侵入検知システムログにおける攻撃データのクラスタリングと傾向分析

水谷 正慶 †

† 日本アイ・ビー・エム 東京基礎研究所
135-8511 東京都江東区豊洲 5-6-52 NBF 豊洲キャナルフロント
masa@jp.ibm.com

ああ 情報技術の発展および普及に伴い, インターネット上での悪意ある人物によるセキュリ ティ侵害が深刻な課題となっている．悪意ある人物がインターネット上で攻撃をする場合，同じ脆弱性を利用した攻撃でも利用される攻撃コードは様々である．これは悪意ある人物が利用するツールに関連すると見られ，攻撃手法や悪意ある人物の変遷を調査する手がかりになると考えられる．本稿では侵入検知・防止システムのログに含まれる攻撃コードをクラスタリングすることで攻撃の傾向を分析する手法について論じる．さらに事例として ShellShock 脆弱性を利用した攻撃にたいし，本手法を用いて分析した結果について述べる．

Clustering and trend analysis of Intrusion Detection System Logs

Masayoshi Mizutani†

†IBM Japan, Tokyo Research
NBF Toyosu Canal Front Building 5-6-52 Toyosu,Koto-ku Tokyo, 135-8511 Japan
masa@jp.ibm.com

Abstract Malicious users leverage exploit code to breach security of systems on the internet. Exploit code has a lot of varieties even if same vulnerability is used by the exploit codes. It's caused by variety of attacker's tool and changes of exploit code can indicate trend of attackers and malicious tools. In this paper, we show a method of clustering attack data and our experiences of analysis.

1 はじめに

インターネットの普及に伴い遠隔からの電子情報機器への攻撃は急増し，その手法も多様化を続けてきた．このような攻撃はIDS(侵入検知システム) [1] をはじめとするセキュリティ装置によって監視することができるため，分析によって防御に有用な情報を入手することができる．例えばIDSによって検知された攻撃コードを分析することで攻撃の成否を確認したり，誤判定である False Positive を起こしたシグネチャの改善に役立てる事ができる．また，同じシグネチャによって検知されたイベントであっても異なる攻撃ツールを利用した攻撃の可能性もあり，攻撃コードの変遷を追跡することによって攻撃ツールの普及に関する調査や，同じ攻撃ツールを利用した攻撃者の調査に利用できる．このような調査や攻撃者や攻撃元ホストの背後関係を調査するのに役立ち，異なるセキュリティ装置

[1] 侵入防止システム (IPS) も同様の機能をもつが本稿では攻撃コードの監視に着目するため IDS に統一して論じる

を連携させるのに役立てる事ができる．

分析では広範かつ多数のデータが必要となるが，データサイズが増えることによって分析者の負担も大きくなる．情報セキュリティに関連する攻撃は複雑化を続けており，攻撃も継続的に実施されている．攻撃に使われるコードは人間の認知できる範囲であれば整理，分類が可能だが，データサイズが大きくなることによってパターンの認識が難しくなってくる．データに対して部分的にパターンを作成してそのパターンにマッチするものとしないものを仕分けし，しないものに対して新しいパターンを作成するというような繰り返し作業での分類はできるが，データサイズが大きくなると繰り返しそのものに時間がかかるようになってしまい，やはり分析者が認知できる範囲が限られてしまう．

本稿では大量の攻撃コードを分析する前に攻撃コードの類似性をもちいてクラスタリングし，分析者の負担を軽減するアプローチについて述べる．大量のデータを機械的にクラスタリングすることにより分析者がデータ分類の試行を繰り返すことなく負担を軽減することが期待される．本稿では ShellShock 脆弱性を利用した攻撃についてクラスタリング手法を適用し，分析者の手を介することなく攻撃ツールや攻撃者の意図に応じて攻撃コードを分類できることを示した．

2 背景

インターネット上の攻撃コードは様々な方法で収集できる．IDS などのセキュリティ装置によって監視対象となっているネットワークの通信から取得したり，ハニーポットやダークネットのような方法によってインターネット上で無差別に攻撃するデータを収集することができる．

攻撃コードは同じ脆弱性を利用した場合でも，異なる攻撃コードとなる可能性がある．まず，同じ脆弱性を利用した攻撃でも異なる攻撃コードを使用できる場合がある．脆弱性の構造上，特定の値をコードに含めればよいわけではなく，値の組み合わせ条件やある値以上もしくは以下のデータを送信することで発現する脆弱性の場合は攻撃コードが一定になるとは限らない．また，脆弱性を利用する部分のコードが同じだとしても，攻撃に用いられるパラメータが異なる場合が挙げられる．例えば攻撃成功時に外部のサーバからマルウェアをダウンロードし攻撃対象のホストに感染させる攻撃の場合，マルウェアを設置している外部サーバが攻撃対象や攻撃時期によって変更される可能性がある．これは，攻撃対象によって攻撃成功時に得られる結果を変えたり，マルウェアを設置している外部サーバを長期間利用することによりそのサーバがブラックリストに掲載されることを回避するといった目的が考えられる．さらにコードの様々な箇所に無意味な命令や分岐を挿入したり一部を暗号化することによって，攻撃の検知や防御を防ぐ難読化によってコードが書き換わる場合がある．

図 1 に ShellShock 脆弱性を利用した攻撃の例を示す．この攻撃は同一の攻撃ツールを使っていると考えられるが攻撃コードに差異がある．一部伏せ字としているが，攻撃成功時に curl コマンドによって外部サーバへアクセスし，なんらかのコードをダウンロードした後にその結果を php コマンドへ標準入力として渡して実行を試みる攻撃となっている．ダウンロード元のサーバや URL のパスが同じことから同じツールを使った同一の攻撃者による攻撃コードと見られるが，URL 後半のパラメータである `tok` には 1 行目と 2 行目で異なる値が格納されている．これはパラメータの名称から，攻撃者が攻撃成功時にアクセスされる外部サーバでどの攻撃が成功したのかを識別するためにコードを変更していると推察される．

このような攻撃コードを分析することにより IDS のシグネチャの改善や新しい攻撃の変化を兆候としてとらえることができる．また，攻撃ツールの変遷や共通点を見つけることにより，攻撃に関する背景を推測することができる．これにより，攻撃時の IP アドレスや攻撃対象などに共通する情報がなかったとしても，同じ攻撃者が実施している攻撃であるということを追跡できる．

しかし，有意なデータにするためには多くのデータを収集する必要がありデータサイズが大

```
1| () { :;}; /bin/bash -c 'curl http://46.*.*.*/search/zz/j.php?tok=***************bb501f952ae8152ce | php
2| () { :;}; /bin/bash -c 'curl http://46.*.*.*/search/zz/j.php?tok=***************c28a1a0649f903d0b | php
```

図 1: 類似するが攻撃コードに差異が見られる例．一部の文字を伏せ字にしている．攻撃成功時にアクセスする URL の構造から，攻撃者が攻撃成功時にアクセスされる外部サーバでどの攻撃が成功したのかを識別するためにコードを変更していると推察される

きくなった場合に分析者が手動ですべてのデータを分析するのは困難である．筆者が多地点で観測した 10,000,000 件の ShellShock 脆弱性を利用した攻撃データについて調査したところ，248,126 種類の攻撃コードが確認された．これは主に攻撃毎にパラメータを変更することによって，攻撃成功時の挙動を変化させることを目的にしていると見られる．これらの攻撃コードを分析者が主導で全てチェックするのは現実的ではなく，機械的にデータを分類し分析を補助することが重要となる．

3 アプローチ

本稿では攻撃データのクラスタリングをすることによって類似した攻撃を分類し分析に活用した．類似する攻撃コードをクラスタとしてまとめることによって攻撃する際に変更されるパラメータの違いによる差異を吸収し，分析に必要となる手間を軽減できる．

クラスタリングのアルゴリズムをアルゴリズム 1 に示す．本アルゴリズムでは攻撃コードのデータセット D とクラスタ間のしきい値 t を与える．$0 < t < 1$ とする．攻撃コードのクラスタリングでは事前にどの程度のクラスタ数になるかの予測がしにくいという特徴があるため，事前にクラスタ数を指定するのではなく，クラスタ間の距離をあらかじめ指定する手法とした．基本的な手順としては，データセット内の攻撃コードを順番に既存のクラスタの代表値と類似度を比較して最も類似度の高かったクラスタに攻撃コードを追加する．ただし，類似度が事前に指定したしきい値を下回っていた場合や既存のクラスタが存在しなかった場合は，d を代表値とする新たなクラスタを作成する．

本稿では ShellShock 脆弱性を利用した攻撃コードを分析の対象とした．ShellShock 脆弱性

Algorithm 1 クラスタリングのアルゴリズム

Require: D, t
  $C \leftarrow \emptyset$
  for all $d \in D$ **do**
    $C_{max} \leftarrow nil$
    $r_{max} \leftarrow 0$
    for all $c \in C$ **do**
      $r \leftarrow ratio(d, c)$
      if $r_{max} < r$ **then**
        $r_{max} \leftarrow r$
        $C_{max} \leftarrow c$
      end if
    end for
  end for
  if $C_{max} = nil \wedge t \leq r_{max}$ **then**
    $c_{new} \leftarrow NewCluster(d)$
    $C \leftarrow C \cup c_{new}$
  else
    $AddToCluster(c, d)$
  end if

を利用した攻撃コードには以下のような特徴がある．

- Bash に実行命令がそのまま引き渡されるためテキスト形式で記述されている
- 攻撃コードの送信に制限がある場合が多く難読化のような処理が入っていない
- 攻撃コードに順序性があり前方から順に実行される命令になっている

これらの性質から攻撃コード間の類似性はレーベシュタイン距離に基づく方法を採用した．アルゴリズム 1 中の $ratio(d, c)$ 関数が類似度を計算する．類似度は攻撃コード d とクラスタ c の代表値のレーベシュタイン距離を計算し，比較対象となる攻撃コードの文字列長を基準に 0 以

上 1 以下の値に正規化する．値が大きいほど類似し，小さいほど異なるものとする．

このアルゴリズムによりクラスタ間の距離が t 以上となるクラスタ郡が C に作成される．計算量は $O(|D| \cdot |C|)$ となるが，しきい値を適切に設定することで $|C|$ は収束してゆくので，平均的には $O(|D|)$ となる．

4 実験と考察

4.1 クラスタリングの結果と考察

第 3 節で示したアルゴリズムにより攻撃コードをクラスタリングし，分割された結果に対して分析を実施した．攻撃コードには複数の環境において検知された ShellShock 脆弱性を利用する攻撃を用いた．HTTP サービスを対象として HTTP リクエストヘッダに含まれていた攻撃コードを対象としている．期間は 2014 年 9 月以降から 2015 月 3 日 27 日までの 10,000,000 件のデータを利用し，しきい値は 0.6 に設定した．

この手法により攻撃コードは 916 個のクラスタに分割された．表 1 および表 2 にクラスタリングした結果とその特徴について示している．表 1 ではクラスタに含まれたイベント数の上位 10 クラスタを示し，表 2 ではクラスタに含まれる攻撃コードの種類数の上位 10 クラスタを示している．攻撃コードの種類は 1 文字でも異なれば違う攻撃コードとして集計している．表 1 で示しているクラスタは全体のイベント数に対して約 55.81%のイベント数を網羅しており，表 2 で示しているクラスタも全体の攻撃種類数に対して約 67.86%の攻撃種類数を網羅している．

表 1 および表 2 で種別数が多いのは攻撃コード内のパラメータと思われる部分を攻撃対象や攻撃の試行毎に変化させているものとなっている．これは攻撃者が調査活動などをする際，どの攻撃対象に対して実施した攻撃であるかを識別するために，トークンとなる識別子や攻撃対象を識別できる情報を利用しているためと考えられる．また，同様の攻撃手法であったとしても攻撃対象となるホスト環境に合わせ，利用するコマンドやファイル名の変化も見られる．このような理由により攻撃コードが多様化しており，データサイズが大きくなった場合に手動での分類が難しいことがわかる．

4.2 クラスタリング結果に対する正規表現の記述

表 3 では種類数上位 10 クラスタに対し，クラスタリングした結果を正規表現でパターン化できるかどうかを示した表である．本稿で示したクラスタリング手法は代表値を決めてこれに類似するデータを新しくクラスタに所属させていく手法である．そのためクラスタ内部に含まれる攻撃コード同士の距離についてはアルゴリズム上保証されておらず，正規表現での記述が難しいクラスタが生成される可能性もある．このため出現する攻撃コードがもっとも変化するクラスタに対してどのような正規表現ができるかについて示した．正規表現は筆者の目視によって手動作成している．

表 3 で示している通り，固定された文字列を含む正規表現を作成できている．また表中ではこれらの正規表現に対してクラスタに含まれている攻撃コードが一致した数と全種類数に対する割合を示している．割合は小数点 2 位以下を切り捨てている．c_{11} から c_{19} までのクラスタでは全てに一致する正規表現を記述することができた．c_1 については `bash_cve_2014_6271_rce` を含まない脆弱性検証のコードが含まれていることがわかったが，クラスタ内の攻撃コード全種類に対する 0.2%ほどの割合であり，許容できると考えられる．これにより攻撃コードのクラスタリングによりパターン化可能なクラスタが生成できることがわかった．

4.3 クラスタごとの攻撃傾向に関する分析

本稿で示した手法を用いることで攻撃の意図やツール別にクラスタを作成できていることから，攻撃の傾向もそれぞれのクラスタで異なってくる．図 2 と図 3 ではそれぞれ c_1 クラスタおよび c_4 クラスタの攻撃元 IP アドレスと攻撃先の関係を示している．赤いノードは攻撃元 IP

表 1: クラスタサイズの上位 10 クラスタとクラスタの特徴

クラスタ	イベント数	攻撃コード種別数	クラスタの代表的な攻撃コードの特徴
c_1	2,774,956	10,023	`bash_cve_2014_6271_rce` という文字列が含まれており，Shell 上で数値計算をした結果を表示させている．Nessus のプラグインによる脆弱性検査に同様のものがある．
c_2	1,211,321	9	curl もしくは wget コマンドを用いて index.cgi，index.png，index.jpg などのファイルを外部サーバよりダウンロードし，その結果をパイプ機能によって perl コマンドへ入力しスクリプトの実行が試みられる．
c_3	315,214	24	0 秒〜60 秒の sleep コマンド実行を試みる．
c_4	262,556	10	`/share/HDB_DATA/.../` というディレクトリを作成したあと，スクリプトとみられるファイルをダウンロードしてそのスクリプトの実行を試みる．QNAP 社の NAS 脆弱性を狙ったワームと見られる．
c_5	227,437	86	スクリプトとみられるファイルをダウンロードしてそのスクリプトの実行を試みる．攻撃成功時に `XSUCCESS` という文字を表示させる特徴がある．
c_6	212,566	36	攻撃対象ホストの`/etc/passwd` ファイルの表示を試みる．
c_7	206,795	8	ShellShock 脆弱性によるコード実行の起点となる文字列のみで，特定のコマンドが指定されていない．
c_8	130,534	27	c_5 と同様の特徴を持つ．
c_9	120,704	20	Web サーバが利用しているディレクトリ以下にファイルを生成し，`CVE-2014-6271` という文字列の書き込みを試みる．
c_{10}	118,951	1,755	`cs` という文字列と MD5 ハッシュ値とみられる文字列の結合した結果の表示を試みる．

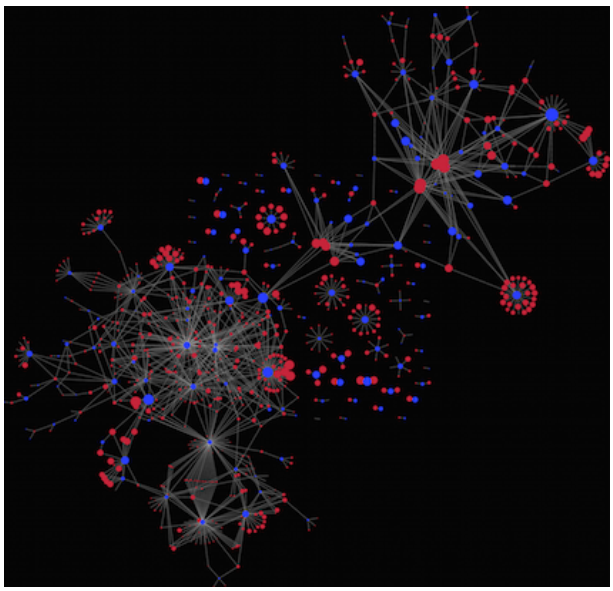

図 2: c_1 の攻撃元 IP アドレスと攻撃先の関係図

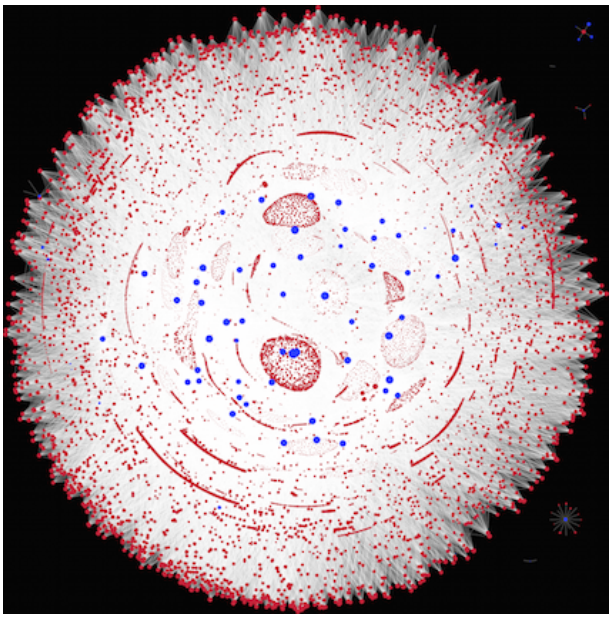

図 3: c_4 の攻撃元 IP アドレスと攻撃先の関係図

表 2: 攻撃コード種類数の上位 10 クラスタとクラスタの特徴

クラスタ	イベント数	攻撃コード種別数	クラスタの代表的な攻撃コードの特徴
c_{11}	44,776	34,180	whsec.us という文字列を含み，ping，curl，wget コマンドの実行によって脆弱性有無の確認を試みる．
c_{12}	31,360	28,009	c_{11} 同様に whsec.us という文字列を含むが，特に`/bin/bash`経由で wget コマンドを起動している点が特徴である．
c_{13}	38,435	24,353	外部のサーバに curl でのアクセスを試みる．アクセスする際の URL に攻撃対象のサーバ名が含まれるのが特徴である．
c_{14}	23,570	21,335	c_{11} 同様に whsec.us という文字列を含み，主に wget あるいは curl コマンドで外部サーバへアクセスしている．
c_{15}	18,206	17,153	外部のサーバにアクセスしシステム情報などの送信を試みる．`/dev/tcp/IP` アドレス/ポート番号 を利用してデータを送信する点が特徴である．
c_1	2,774,956	10,023	表 1 の説明と同じ．
c_{16}	9,982	9,003	c_{14} と同様の特徴を持つ．
c_{17}	10,118	8,623	php コマンドによって`file_get_contents`関数を呼び出し，外部のサーバへのアクセスを試みる．アクセスする際の URL に攻撃対象のサーバ名が含まれるのが特徴である．
c_{18}	9,449	8,255	c_{14} と同様の特徴を持つ．
c_{19}	7,879	7,456	c_{15} と同様の特徴を持つ．

アドレスを示しており，青いノードは攻撃先を示している．ノード間につながりがある場合は攻撃先において攻撃元 IP アドレスによる攻撃が確認されたことを意味する．ノードは大きいほど攻撃検知数が多いことを示している．

図 2 に示している c_1 のイベントは Nessus による脆弱性調査だと考えられる．脆弱性検査は外部から不正に実施されるものもあるが，正常な診断行為として実施される場合もある．このような診断行為は定期的に特定のホストから実施される場合も多いため，図 2 の攻撃元 IP アドレスと攻撃先の関係はエッジ数が少なく，ノードも一定以上の大きさをもつものが多く，発散していない状態となっている．一方で図 3 は攻撃元 IP アドレスの数が c_1 に比べて極端に多いことがわかる．c_4 はイベント数では c_1 より少ないが，c_4 はワームと見られる動作をする攻撃コードのクラスタとなっており，数多くのホストが攻撃に参加していることが読み取れる．さらに攻撃先についても無差別に対象を選んでいるためか攻撃元 IP アドレスとの間に特定の関係性はなく，全方位的に攻撃が実施されている事が推測できる．

このように同じ脆弱性を利用した攻撃でも攻撃ツールや攻撃者の意図によっても攻撃の振る舞いは大きくことなるものであり，これを発見するためにもクラスタリングを足がかりとして探索的な分析を実施する意味があると言える．

5 今後の課題

5.1 既存クラスタリング手法の適用と比較

本稿ではクラスタを作成する際に独自の手法を利用している．これは最も単純な方法でどの程度の効果がでるかを測る目的があったが，既存クラスタリング手法でも文字列を分類する手法がある．今後，これらの手法と本稿の手法を比較検討していく必要がある．

5.2 他の脆弱性を利用したクラスタリングの有用性検証

また本稿では ShellShock 脆弱性を対象とした攻撃のみについてクラスタリングおよび分析を試みた．これは攻撃コードの意図がくみやす

表 3: 攻撃コード種類数の上位 10 クラスタと正規表現の記述に寄る網羅率

クラスタ	正規表現	一致種類数
c_{11}	Gecko/20100101 Firefox/13.0.1 WhiteHat Security	34,180 (100.0%)
c_{12}	(wget|curl) http://whsec.us/\d{8}/\d+/	28,009 (100.0%)
c_{13}	/bin/bash \-c \'curl http://\d+\.\d+\.\d+\.\d+/search/	24,353 (100.0%)
c_{14}	(wget|curl) http://whsec.us/\d{8}/\d+/[0-9a-f]+	21,335 (100.0%)
c_{15}	> /dev/tcp/[\d\.]+/+\d+; /bin/uname \-a > /dev/tcp/[\d\.]+/+\d+	17,153 (100.0%)
c_{1}	echo \"bash_cve_2014_6271_rce Output : \\$\(\(\d+\+\d+\)\)	10,006 (99.8%)
c_{16}	(wget|curl) http://whsec.us/\d{8}/\d+/[0-9a-f]+	9,003 (100.0%)
c_{17}	/bin/bash -c \"php -r \\\\\"file_get_contents\\(\'http://	8,623 (100.0%)
c_{18}	(wget|curl) http://whsec.us/\d{8}/\d+/[0-9a-f]+	8.225 (100.0%)
c_{19}	> /dev/tcp/[\d\.]+/+\d+; (/bin/uname \-a|echo .+) > /dev/(tcp|udp)/[\d\.]+/+\d+	7,456 (100.0%)

く，攻撃コードのほとんどが ASCII 文字列による bash 命令で構成されているため距離を文字列比較の関数で利用しやすかったなどの理由があげられる．一方で ShellShock 脆弱性を利用した攻撃はコード記述の自由度が高くもともと改変がしやすいという特徴がある．そのため別の脆弱性を利用した攻撃ではまた異なった傾向がでる可能性があり，検証が必要である．

5.3 クラスタからのパターン作成

クラスタリングした攻撃コードを正規表現を用いて表現できることを第 4 節で示したが，本稿では筆者が手作業にて正規表現を記述した．これを自動化することによって新しい攻撃コードが出現した場合でも，一定数のデータを蓄えることで IDS のシグネチャなどのパターンを迅速に作成することができると期待される．これはシステムログのフォーマット自動生成技術などの応用により実現可能であると見込まれる．

6 まとめ

本稿ではセキュリティ機器，特に IDS から出力されるログに着目し，攻撃の傾向を分析するにあたって攻撃コードの類似性に基づくクラスタリング手法について述べた．さらに ShellShock 脆弱性を対象とした攻撃に対してクラスタリング手法を適用し，攻撃ツールに基づくと考えられる分類ができることを示した．これにより攻撃コードを適切かつ機械的に分類することで分析者にかかる負担を軽減しつつ，攻撃者に利用されている攻撃ツールの分類や攻撃傾向の分析などの探索的なデータ分析に役立つことを示した．