示す。この精度の評価には 2016/7/1〜2016/7/31 のクエリ統計データを 1 時間毎に作成したものをテストデータとして感度と特異度を計測し、1 ヶ月分、すなわち 31 × 24 個のテストデータに対する感度と特異度の平均値を出した。

表 2 ルール作成に用いたデータの期間とその時のルール数

ルール作成期間	ルール数
2016/05/1 0:00〜0:59:59.99999 (1 時間)	4
2016/05/1 0:00〜1:59:59.99999 (2 時間)	15
2016/05/1 0:00〜3:59:59.99999 (4 時間)	17
2016/05/1 0:00〜7:59:59.99999 (8 時間)	28
2016/05/1 0:00〜15:59:59.99999 (16 時間)	55
2016/05/1 (1 日)	72
2016/05/1〜2016/05/2 (2 日)	103
2016/05/1〜2016/05/4 (4 日)	138
2016/05/1〜2016/05/7 (1 週間)	198
2016/05/1〜2016/05/14 (2 週間)	272
2016/05/1〜2016/05/31 (1 ヶ月)	539
2016/05/01〜2016/06/30 (2 ヶ月)	871

5.2 結果

各期間からのルールでの精度について、感度と特異度の平均値を**図 6** に示す。また 2 ヶ月分のデータからルールを作成した時の 1 時間毎の感度の値の推移を**図 7**、特異度の値の推移を**図 8** に示す。

まず感度についての考察を述べる。図 6 から、感度はデータ数を増やすに連れて上がっていっていることが分かり、2 ヶ月の時には約 82%となっている。本研究ではサーバリストをポートチェックによって作成したため、サーバリストの中にはほとんど稼働していないサーバも登録されている。そこで教師データの作成方法を改善すればさらに高い感度での識別が行えるのではないかと考えられる。また図 7 から 2 ヶ月分のデータからルールを作成した時は、8 つの時間帯だけが 0.6 以下となったが、それ以外は比較的高い精度を保てているとわかる。この原因としては学内のサーバの多くで通常時と異なる何かが起こっているか、ルール自体を改善する必要があるかのどちらかである。この時間帯にサーバ群に何があったのかを検証することについては今後の課題とする。

次に特異度についての考察を述べる。図 6 から、特異度はデータ数に関係なくほぼ 100%であることが分かる。最大では感度と同様に 2 ヶ月の時で 99.6%であり、この時、1 時間あたり平均約 7 ホストがサーバと識別されていた。この偽陰性のホストはサーバリストに登録されていないサーバかもしくは別の要因が考えられるため、今後の調査が必要となる。また図 8 から 2 ヶ月分のデータからルールを作成した時は、時間帯に関係なく全ての時間で高い精度を保てているとわかる。

これらのことからデータ数を増やすことで、DNS クエリのデータからサーバに関しては 8 割以上、非サーバに関し

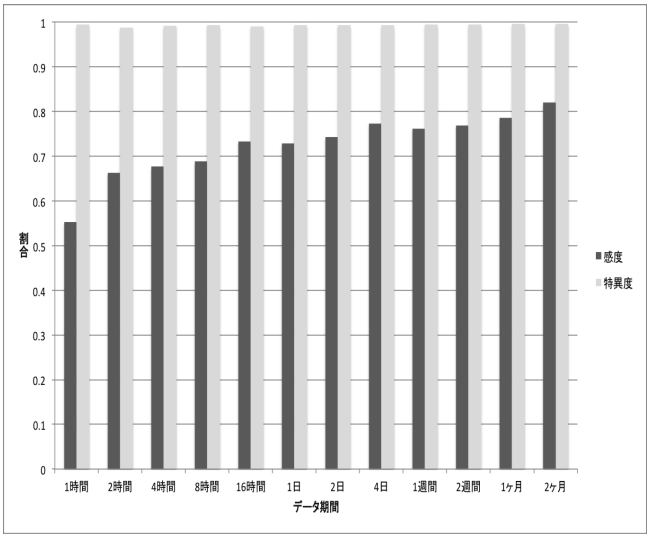

図 6 教師データに用いるデータの期間の変化による比較の結果

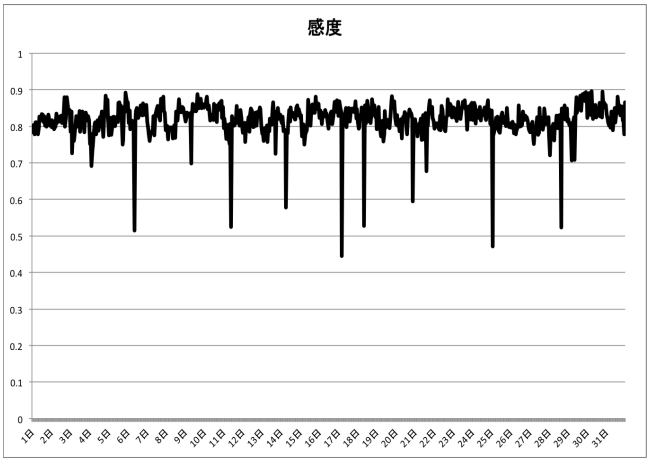

図 7 2 ヶ月分のデータでの 1 時間毎の感度の値の推移

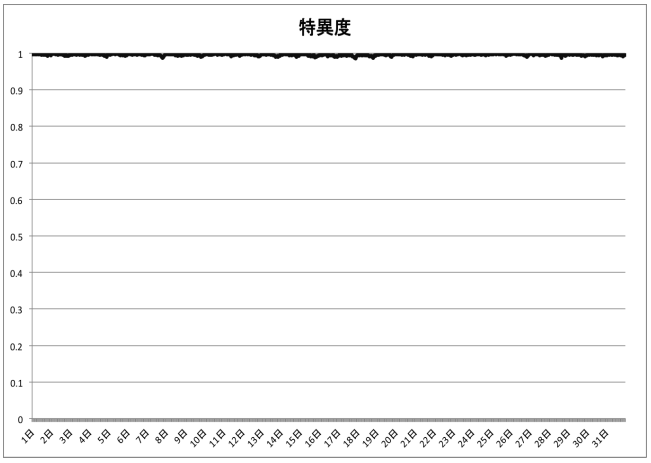

図 8 2 ヶ月分のデータでの 1 時間毎の特異度の値の推移

てはほぼ全てを正しく識別できると言える。

5.3 作成されたルールの考察

1 番精度の高かった 2016/05/01〜2016/06/30 での 2 ヶ月分のデータから機械学習を行った時に作成されたルールについて述べる。C5.0 では識別する全クラスに対して複数のルールを作成し、作成されたルールの 1 つ 1 つに信頼値 [7] という数値を割り振る。そしてルールに従ってクラス決定を行うがこの時複数のクラスのルールが当てはまった場合には、クラス毎に当てはまったルールの信頼値を合計しこの数値が一番高いクラスを選択する。つまりサーバとサーバ以外のどちらのルールにも当てはまった場合はその信頼値の合計が大きい方を採択する。このような方式から C5.0 ではサーバとサーバ以外の両方のルールが作成されるため、作成されたルールをサーバとサーバ以外に分けた時のそれらのルールの特徴について考察する。

表 3 2016/05/01〜2016/06/30 での各レコードタイプのクエリ数

レコードタイプ	クエリ数
A	991,339,953
AAAA	268,236,280
ANY	31,066,915
CNAME	198,681
MX	4,032,907
NS	902,939
PTR	40,821,288
SOA	2,003,621
SRV	1,830,351
SSHFP	100
TXT	6,635,749
unknown	5,213,377
合計	1,352,282,161

各レコードタイプのクエリが 2016/05/01〜2016/06/30 の期間で何件あるかを、**表 3** に示す。また**表 4** に、サーバとそれ以外に分けたときの詳細なルール数と使われている属性値の数を示す。表 3 と表 4 から、ルール数はクエリ数によっても影響されることがわかる。しかし、サーバ以外のルールと比較して、サーバのルールでは PTR レコードや TXT レコードや通信時間の割合などが頻繁に使われているため、これらの属性はサーバを識別するのに重要な属性と考えられる。

次に作成したルールに使用されている条件式から、各属性値に対するサーバとそれ以外のルールの特徴を述べる。

- クエリ数
  クエリ数が多いものはサーバの割合が増える傾向があるが、クエリ数だけでの判断はしづらい。クエリ数が上位もしくは下位のホストに対して他の属性値と組み合わせて使われている。
- A
  サーバは A レコード以外のレコードタイプも検索しに行くことが多いため、全体のうち A レコードの割合が低めになる傾向がある。そこで「A レコードが◯%以下」というルールが多く使われている。
  それに対して通常のクライアントはドメインの正引きが多いため A レコードの割合が高くなる。サーバ以外

表 4 サーバとそれ以外に分けたときの詳細なルール数と使われている属性値

クラス	ルール数	属性値	条件式での使用数
サーバ	678	クエリ数	772
		A	556
		AAAA	681
		ANY	23
		CNAME	25
		MX	141
		NS	163
		PTR	699
		SOA	288
		SRV	241
		SSHFP	0
		TXT	401
		unknown	152
		通信時間の割合	797
サーバ以外	193	クエリ数	199
		A	123
		AAAA	158
		ANY	19
		CNAME	4
		MX	136
		NS	101
		PTR	166
		SOA	78
		SRV	57
		SSHFP	0
		TXT	86
		unknown	61
		通信時間の割合	146

の中にはクライアントが多いため、サーバ以外のルールとして A レコードが一定数以上というのが用いられる。

- AAAA
  AAAA レコードに関しても A レコードと同様のことが言える。
- ANY
  ほとんどのホストがあまり使用しないレコードタイプであるため、ANY レコードから識別は難しい。ルールで使用される場合のほとんどは「ANY が 0 に近い時」という形で用いられる。ただし、一部のホストが ANY レコードを大量に検索しており、そのためクエリ数自体は少なくない。
- CNAME
  クエリ数も少なく、かといって誰でも使うレコードタイプであるため、識別の判断材料にならない。そのため識別ルールには殆ど使われていない。
- MX
  メールのやり取りに使われるレコードタイプであるため、主にメールサーバの識別に重要となると予想した。多くのルールで 0.1%付近を閾値としてそれより多いものはサーバと識別されている。その上で MX レコードが 0%に近いものを他の条件式と組み合わせて識別するのに用いられる。
- NS
  サーバ以外のホストではあまり使われず、「0%に近いものをサーバ以外」とするというルールが多かった。サーバの一部では数%の利用が見られ「3%以上ならサーバ」といったようなルールがいくつか見られた。
- PTR
  サーバで使われることの多いレコードタイプで、サーバでは一定数以上、サーバ以外では一定数以下というルールでの識別が多く見られた。ただしクエリ数が多いため識別の際には PTR レコードのみでの判断は難しく、他のレコードタイプと一緒に用いられる。
- SOA
  サーバではほとんど使われず、「0%に近ければサーバ」というように使われていた。逆にサーバ以外の判断材料として、クエリ量が少しあればサーバ以外であるというルールがいくつか見られた。
- SRV
  サービスを検索するレコードタイプであるためクライアントが利用することが多く、SOA レコードと同様の傾向が見られた。
- SSHFP
  クエリ数自体が極端に少なく、判断材料にならなかった。
- TXT
  サーバ以外で使われがちだが、ほとんどのホストで使うことはなく一部のホストで使用されるため TXT レコードのみでの判断は難しい。他の属性値と合わせて利用されることが多い。
- unknown
  unknown の中には DNSSEC など他のレコードタイプがあるが、サーバとそれ以外のどちらの場合でも unknown が 0%に近い時のルールがほとんどで、他の属性値と組み合わせて利用される。
- 通信時間の割合
  図 3 より、サーバはこの割合が高いものが多い。この割合が高いときはサーバの割合が多くなる。

これらは特徴であり実際にはルール 1 つに対していくつかの条件式を組み合わせて使用されている。表 4 からルール 1 つに対して平均すると約 7.2 個の条件式が使われていることが計算できる。

6. おわりに

本研究では DNS クエリを分析することによってホストをサーバと非サーバの 2 種類に分類することが可能であるかどうかの確認を行った。クエリのデータとして筑波大学

に設置されているフルサービスリゾルバへの DNS クエリを用いた。またサーバのリストを作成することでこの 2 つを教師データとした機械学習によっていくつかのルールを作成し、その精度について検証した。

本研究では、複数の教師あり機械学習アルゴリズムによる判定の結果と教師データのデータ数の変化による比較実験の実験を行った。教師データとしてクエリのデータと正解となるサーバリストを用いることでルールを作成し、そのルールに対する評価を行った。クエリデータとしてクエリ数、総クエリ数に対する各レコードタイプの割合、通信時間の割合を用いた。しかし、サーバリストに登録されているホストの中にはクエリ数の極端に少ないものもあり、その識別は困難だと考えるため毎時 60 件以上のクエリを送信しているホストのみを本研究の調査対象とした。サーバリストは調査対象のサーバのポートチェックを行うことで作成した。

本研究では教師あり機械学習アルゴリズムによる比較実験の結果と考察を述べた。その結果 C5.0 などのいくつかのアルゴリズムに関してはほぼ同等の精度で識別されており、大きな差がないことがわかった。

また本研究では教師データのデータ数の変化による比較実験の結果と考察を述べた。教師データに用いるクエリのデータ数を 1 時間分から 2 ヶ月分まで増大させた場合、この期間が長くなるに連れて感度が最大 82%まで上がっていくことがわかった。また作成したルールに対しその詳細を調査すると、特に MX,PTR,SOA,SRV, 通信時間の割合などはサーバとサーバ以外で傾向が異なるということがわかった。

今後の課題として、より正確なサーバリストの作成とクエリ数が 60 以下のホストの調査があげられる。そして、この研究を利用して実際にマルウェアを検知出来るか検証していくことが必要になる。

謝辞 また、多くの御指導、御助言を賜りました中井央先生に心より感謝申し上げます。ログデータの利用にご協力頂きました学術情報メディアセンター、およびログデータの見方に関してご協力いただきました藤原和典様には深く感謝申し上げます。

参考文献

[1] Christian J. Dietrich, Christian Rossow, Felix C. Freiling, Herbert Bos, Maarten van Steen, and Norbert Pohlmann. On botnets that use dns for command and control. In *Proceedings of the 2011 Seventh European Conference on Computer Network Defense*, EC2ND '11, pp. 9–16. IEEE Computer Society, 2011.

[2] Hannes Federrath, Karl-Peter Fuchs, Dominik Herrmann, and Christopher Piosecny. *Privacy-Preserving DNS: Analysis of Broadcast, Range Queries and Mix-Based Protection Methods*, pp. 665–683. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

[3] Dominik Herrmann, Christian Banse, and Hannes Federrath. Behavior-based tracking: Exploiting characteristic patterns indns traffic. *Computers & Security*, Vol. 39, Part A, pp. 17 – 33, 2013. 27th {IFIP} International Information Security Conference.

[4] Dominik Herrmann, Max Maaß, and Hannes Federrath. *Evaluating the Security of a DNS Query Obfuscation Scheme for Private Web Surfing*, pp. 205–219. Springer Berlin Heidelberg, Berlin, Heidelberg, 2014.

[5] CricketLiu(著), PaulAlbitz(著), 小柏伸夫 (訳). DNS&BIND 第 5 版. オライリー・ジャパン, 2008.

[6] P. Mockapetris. DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION. `https://www.ietf.org/rfc/rfc1035.txt`.

[7] RuleQuest Research. C5.0: An Informal Tutorial. `https://www.rulequest.com/see5-unix.html`.

[8] RuleQuest Research. RuleQuest Research Data Mining Tools. `https://www.rulequest.com/index.html`.

[9] Waikato 大学. Weka 3: Data Mining Software in Java. `http://www.cs.waikato.ac.nz/ml/weka/`.

[10] デニス・アルトゥロ・ルデニャ・ロマニャ, 杉谷賢一, 久保田真一郎, 武藏泰雄. DNS によるスパムボットとホスト探索活動の検知. 情報処理学会研究報告インターネットと運用技術（IOT）, Vol. 2008, No. 87, pp. 1–6, SEP 2008.

[11] 一瀬光, 金勇, 飯田勝吉. DNS における TXT レコードの利用方法の分析 (インターネット運用・管理, 一般). 電子情報通信学会技術研究報告. IA, インターネットアーキテクチャ, Vol. 114, No. 216, pp. 13–18, SEP 2014.

[12] 一瀬光, 金勇, 飯田勝吉. DNS における直接外部クエリー分析と不正通信検知手法の検討 (インターネットアーキテクチャ). 電子情報通信学会技術研究報告, Vol. 114, No. 495, pp. 173–178, MAR 2015.

[13] 株式会社 LAC. 遠隔操作ウイルスの制御に DNS プロトコルを使用する事案への注意喚起. `http://www.lac.co.jp/security/alert/2016/02/01_alert_01.html`.

[14] 足立堅一. 実践統計学入門. 株式会社篠原出版新社, 2001.

[15] 井出剛. 入門 機械学習による異常検知 -R による実践ガイド-. 株式会社コロナ社, 2015.

[16] 石橋圭介, 佐藤一道, 西田晴彦. 階層的集約エントロピーを用いた DNS ヘビーユーザの分類 (ネットワーク異常・侵入検知, インターネットと情報倫理教育, 一般). 電子情報通信学会技術研究報告. 技術と社会・倫理, Vol. 110, No. 429, pp. 237–242, FEB 2011.

[17] 津田航, 門林雄基, 藤原寛高, 山口英. 機械学習を用いた DNS 応答パケット分析によるマルウェア感染端末検知手法の検討 (情報通信システムセキュリティ). 電子情報通信学会技術研究報告 = IEICE technical report : 信学技報, Vol. 114, No. 340, pp. 67–72, NOV 2014.

[18] 武蔵泰雄, 松葉龍一, 杉谷賢一. プロトコル異常検知による A レコード型 DNS パケット分散サービス妨害攻撃の阻止. 情報処理学会研究報告インターネットと運用技術（IOT）, Vol. 2005, No. 83, pp. 23–28, AUG 2005.