一般社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report
IA2017-25 (2017-11)

低対話型ハニーポットとダークネットの関連付けによる新たなスキャン活動の検知手法の検討

秋吉　亮† 小谷　大祐†† 岡部　寿男††

† 京都大学大学院情報学研究科 〒606-8501 京都府京都市左京区吉田本町
†† 京都大学学術情報メディアセンター 〒606-8501 京都府京都市左京区吉田本町
E-mail: †aky@net.ist.i.kyoto-u.ac.jp, ††{kotani,okabe}@media.kyoto-u.ac.jp

あらまし　本稿では，低対話型ハニーポットとダークネットを利用した新たなスキャン活動の検知手法について論じる．高対話型ハニーポットは，豊富な情報を得られるが，セキュリティの観点から実装の困難性に課題がある．一方で，低対話型ハニーポットは，比較的安全に実装可能であるが，観測可能な攻撃は限定される．そこで，低対話型ハニーポットに，ダークネットを組み合わせ，両者で得られたデータ同士の関連付けを行う手法の検討を行う．これにより，豊富なデータの取得が可能となり，安全かつ効率的に新たなスキャン活動の自動検知が期待できる．本稿では，予備実験として Lurker で収集したデータのペイロードに基づいたクラスタリングを行い，その考察を行った．
キーワード　ダークネット, 低対話型ハニーポット, クラスタリング

Investigation of A Method for Detecting New Scanning Activities by Correlating Low-Interaction Honeypots with Darknet

Ryoh AKIYOSHI†, Daisuke KOTANI††, and Yasuo OKABE††

† Graduate School of Informatics, Kyoto University Yoshida-Honmachi, Sakyo-ku, Kyoto, 606-8501 JAPAN
†† Academic Center for Computing and Media Studies, Kyoto University Yoshida-Honmachi, Sakyo-ku, Kyoto, 606-8501 JAPAN
E-mail: †aky@net.ist.i.kyoto-u.ac.jp, ††{kotani,okabe}@media.kyoto-u.ac.jp

Abstract　In this paper, we discuss a method for detecting new scanning activities by using low-interaction honeypots and darknet. High-interaction honeypot can obtain rich information, however there is a problem in implementation difficulty from a viewpoint of security. In contrast, although low-interaction honeypot can obsarve only limited attacks, that can be implemented relatively safely. Then we investigate a method to associate data obtained from low-interaction honeypot with those from darknet. With this, new scanning activities can be detected automatically, as it is possible to get rich information safely and efficiently. In this paper, we conducted the experimental tests which classify data obtained by Lurker based on the payloads and its consideration.
Key words　Darknet, Low-interaction honeypot, Classification

1. はじめに

インターネット上では，ボットネットのワームによるスキャンや，Web ページの改ざん，個人情報の漏洩等，様々なサイバー攻撃が発生している．攻撃の発見・対応が遅れることで，さらなる被害の拡大が予想されるため，インシデント被害を最小限に抑えるためには，攻撃の早期発見が求められる．サイバー攻撃を検知・防御するためのシステムとして，IDS (Intrusion detection system) があり，最も有名なものとしては，Snort [1] や Bro [2] などのシグネチャベースの侵入検知システムが存在する．しかしながら，近年では IoT 機器をターゲットとするワームの Mirai やその亜種の発生をはじめとして，次々と新たな攻撃が観測されている．新たな攻撃はシグネチャを追加するまで検知できないため，シグネチャベースの IDS でこれらを検知することは困難である．したがって，シグネチャに依存しない侵入検知システムが必要となってきてる．

ダークネットは，インターネット上で到達可能な IP アドレスのうち，ホストが割り振られておらず，正規のトラフィックがほとんど，もしくは全く観測されない IP アドレス空間のことを指す [3]．通常のインターネットの利用では，未使用の IP

アドレスに対してパケットが到達することはほとんど無いが，実際には相当数のパケットが観測されている．ダークネットに到達するパケットのほとんどは，送信元 IP アドレスの偽装によるバックスキャッタや，リモート感染型のワームが送信するスキャン等の不正な活動に起因している．したがって，ダークネットで得られるデータの分析を行うことで，サイバー攻撃やマルウェア感染などの大局的な傾向を知ることができる．一方で，ダークネットではワームが送信するスキャン等の不正な活動の最初の 1 パケットのみを観測することができるので，TCP のペイロードを観てプロトコルやアプリケーション単位のスキャンを識別することが困難であるという課題がある．

ハニーポットは，意図的に不正アクセスを受ける事で，エクスプロイトコードやマルウェアの検体を収集することを目的として使用されるシステムの事を指す．目的の収集データによって模倣するサービスや実装が異なっており，対話レベルによって高対話型ハニーポットと低対話型ハニーポットに分類することができる．高対話型ハニーポットは，実際のサーバやサービスを利用したものである．実際のシステムを利用しており，攻撃者は様々な行動をとることができるため，攻撃者の挙動を詳細に観測することができる．一方で，通常のシステムを利用することによって，侵入された際のリスクが大きいという問題がある．例として Honeynet Project [4] がある．低対話型ハニーポットは，特定の OS やサービスの挙動やその脆弱性を模倣することで，攻撃者の挙動を観測するためのシステムである．実際のシステムを使用しないため，高対話型ハニーポットよりも安全に環境の構築を行うことができ，また，高度に模倣を行うことで，実際のシステムとほぼ同じ挙動を攻撃者に見せることができる．一方で，攻撃者は侵入したシステムがハニーポットもしくは仮想マシンであるかどうかを検知する場合があり，攻撃者の侵入後の挙動を観測できない事があるという欠点がある．低対話型ハニーポットの例としては，SSH の挙動を模倣する Cowrie [5] や複数のプロトコル (BLACKHOLE, EPMAP, FTP, HTTP, MEMCACHE, MIRROR, MQTT, MSSQL, MYSQL, PPTP, SIP, SMB, TFTP, UPNP) に対応した Dionaea [6], TCP SYN に対して SYN + ACK および ICMP Echo に対して ICMP Reply を返す Lurker [7] などがある．

本研究では，マルウェアや攻撃者が使用するツールによる新たなスキャン活動を安全かつ効率的に早期検知を行う目的とする．安全な環境を実現するために，低対話型ハニーポットは特に対話レベルの低い Lurker を使用した．Lurker は TCP SYN と TCMP Echo にしか応答せず，ペイロードは基本的に最初の 1 パケットしか得られないため，Cowrie や Dionaea のような低対話型ハニーポットと比較するとそれらが対応するプロトコルに関して得られる情報量の面で劣るが，特定のプロトコルに依存せずにデータを取得できるため、多様なスキャン活動の検出が可能である．Lurker で得られたデータにダークネットで得られたデータを関連付けることで，豊富なデータの取得が可能となりその課題を解決する．本稿では，予備実験として Lurker で収集したデータのペイロードに基づいたクラスタリングを行い，その考察を行った．

以下に本稿の構成を述べる．第 2 章では，関連研究を紹介する．第 3 章では，本研究で提案する手法について述べる．第 4 章では，提案手法に基づいて行った予備実験とその結果について述べる．第 5 章では，今後の方針について述べる．

2. 関 連 研 究

ハニーポットのデータとダークネットのデータを関連付けるものとしては，Correlation Analysis System [8] がある．これは，Honeypot で収集したマルウェアと Darknet のパケットを関連付け，システム管理者に分析結果を通知するシステムである．本システムは，Macro analysis Sysmtem (MacS), Micro analysis System (MicS), Network and malware enchaining System (NemeSys), Incident Handling System (IHS) の 4 つの要素に分けられる．MacS では，ダークネットおよび ICMP と SYN にのみ応答する低対話型ハニーポットで収集されたデータを使用する．ダークネットのデータから自己回帰モデルでスコアを算出し，スコアが閾値を超えたら変化点として検出する．また，低対話型ハニーポットで得られたデータからエクスプロイトコードを検出する．MicS では，ハニーポットで収集した実行可能マルウェアのデータを使用する．実際にマルウェアを動作させる動的解析を行い，マルウェアが使用した API とその引数，変更ファイル・変更レジストリ，アクセス先 URL 等を取得する．また，インターネットエミュレータの機能を含むサンドボックスを使用した分析を行い，インターネットエミュレータ上のログや，マルウェアが使用した API 等の情報を取得する．NemeSys では，MacS および MicS で得られた結果からプロファイルを作成し，プロファイル同士の類似性を計算することで，両者で得られたデータを関連付ける．IHS は分析結果をシステム管理者に提供するシステムであり，各アナライザからレポートされた情報を管理する．システム管理者は GUI でこれらの情報にアクセスすることで，インシデントへの早期対応が可能となる．ここでは，収集する実行可能なマルウェアがハニーポットで収集可能なものに限られるという点が課題である．

Hoepers らは高対話型ハニーポットの実装方法および得られた結果を CSIRT に適用するためのアラート方法について述べている [12]．ハニーポットの実装のシステム構造を図 1 に示す．Firewall では全ての受信パケットを許可し，送信元を偽装したパケットのような不正な疑いのある送信パケットをブロックする．Hogwash では，シグネチャの存在する不正な送信パケットをブロックする．これらにより，感染端末となったハニーポットが攻撃者となることを防ぐ．Forensics はファイルサーバであり，IDS やネットワーク上を流れるパケットを収集したデータを保管している．ハニーポットのディスクイメージも保存しており，緊急時にはハニーポットを保存されたディスクイメージの状態に復元する事が可能である．また，パケットのフィルタリングについて詳細にルールを記述することが可能であり，コネクション数制限や帯域制限を行うことができる．豊富な拡張性を持つ一方で，システムが大規模であるため，実装および保守が容易ではないという課題がある．

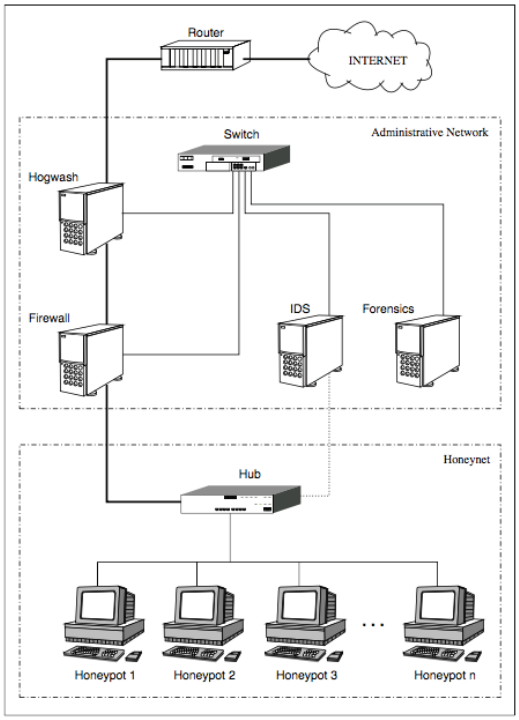

図 1　システム構造

ペイロードに基づいた侵入検知システムとしてはPCNAD [11] がある．PCNAD は，ペイロードに n-gram($n = 1$) をバイト単位で適用してプロファイルを作成する．ペイロードは多くの場合プロトコルおよびペイロード長によってその内容が異なるため，作成したプロファイルは宛先ポート番号・ペイロード長毎に分割する．また，この分割によって，プロファイルが肥大化する問題が発生するが，ペイロード長の差が閾値 l_t 以下の場合にプロファイル同士をマージすることでこの問題を解決する．マージ後にプロファイル同士のマンハッタン距離 m_d を計算し，m_d が閾値 m_t 以下のときに同じクラスタにまとめることで，クラスタリングを行う．受信パケットに対しても同様の計算を行い，それまでに作成されたどのクラスタにも属さなかった場合にアラートを発することで新たな攻撃の検出が可能となる．

ダークネットのデータを利用した研究としては，Cowie らが機械学習に適用するための学習データの生成を可能としている [9]．ダークネットで得られたデータと既知のインシデントを関連付けるために，手動および自動生成の 2 つのデータ生成方法について検討している．手動生成では，膨大なデータに対して手動で関連付けを行う必要があり，関連付けを行う人物の知識に依存するヒューリスティックについて考慮する必要がある．一方自動生成では，既知のマルウェア挙動のモデルを利用してシミュレートされたデータセットの構築を可能としている．また，小出らは独自のネットワークスタックを用いて通信を行うマルウェアがヘッダ情報に固有の特徴を持つという性質を利用して，不正な通信の特定を可能とした [10]．

3. 提 案 手 法

本研究では，低対話型ハニーポットとダークネットの両者で得られたデータを関連付けることで，安全かつ効率的に新たなスキャン活動の自動検知を行うことを目的としている．

ペイロード長 [byte]	マージ済ペイロード長 1 [byte]	マージ済ペイロード長 2 [byte]
1	3.5	[2]
2	3.5	[2, 3]
3	3.5	[2, 3, 4]
4	3.5	[3, 4, 5]
5	3.5	[4, 5]
6	3.5	[5]
9	9	[9]

表 1　ペイロード長に基づくクラスタリングの例

3.1 ペイロードのクラスタリング

パケットのペイロードにラベルを付加するため，クラスタリングを行う．これにより，特定のラベルが付加された攻撃に関するトラフィック量がそれまでとは異なる挙動を示した場合，もしくは，新たなラベルが生成された場合に，新たなスキャン活動として検知することが可能となる．また，本研究では，新たなスキャン活動の早期検知を目的としているため，クラスタリングは計算量の比較的小さい 1-gram を使用した．ペイロードに対してバイト単位で 1-gram を使用し，文字の頻度分布をプロファイルとする．また，プロトコルやペイロード長が異なる場合ペイロードの内容も大きく異なるため，プロファイルを宛先ポート番号，および，ペイロード長毎に作成．

ペイロード長の差が閾値 l_t 以下のプロファイル同士をマージし，マージされたペイロード長の平均をマージ済ペイロード長とする．表 1 に $l_t = 2$ とした場合の具体例を示す．マージ方法はいくつかの方法が考えられるが，ここでは 2 つの例について述べる．マージ済ペイロード長 1 は，最も近いペイロード長の差が l_t を超えるところで分割し、分割された各ペイロード長の平均をマージ済ペイロード長としてマージしたものである．この方法では，ペイロード長 1 と 6 の差は l_t を大きく上回っているにもかかわらず同じ長さにマージされるという問題がある．マージ済ペイロード長 2 は，マージ済ペイロード長をリストの形で保持し，同じ長さにマージされた全てのペイロード長の組の差は l_t 以下を満たすことで上述の問題を解決している．しかし，この方法では，プロファイルが肥大化するため，計算量が肥大化するという問題がある．本研究では，マージ済ペイロード長 2 の方法を使用した．

ペイロード長によってマージされたプロファイルにおいて，ペイロード同士の距離を式 (1) で定義されるマンハッタン距離 m_d の算出を全ての組み合わせに対して行う．

$$m_d(x, y) = \sum_{k=1}^{n} |x_k - y_k| \tag{1}$$

m_d が閾値 m_t 以下の場合に同じクラスタと判定し，パケットにラベルを付加する．

3.2 パケットのクラスタリング

ダークネットおよび Lurker で得られたパケット毎のプロファイルを作成するために，以下のパラメータを使用する．

- 日時

- ID
- TTL
- srcIP, dstIP
- プロトコル ([TCP, UDP, ICMP])
- srcPort, dstPort (TCP, UDP の場合)
- seqnum (TCP の場合)
- acknum (TCP の場合)
- flags (TCP の場合)
- win (TCP の場合)
- opts (TCP の場合)
- ペイロード (TCP かつペイロードを含む場合)

プロファイルを作成後，クラスタリングを行う．

3.3 異常検知

キャプチャしたパケットのペイロードに対して，前述の手法と同様の手法でラベルを付加した後に，ダークネットとの関連付けを行い，プロファイルを作成する．作成したプロファイルがそれ以前に作成したどのクラスタにも当てはまらなかった場合，もしくは，それまでとは異なるトラフィック量を検出した場合に，新たなスキャン活動として検知を行う．

4. 実　　験

4.1 使用データ

実験には，低対話型ハニーポットとしての Lurker，および，ダークネットで収集したデータを使用した．Lurker にはダークネットとして観測しているネットワークに近接する IP アドレスを 1 つ割り振っており，ダークネットは 12 個の/24 ネットワークを観測している．ダークネットは 2016 年 12 月 14 日から 2017 年 9 月 29 日までのデータを使用し，Lurker は 2017 年 12 月 18 日から 2017 年 9 月 29 日までのデータを使用した．

4.2 実　　験

4.2.1 バイト単位の頻度分布

まず，プロトコル毎のバイト単位での頻度分布についての分析を行った．頻度分布として得られた結果の例を図 2, 3 に示す．図 2 は SSH に関連するポート番号宛のバイト単位の頻度分布をプロットした図であり，図 3 は同様に HTTP に関連するポートの頻度分布をプロットした図である．また，どちらも最も多かった文字を 1 として正規化している．図 2 では，22 番ポート以外の 3 つは似た傾向を示すことが分かった．このような結果が得られた原因として，22 番ポート宛パケットのほとんどは SSH クライアントの PUTTY からのリクエストもしくはそれを模倣したリクエストであり，その他の 3 つのポート番号宛のパケットのほとんどは SSH ライブラリの libssh を利用したリクエストである為であった．図 3 では，NULL バイトの頻度の差がみられるが，その他の文字の分布は似た傾向を示していた．

4.2.2 ペイロードのクラスタリング

予備実験として，$P_{d,l}(d = 80)$，すなわち，80 番ポート宛のパケットのペイロードに対して実験を行った．1-gram を用いてプロファイル $P_{d,l}$ を計算し，ペイロード長 l の差が 5 以下のプロファイルをマージした．マージ後，マージ済みプロファイルにおけるマンハッタン距離の差 m_d が閾値以下のものを同じクラスタとしてクラスタリングを行った．閾値 m_t は，固定長に場合，ペイロード長によってその影響度に差が出るため，$m_t = payload\_length * 0.2$ として定義した．

ペイロードのクラスタリングを行った結果から，式 (2) によってペイロード自体は同じではないが，同じクラスタと判定されたペイロードの組を抽出した．

$$(m_d != 0)\&\&(m_d <= m_t) \tag{2}$$

抽出した出力結果の一例を出力結果 1, 2 に示す．出力結果 1 は，マージ済みペイロード長が 19 で同じクラスたと判定されたペイロードの組である．ここでは，HTTP のバージョンが異なっている．出力結果 2 は，マージ済みペイロード長が 193 の組であり，phpMyAdmin の脆弱性を狙ったリクエストであるが，標的とする phpMyAdmin のバージョンが異なっている．これらの他にも，大文字・小文字が異なるものや，改行コードが異なるものが同じクラスタと判定された組が存在した．このクラスタリング結果に対して，今後ダークネットとの関連付けによるスキャン活動の検知結果に応じて，閾値の検討や他のクラスタリング手法の検討を行う．また，本稿では 80 番ポート宛のパケットのみのクラスタリングを行ったが，計算に非常に長い時間がかかった．したがって，他のポート番号宛についても同様の計算を行うためには，高速なアルゴリズムの検討が必要である．

5. 今後の方針

本稿では，低対話型ハニーポット (Lurker) で得られたデータから，ペイロードに基づくクラスタリングを行った．その結果，攻撃対象が異なるものを同じクラスタと判別されたものが存在することが分かった．したがって，ペイロードのクラスタリングについてさらなる検討を行う．また，Lurker とダークネットの関連付けを行うために，さらなる予備実験やサーベイを行ったのちに，実装・評価を行う．

文　　献

[1] Snort - Network Intrusion Detection & Prevention System, https://www.snort.org/, (October, 2017)

[2] The Bro Network Security Monitor, https://www.bro.org/, (October, 2017)

[3] Harrop, Warren, and Grenville Armitage. "Defining and evaluating greynets (sparse darknets)." Local Computer Networks, 2005. 30th Anniversary. The IEEE Conference on. IEEE, 2005.

[4] The Honeynet Project, https://www.honeynet.org/, (October, 2017)

[5] micheloosterhof/cowrie: Cowrie SSH/Telnet Honeypot, https://github.com/micheloosterhof/cowrie, (October, 2017)

[6] DinoTools/dionaea: Home of the dionaea honeypot, https://github.com/DinoTools/dionaea, (October, 2017)

[7] m-mizutani/lurker: Lurker monitors an incoming network TCP SYN packet to invalid port number, and will respond a TCP SYN-ACK packet to collect packet payload from attacker, https://github.com/m-mizutani/lurker

[8] Nakao, Koji, et al. "Practical correlation analysis between scan and malware profiles against zero-day attacks based on darknet monitoring." IEICE TRANSACTIONS on Information and Systems 92.5 (2009): 787-798.

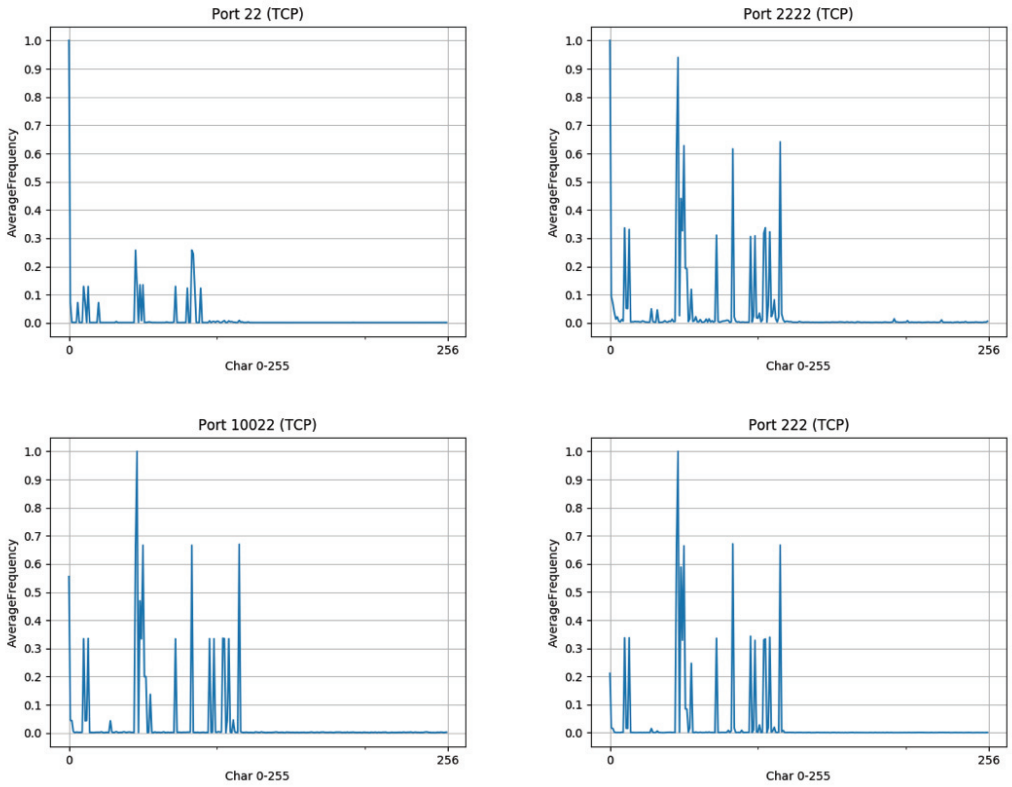

図 2　SSH に関連するポート番号宛のバイト単位の頻度分布

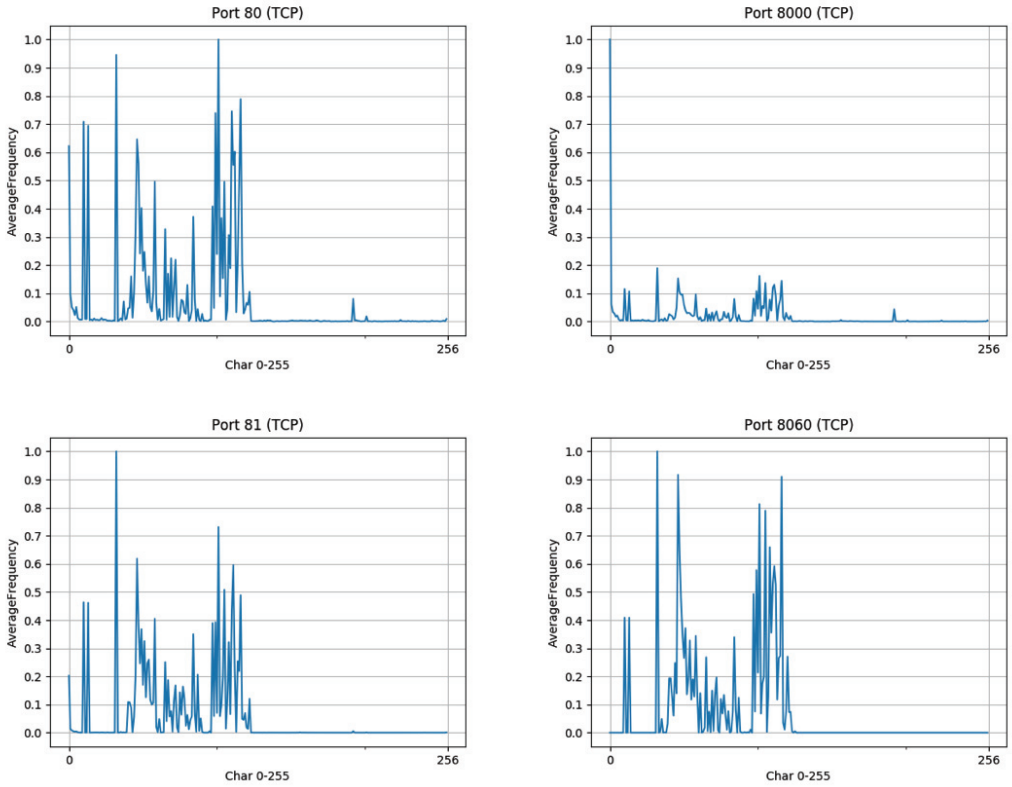

図 3　HTTP に関連するポート番号宛のバイト単位の頻度分布

[9] Cowie, Bradley, and Barry Irwin. "Data classification for artificial intelligence construct training to aid in network incident identification using network telescope data."ÂăProceedings of the 2010 Annual Research Conference of the South African Institute of Computer Scientists and Information Technologists. ACM, 2010.

[10] 小出駿, et al. "通信プロトコルのヘッダの特徴に基づく不正通信の検知・分類手法." コンピュータセキュリティシンポジウム

2014 論文集 2014.2 (2014): 48-55.
[11] Thorat, Sandeep A., et al. "Payload content based network anomaly detection." Applications of Digital Information and Web Technologies, 2008. ICADIWT 2008. First International Conference on the. IEEE, 2008.
[12] Hoepers, Cristine, Klaus Steding-Jessen, and Antonio Montes. "Honeynets Applied to the CSIRT Scenario." Proceedings of the 15th Annual Computer Security Incident Handling Conference. 2003.

出力結果 1　ペイロード長 19 で同じクラスタと判定されたペイロード

```
Length: 19.000000
Manhattan Distance: 2
--
HEAD / HTTP/1.0


--
HEAD / HTTP/1.1


====
```

出力結果 2　ペイロード長 193 で同じクラスタと判定されたペイロード

```
Length: 193.000000
Manhattan Distance: 9
--
GET /phpMyAdmin-2.11.1-all-languages/scripts/setup.php HTTP/1.1
Accept: */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: ZmEu
Host: 133.69.4.2
Connection: Close


--
GET /phpMyAdmin-3.0.0.0-all-languages/scripts/setup.php HTTP/1.1
Accept: */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: ZmEu
Host: 133.69.4.2
Connection: Close


====
```