Vol.2022-CSEC-97 No.21
Vol.2022-IOT-57 No.21
2022/5/20

ハニーポットで観測されたサイバー攻撃の対象機器及び脆弱性の自動推定手法の提案

九鬼琉[1]　植田岳洋[2]　佐々木貴之[3]　吉岡克成[4]　松本勉[4]

概要：近年，IoT の発展に伴い，IoT 機器への攻撃が活発化している．これらの攻撃実態を把握するため，我々は IoT 機器を模倣したハニーポットを設置し，1 ヶ月あたり平均 55,000 件の攻撃を観測している．ハニーポットで観測される攻撃がどのような脆弱性・機器を狙ったものであるのかを把握することは，攻撃の傾向を把握し適切な対応を行っていくために必要不可欠である．そのため，我々はハニーポットで観測した攻撃に対して攻撃対象の脆弱性や機器の情報を紐付けるためのタグ付けを行っているが，タグ付けを行うためのルール作成はこれまで人手で行ってきた．しかし，1 ヶ月あたり平均 55,000 件という膨大かつ多様な攻撃に対し，それらのタグ付けルールの作成を全て人手で行うことは現実的ではない．本論文では，観測した攻撃に対して低コストかつ網羅的な分析を行うために，観測した攻撃の特徴(シグネチャ)と攻撃対象の脆弱性や機器の情報を紐付けるためのタグ付けルールを自動的に生成する手法を提案する．提案手法は，ハニーポットのイベントログから攻撃の特徴となるシグネチャを生成し，NVD や Exploit Database などの公開脆弱性データベース(VDB)よりシグネチャに関連する情報を自動で収集することにより，シグネチャと攻撃対象の脆弱性や付加情報を紐付けるためのタグ付けルールの生成を行う．提案手法を用いて，人手でタグ付けルールを作成済みの攻撃ログに対し攻撃情報の自動推定及びタグ付けルールの生成を行った結果，83.2%の攻撃に対して正しいルール生成に成功し，誤ったルールの生成は 3.2%であった．

キーワード：ハニーポット，セキュリティ，IoT 機器，攻撃解析

Towards Identification of Targeted Devices and Vulnerabilities from Cyber-attacks Observed by Honeypots

Ryu Kuki[1]　Takahiro Ueda[2]　Takayuki Sasaki[3]　Katsunari Yoshioka[4]　and Tsutomu Matsumoto[4]

Abstract:
With the development of IoT, cyber-attacks against IoT devices have increased. To monitor these attacks, we have operated honeypots that imitate IoT devices and have observed 55,000 attacks per month on average. To understand attack trends and take appropriate measures, it is essential to understand cyber-attacks target which vulnerabilities and devices. We manually created tagging rules and tagged a part of cyber-attacks observed by honeypots with vulnerabilities and target devices. However, it is unrealistic to create tagging rules for many types of attacks observed by the honeypots. In this paper, we propose an automatic tagging rule generation method to link observed attack characteristics (signature) with vulnerabilities and devices, to analyze observed attacks in a low-cost and comprehensive way. This method generates signatures each of which characterizes an attack observed by a honeypot. Then, the method collects information related to the signature from public vulnerability databases (VDB), such as NVD and Exploit Database, and generates tagging rules to associate the signatures with attack information such as CVE numbers and target devices. We implemented the method, and an evaluation result showed that correct tagging rules were successfully generated for 83.2% of the attacks, and incorrect tagging rules were generated only for 3.2% of the attacks.

Keywords: Honeypot, Security, IoT devices, Attack analysis

1. はじめに

近年，Wi-Fi ルータや Web カメラをはじめとした IoT(Internet of Things)機器の普及に伴い，IoT 機器を狙ったサイバー攻撃が活発化している．NICTER の観測レポート 2021 [1]によると，IoT 機器に特徴的なポート番号を狙った通信や大規模なスキャンがサイバー攻撃関連通信の過半数を占めていることが報告されている．

こうした IoT 機器への攻撃実態を把握するため，我々は様々な IoT 機器の応答をインターネット上の機器から収集し模倣する適応型ハニーポットである X-POT を設置し，1 ヶ月あたり平均 55,000 件の攻撃を観測している．しかし，X-POT では IoT 機器の応答をインターネット上の機器から収集し利用しているため，そのままではハニーポットがどのような機器を模擬しているのかをハニーポットの運用側が把握できず，攻撃のみが観測される形となる．そこで，ハニーポットで観測されたこれらの攻撃について，その詳細を分析することにより，いつ，どのような機器・脆弱性を狙った攻撃がどれくらいの規模で行われているのかとい

1 横浜国立大学理工学部 数物・電子情報系学科
2 横浜国立大学大学院 環境情報学府
3 横浜国立大学先端科学高等研究院
4 横浜国立大学大学院環境情報研究院/先端科学高等研究院

Vol.2022-CSEC-97 No.21
Vol.2022-IOT-57 No.21
2022/5/20

った攻撃の実態を可視化することができ，攻撃の傾向の把握やその後の適切な対応へと役立てることができる．そのため，我々はハニーポットで観測された攻撃について，その詳細を調査し，それぞれの攻撃に対して既知の攻撃か未知の攻撃か, そして既知の攻撃であれば攻撃の対象機器や，CVE ID などの脆弱性情報のタグ付け(ラベリング)を行っている．この攻撃の詳細についての調査及びタグ付けのためのルール作成はこれまで手作業により行ってきた．しかし，ハニーポットで日々観測されている攻撃の全容を把握するためには，膨大な観測データに対し迅速かつ網羅的な調査が必要であり，これらの攻撃調査及びタグ付けのためのルールの作成を全て手作業で行うことは分析にかかるコスト・迅速性といった観点から現実的ではない．

そこで本論文では，ハニーポットで観測されたサイバー攻撃について低コストかつ網羅的な分析を行うため，観測した攻撃の特徴を基に対象の機器や脆弱性についての情報を自動的に推定し，推定した情報と攻撃の観測ログの紐付けを行うためのタグ付けルールを生成する手法を提案する．提案手法は，ハニーポットの観測イベントのログから攻撃の特徴となるシグネチャを生成し, NVD や Exploit Database などの公開脆弱性データベース(OSVDB)をはじめとしたインターネット上の公開情報から当該攻撃に関連する情報を収集することにより攻撃の対象機器や脆弱性を自動的に推定し，シグネチャとの紐付けを行うためのタグ付けルールの生成を行う．

提案手法の性能を評価するため， X-POT で観測された 95 件の新規 HTTP リクエストについて，提案手法を用いてタグ付けルールの自動生成を行い，人手によるルール生成結果との比較を行った．その結果，83.2%にあたる 79 件の攻撃に対して正しいルール生成に成功した一方で，誤ったルールの生成は 3.2%にあたる 3 件であった．

2. 研究背景：ハニーポットによる攻撃観測と多角的分析のための統合アーキテクチャ

ハニーポットとは，脆弱なシステムを模したホストを囮として設置することにより，サイバー攻撃を誘引し観測を行うシステムである．その一種として，様々な IoT 機器の応答をインターネット上の機器から収集し，模倣する機器の種類を拡張可能な適応型ハニーポットである X-POT[2]が提案されている．

図 1 は，X-POT とデータ解析のアーキテクチャの全体図である. 当該アーキテクチャはハニーポット, 動的解析部，攻撃タグ付け部から構成されており，ハニーポットで直接観測した攻撃コードと，ハニーポットで収集された検体を動的解析にかけて得られた攻撃リクエストの両者が攻撃タグ付け部に渡される．攻撃タグ付け部では，観測イベントのうち明らかに攻撃であるとわかる通信について既存のタグ付けルールと照合し，既知の攻撃であれば攻撃の対象となっている脆弱性や機器情報のタグ付けを行う．現在は，脆弱性攻撃後にマルウェア本体を感染機器にダウンロードするために用いるコマンドである curl もしくは wget が含まれたリクエストを攻撃とみなしているが，この条件では攻撃の取りこぼしが生じるため, 今後改善する予定である．既存のタグ付けルールにマッチしない未知の攻撃については，その攻撃についての詳細を調査し，タグ付けルールを追加することによりタグ付けを行う．X-POT とデータ解析のアーキテクチャの詳細については論文[3]で述べる．

本論文では，このうちタグ付けルールの生成部分において，公開脆弱性データベースをはじめとしたインターネット上の公開情報をもとに攻撃対象の機器や脆弱性についての情報を推定し，タグ付けルールの自動生成を行う手法を提案する．

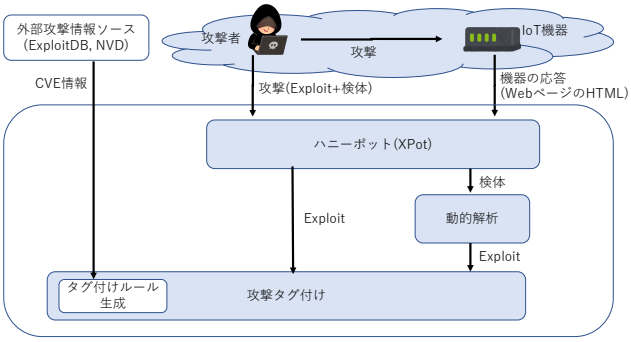

図 1 X-POT とデータ解析アーキテクチャ

3. 関連研究

ハニーポットで収集した攻撃データの分析は多数行われている．論文[4]によってハニーポットとそのデータに関する分析がまとめられている．データを分析する観点は様々であるが，特に攻撃対象に関する分析は多くの研究の興味の対象となっており，その攻撃先のポートとプロトコルの分析を行なった研究[5][6][7][8]，攻撃対象のサービスを分析した研究[9][10]，対象の脆弱性を分析した研究[10][11], ハニーポット等で収集されたマルウェア検体が有する脆弱性攻撃機能を分析した研究[12]などが存在する．

インターネット上の脆弱性情報を攻撃と紐付ける研究として，Exploit Database の情報から CVE 情報を構成するという研究[13]も行われている．この研究[13]では，機械学習ベースの手法を用いることによって，Exploit Database 上にある攻撃情報から CVE 情報を生成することで，攻撃に関する情報をいち早く把握している．

4. 問題定義

ハニーポットでは毎日膨大な件数の攻撃が観測されており，それぞれの攻撃に使用されている脆弱性も様々である．また，日々新しい攻撃手法が現れている．

ハニーポットで観測される攻撃がどのような脆弱性・機

器を対象としたものであるのかを捉えるためには，ハニーポットで観測された HTTP リクエストに対して，攻撃対象とされている機器や脆弱性についての情報を調査し，それらの情報をタグ付けするためのルールを作成する必要がある．

しかし，ハニーポットで日々観測されている膨大な数の攻撃の全てに対して，手作業で脆弱性情報の調査を行いタグ付けのためのルールを作成することは現実的ではない．

タグ付けルールのためのルールを自動で生成することを考えると，以下の技術的な問題が存在する．

問題 1：ルール作成のための脆弱性の情報を，インターネットから効率的に収集する必要がある．

問題 2：誤ったルールの生成を抑える必要がある．

5. 提案手法

ハニーポットで観測されたイベントに関連する情報を NVD や Exploit Database などの公開脆弱性データベースから収集し，攻撃の対象となっている機器や脆弱性を推定することにより，ハニーポットで観測した攻撃に対してタグ付けを行うためのルールの自動生成を行う手法を提案する．脆弱性の情報が集約されたデータベースを利用することで情報収集の効率化を図っており，さらに複数のデータベースを利用することでカバレッジを広げ，問題 1 を解決している．加えて，それぞれのデータベースから得られた脆弱性の情報（例えば CVE 番号）が一致しない場合に，アラートを出力し，人の確認のステップを設けることで問題 2 を解決している．

5.1 提案手法概要

図 2 は今回提案するタグ付けルール自動生成手法の全体図である．

はじめに，ハニーポットで観測されたイベントのうち，攻撃の際にマルウェアのダウンロードのために使用される wget や curl などのコマンドが含まれていて明らかに攻撃であるとわかる HTTP リクエストについて，既存のタグ付けルールにマッチするか否かを判定する．

次に，当該イベントが既存のタグ付けルールにマッチする場合は，タグ付けルールに従って観測ログへのタグ付けを行う．そして既存のタグ付けルールにマッチしない HTTP リクエストに対して，提案手法を用いて攻撃についての情報収集及びタグ付けルールの生成を行う．具体的には，まず，イベントの観測ログから攻撃を特徴づける情報として HTTP リクエストのパス情報を抽出し，それに関連する脆弱性情報の検索を行う．検索対象は公開脆弱性データベースからあらかじめ脆弱性情報を収集した情報を保存している内部データベース，公開脆弱性データベースが提供している検索 API，GitHub のコード検索 API の 3 種とした．これらの検索結果をもとに，観測したイベントに関連する攻撃対象機器や脆弱性を推定する．ここで，内部データベースや外部の各公開脆弱性データベースから得られた脆弱性情報が複数存在する場合，アラートを出し人の目で確認を行った上でタグ付けルールに反映する情報を選択する．これらの情報を元に，攻撃対象機器及び脆弱性情報を観測ログにタグ付けするためのルールの自動生成を行い，タグ付けルールのリストへと追加する．以下では，それぞれのステップについて詳しく述べる．

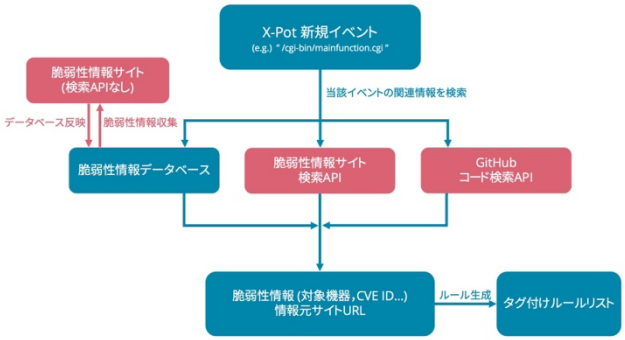

図 1　タグ付けルール自動生成手法 全体図

5.2 公開脆弱性データベースからの脆弱性情報の収集

検索対象の公開脆弱性データベースの選定にあたっては，掲載情報の網羅性や重複を考慮し，NVD[14], Exploit Database[15], Vulners[16], CNVD[17]の 4 サイトを採用した．以下では，それぞれの公開脆弱性データベースからの脆弱性情報の具体的な検索・収集手法について述べる．

NVD 及び Vulners については開発者向けのオープン API が公開されており[18][19]，API を介して脆弱性情報の検索や詳細情報の取得を行うことができる．このオープン API を利用することにより，脆弱性情報の収集を行えるようにした．

Exploit Database については検索用のオープン API は用意されてないものの，Exploit Database を管理している Offensive Security の公式 GitHub リポジトリ[20]において，Exploit Database の Web サイト上に掲載されている脆弱性情報について，その Exploit コード及びシェルコードが公開されている．この GitHub リポジトリ内を GitHub の開発者向け REST API を用いて検索することにより，脆弱性情報の収集を行えるようにした．

CNVD についても検索用のオープン API は用意されてないものの，CNVD の公式 Web サイト上において，新規の脆弱性についての情報が XML ファイル形式で毎週公開されている[21]．このファイルをダウンロードして Python で読み込んで解析し，脆弱性情報の検索に必要な情報を抽出して内部データベースへ格納することにより，脆弱性情報の検索を行えるようにした．

収集した脆弱性情報を格納するための内部データベースのデータ構造を表 1 に示す．収集した脆弱性情報を格納

する内部データベースには，タグ付けに用いる CVE ID や対象機器についての情報(カラム名: id)，情報取得元のサイト URL(カラム名: url)，脆弱性についての解説文や Exploit コード(カラム名: description)，さらに description から正規表現を用いて抽出したリクエストパス情報 (カラム名: path)を保存した．

図 3 は脆弱性についての解説文や Exploit コードである description からリクエストパス情報である path の抽出を行う例である．正規表現を用いて脆弱性の概要文からリクエストパスである”main/calendar/agenda_list.php”を抽出している．

図 3　リクエストパス情報の抽出例

上記の例のように脆弱性についての解説文や Exploit コードからのリクエストパスの抽出を行うことにより，内部データベースから脆弱性情報の検索を行う際にリクエストパス文字列の完全一致による検索を行うことができ，部分一致による検索を行った場合に比べて精度の高い検索を行うことができる．

表 1 内部データベース構造

カラム名	格納データの内容
id	CVE ID / CNVD ID や対象機器情報など，脆弱性情報を識別・特定するタグ． 最終的にタグ付けルールの生成に用いる．
url	脆弱性情報の収集元である公開脆弱性データベースの当該ページ URL
description	脆弱性についての概要や解説文など
path	description から正規表現を用いて抽出したリクエストパス

5.3 タグ付けルールの生成

ハニーポットで観測されたイベントについて，HTTP リクエストのパス情報をもとに攻撃を特徴づけるシグネチャを生成し，攻撃に関連する情報を収集するため，公開脆弱性情報データベースからあらかじめ脆弱性情報を収集した内部データベース，公開脆弱性データベースが提供している検索 API，GitHub のコード検索 API のそれぞれから検索を行った．

これらの検索結果から得られた情報をもとに攻撃対象の機器及び脆弱性の推定を行う．ここで，内部データベースや外部の各公開脆弱性データベースから得られた脆弱性情報が一致しなかった場合，アラートを出し人の目で確認を行った上でタグ付けルールに反映する情報を選択する．これは攻撃を特徴づけるシグネチャに対応する脆弱性情報(CVE ID など)が複数存在しているケースがあり，自動での判断が困難なためである．人の目による確認ステップを設けることにより，誤ったルールが生成されるのを抑えることができる．これらの情報を元に最終的にハニーポットの観測イベントのログに対して攻撃対象機器や脆弱性の CVE ID の紐付けを行うためのタグ付けルールの自動生成を行った．

生成したタグ付けルールは以下のような YAML 形式で記載される．

```
    condition:
    - /dnslookup.cgi
    memo: https://cve.mitre.org/cgi-bin/cvename.cg
i?name=CVE-2017-6334
    tag: CVE-2017-6334
```

condition に HTTP リクエストのパス情報を始めとした攻撃の特徴となるシグネチャを記載し，tag に攻撃対象機器や脆弱性の CVE ID を記載する．さらに，memo には脆弱性情報の収集元として公開脆弱性データベースの当該ページの URL を記載する．攻撃を特徴づけるシグネチャに対応する脆弱性が複数存在する場合は，それぞれの脆弱性に対応する攻撃対象機器や脆弱性の CVE ID がカンマ区切りで tag に列挙される．

作成したタグ付けルールを用いて観測ログへのタグ付けを行う際は，condition に指定された文字列が HTTP リクエスト中に含まれるか否かで判定が行われ，含まれる場合には tag に指定された文字列がタグとして付与される．

6. 評価

6.1 評価手順

攻撃対象の機器及び脆弱性の自動推定手法について，その性能を評価するため人手での調査結果と比較することにより評価を行った．はじめに，X-POT や研究室で運用している他のハニーポット(IoT/Telnet ハニーポット，複数プロトコル対応ハニーポット)で観測された 95 件の新規 HTTP リクエストについて，あらかじめ手作業による脆弱性情報の調査を行い，関連する脆弱性情報の自動推定を行った．

次に，手作業での調査によってタグ付けを行ったデータセットに対して，今回の提案手法を用いて攻撃対象の機器及び脆弱性の自動推定を行い，脆弱性情報の合致件数，及び誤ったルールを生成してしまった誤生成件数の集計を行った．

評価基準として，今回の提案手法を用いて推定された脆弱性情報の候補の中に手作業での調査によってタグ付けされた脆弱性情報が含まれており，手作業での調査と同様のタグ付けルールが生成できた場合を「成功」，手作業での調査とは異なる脆弱性情報のみが推定された場合を「失敗(誤検知)」，攻撃に関連する脆弱性情報を収集することができ

ずタグ付けルールの生成が行えなかった場合を「失敗(情報なし)」とした.

6.2 評価結果

評価手順に従いタグ付けルールの自動生成を行い，手作業での調査によるタグ付けとの比較を行ったところ，X-POT で観測された 95 件の新規 HTTP リクエストのうち約 83%にあたる 79 件について, 提案手法を用いて推定された脆弱性情報の候補の中に手作業での調査結果と同じ脆弱性情報が含まれており，正しいタグ付けルールの生成を行うことができた(「成功」). 情報元サイト別の集計では，NVD では 32 件，Exploit Database では 37 件, Vulners では 75 件, CNVD では 16 件の脆弱性情報が得られた．(詳しくは，付録の表 2 参照)

一方で，全体の 3%にあたる 3 件について，手作業での調査結果とは異なる結果が出力された(「失敗(誤検知)」). この 3 件について，攻撃を特徴づけるシグネチャとして使用した HTTP リクエストのパス情報を以下に示す.

```
/cgi-bin/luci/
/cgi-bin/;
/shell?
```

また，全体の 14%にあたる 13 件について，攻撃に関連する脆弱性情報を収集することができずタグ付けルールの生成が行えなかった(「失敗(情報なし)」). この 13 件について，攻撃を特徴づけるシグネチャとして使用した HTTP リクエストのパス情報を以下に示す.

```
/cdn-cgi/l/chk_captcha
/cgi-bin/weblogin.cgi?
/linuxki*/experimental/vis/ki443vis.php?type=
/portal/__ajax_explorer.sgi?action
/kylin/api/cubes/kylin_streaming_cube
/cgi/setPwd?pwd
/simplecfg.xml
tmBlock.cgi
hndBlock.cgi
/picdesc.xml
/v1.24/containers/create
/manager_dev_ping_t.gch
/login.gch
```

7. 考察

7.1 課題

手作業での攻撃情報の調査によるタグ付けルール作成結果との比較を行い，攻撃情報の自動推定手法の性能評価を行ったところ，前述の通り 8 割強のイベントについて手作業によるタグ付けルール作成と合致するルールを生成することができた．一方で，全体の 3.16%にあたる 3 件について，手作業によるタグ付けルール作成結果とは異なる脆弱性情報に紐づくタグ付けルールが生成された

手作業によるタグ付けルール作成結果とは異なるタグ付けルールが自動生成された一因として，攻撃情報を特徴づけるシグネチャ生成の不適切さが挙げられる．シグネチャは原則として HTTP リクエストのパス情報をもとに生成されるが，手作業によるタグ付けルール作成結果とは異なるタグ付けルールが自動生成された 3 件についてはいずれもリクエストパスが ”/shell?” のように短いものであったことで，リクエストパス情報だけでは正確な攻撃情報の特定ができなかった可能性が考えられる．このような誤生成を減らすためには，リクエストパス情報に加えて HTTP リクエストのクエリ文字列(パラメータ)等の付加情報を合わせて攻撃情報を特徴づける情報としてシグネチャとみなすことにより，シグネチャに関連する攻撃情報の件数を絞り込み，こうした謝検知の割合を減らすことができると考えられる．また，実際に提案手法を用いて運用を行う際は自動生成されたタグ付けルールをそのままタグ付けルールのリストへ追加するのではなく，タグ付けルールのリストへの追加前に人の目による確認プロセスを設けているため，誤った脆弱性情報が推定された場合でもその誤った情報が観測ログへタグ付けされることを防ぐことができる.

また，今回の提案手法では脆弱性情報の検索対象として代表的な公開脆弱性データベース 4 サイトを採用したが，脆弱性情報の網羅性を向上させるため，これら以外の公開脆弱性データベースを情報収集対象として追加することを検討する必要がある.

さらに，新しく発見された脆弱性情報は，今回検索対象とした公開脆弱性データベース以外に製品開発ベンダーの公式サイトや個人の脆弱性発見者が運営するブログにおいて公開されることがある．これらは公開脆弱性データベースのように書式が統一されておらず自動化による脆弱性情報の収集の難易度が高いという課題があるものの，これらのサイトから脆弱性情報を収集し，観測した攻撃についての調査に反映させることにより，特に発見されてから日が浅い新しい脆弱性を利用した攻撃について，より高い精度でタグ付けルールの自動生成ができるようになることが期待される.

また, 今回は X-POT で観測した HTTP リクエストについて攻撃情報の調査を行ったが，シグネチャの生成などを工夫することにより他のプロトコルでの観測データの分析においても当手法を適用することができると考えられる.

7.2 提案手法の有用性

提案手法を用いることにより，ハニーポットで観測したサイバー攻撃の分析において，手作業によって攻撃対象の機器や脆弱性についての調査，及び観測ログへのタグ付けルール作成を行ったイベントのうち 8 割強について，攻撃情報の自動推定を行い，タグ付けルールを生成することができた．これにより，手作業で攻撃の分析を行っていた従来よりも多くの攻撃についてその詳細を分析することができるようになる．さらに，タグ付けルールの生成時に情報元となった Web ページの URL を併記することにより，後から人の目でも容易に攻撃についての関連情報を確認することができるようになった.

したがって，提案手法を用いることで従来よりも低コストかつ網羅的に，ハニーポットで観測したサイバー攻撃の分析を行うことができるといえる．

8. 結論

本論文では，ハニーポットで観測したサイバー攻撃について網羅的な分析を行うために，インターネット上の公開脆弱性情報をもとに観測した攻撃の特徴と攻撃対象の脆弱性や機器の情報を紐付けるためのタグ付けルールを自動的に生成する手法を提案した．提案手法を用いて評価実験を行った結果，手作業での攻撃情報の調査によって作成したタグ付けルールのうち約 8 割強のイベントについて攻撃対象の脆弱性や機器の情報を推定し，タグ付けルールを生成することができた．今後は，情報元となるサイトを追加することでより高い精度での自動生成を行えるよう改良を行いつつ，ハニーポットで観測したサイバー攻撃の分析において本手法を活用していく予定である．

謝辞 本研究は総務省の「電波資源拡大のための研究開発(JPJ000254)」における委託研究「電波の有効利用のための IoT マルウェア無害化／無機能化技術等に関する研究開発」によって実施した成果を含む．

参考文献

[1] 国立研究開発法人 情報通信研究機構 “NICTER 観測レポート 2021”. NICT-情報通信研究機構 (オンライン). https://www.nict.go.jp/cyber/report/NICTER_report_2021.pdf. [アクセス日: 2022/03/16]

[2] Seiya Kato, Rui Tanabe, Katsunari Yoshioka, Tsutomu Matsumoto, "Adaptive Observation of Emerging Cyber Attacks targeting Various IoT Devices," IFIP/IEEE International Symposium on Integrated Network Management (IM), 2021.

[3] 佐々木貴之, 九鬼琉, 植田岳洋, 鮫嶋海地, Guo Binnan, 市川詩恩, 山口陽平, 岡田晃市郎, 吉岡克成, 松本勉, "ハニーポットによる攻撃観測と多角的分析のための統合アーキテクチャの提案", 第 97 回 CSEC 研究発表会, 2022

[4] Nawrocki, Marcin, Matthias, Wählisch, Thomas C., Schmidt, Christian, Keil, and Jochen, Schönfelder. "A Survey on Honeypot Software and Data Analysis." (2016).

[5] F. Pouget, M. Dacier *et al.*, “Honeypot-based forensics,” in *AusCERT Asia Pacific Information Technology Security Con- ference*, 2004.

[6] J. Francois, O. Festor *et al.*, “Activity monitoring for large honeynets and network telescopes,” *International Journal on Advances in Systems and Measurements*, vol. 1, no. 1, pp. 1– 13, 2008.

[7] S. Almotairi, A. Clark, G. Mohay, and J. Zimmermann, “Characterization of attackers’ activities in honeypot traffic using principal component analysis,” in *Network and Parallel Computing, 2008. NPC 2008. IFIP International Conference on*. IEEE, 2008, pp. 147–154.

[8] O.Thonnardand, M.Dacier,“A framework for attack patterns’ discovery in honeynet data,” *digital investigation*, vol. 5, pp. S128–S139, 2008.

[9] R. McGrew, “Experiences with honeypot systems: Development, deployment, and analysis,” in *System Sciences, 2006. HICSS’06. Proceedings of the 39th Annual Hawaii Interna- tional Conference on*, vol. 9. IEEE, 2006, pp. 220a–220a.

[10] E. Alata, V. Nicomette, M. Dacier, M. Herrb *et al.*, “Lessons learned from the deployment of a high-interaction honeypot,” *arXiv preprint arXiv:0704.0858*, 2007.

[11] G. Portokalidis, A. Slowinska, and H. Bos, “Argos: An emulator for fingerprinting zero-day attacks for advertised honeypots with automatic signature generation,” in *Proceedings of the 1st ACM SIGOPS/EuroSys European Conference on Computer Systems 2006*, ser. EuroSys ’06. New York, NY, USA: ACM, 2006, pp. 15–27. [Online].

[12] Arwa Abdulkarim Al Alsadi, Kaichi Sameshima, Jakob Bleier, Katsunari Yoshioka, Martina Lindorfer, Michel van Eeten, Carlos H. Ganan, "No Spring Chicken: Quantifying the Lifespan of Exploits in IoT Malware Using Static and Dynamic Analysis," The 17th ACM ASIA Conference on Computer and Communications Security (ACM ASIACCS 2022), 2022.

[13] Sun, Jiamou, Zhenchang, Xing, Hao, Guo, Deheng, Ye, Xiaohong, Li, Xiwei, Xu, and Liming, Zhu. "Generating Informative CVE Description From ExploitDB Posts by Extractive Summarization." (2021)

[14] NVD, National Institute of Standards and Technology (オンライン). https://nvd.nist.gov/. [アクセス日: 2022-03-10]

[15] Exploit Database, Offensive Security (オンライン), https://www.exploit-db.com/. [アクセス日: 2022-03-10]

[16] Vulners, Vulners (オンライン). https://vulners.com/. [アクセス日: 2022-03-10]

[17] CNVD, 国家计算机网络应急技术处理协调中心 (オンライン). https://www.cnvd.org.cn/. [アクセス日: 2022-03-10]

[18] API Vulnerabilities - NVD, National Institute of Standards and Technology (オンライン). https://nvd.nist.gov/developers/vulnerabilities. [アクセス日: 2022-03-10]

[19] API reference - Vulners wiki, Vulners (オンライン). https://docs.vulners.com/API_wrapper/api/. [アクセス日: 2022-03-10]

[20] offensive-security/exploitdb: The official Exploit Database repository, GitHub (オンライン). https://github.com/offensive-security/exploitdb. [アクセス日: 2022-03-10]

[21] 共享漏洞列表, 国家计算机网络应急技术处理协调中心 (オンライン). https://www.cnvd.org.cn/shareData/list. [アクセス日: 2022-03-13]

Vol.2022-CSEC-97 No.21
Vol.2022-IOT-57 No.21
2022/5/20

付録

表 2　情報元サイト別の脆弱性情報ヒット結果

シグネチャ	NVD	Exploit-DB	Vulners	CNVD	タグ付けルール生成
/cgi-bin/nobody/Search.cgi	情報なし	情報なし	脆弱性情報ヒット	情報なし	成功
/cgi-bin/supervisor/CloudSetup.cgi	情報なし	情報なし	脆弱性情報ヒット	情報なし	成功
/language/Swedish	情報なし	脆弱性情報ヒット	脆弱性情報ヒット	情報なし	成功
/cdn-cgi/l/chk_captcha	情報なし	情報なし	情報なし	情報なし	情報なし
clntudp_call	脆弱性情報ヒット	情報なし	情報なし	脆弱性情報ヒット	成功
wls-wsat/CoordinatorPortType	情報なし	情報なし	脆弱性情報ヒット	情報なし	成功
/cgi-bin/mainfunction.cgi	脆弱性情報ヒット		脆弱性情報ヒット	脆弱性情報ヒット	成功
/_search?pretty	情報なし	脆弱性情報ヒット	脆弱性情報ヒット	情報なし	成功
/console/images/%252E%252E%252Fconsole.portal	情報なし	情報なし	脆弱性情報ヒット	情報なし	成功
/RPC2 methodCall	情報なし	脆弱性情報ヒット	脆弱性情報ヒット		成功
/apply_sec.cgi ping	脆弱性情報ヒット	情報なし	脆弱性情報ヒット	情報なし	成功
/cgi-bin/weblogin.cgi?	情報なし		情報なし	情報なし	情報なし
/linuxki*/experimental/vis/ki443vis.php?type=	情報なし	情報なし	情報なし	情報なし	情報なし
/ping.cgi?pingIpAddress=	脆弱性情報ヒット	脆弱性情報ヒット	脆弱性情報ヒット	情報なし	成功
/portal/__ajax_explorer.sgi?action	情報なし	情報なし	情報なし	情報なし	情報なし
/kylin/api/cubes/kylin_streaming_cube	情報なし	情報なし	情報なし	情報なし	情報なし
/boafrm/formSysCmd	脆弱性情報ヒット	情報なし	脆弱性情報ヒット	脆弱性情報ヒット	成功
/action.php	脆弱性情報ヒット	誤検知	脆弱性情報ヒット	誤検知	成功
/ajax/render/widget_tabbedcontainer_tab_panel	脆弱性情報ヒット	脆弱性情報ヒット	脆弱性情報ヒット	情報なし	成功
/cgi/setPwd?pwd	情報なし	情報なし	情報なし	情報なし	情報なし
/status.cgi?cmd	誤検知	脆弱性情報ヒット	脆弱性情報ヒット	誤検知	成功
/manager?action=	脆弱性情報ヒット	脆弱性情報ヒット	脆弱性情報ヒット	情報なし	成功
/tools/ajax/ConsoleResult.html	情報なし	情報なし	脆弱性情報ヒット	情報なし	成功
/ws/rest/v1/concept	情報なし	脆弱性情報ヒット	脆弱性情報ヒット	情報なし	成功
/index.php?s=captcha	情報なし	脆弱性情報ヒット	脆弱性情報ヒット	情報なし	成功
/mnt_ping.cgi	脆弱性情報ヒット	情報なし	脆弱性情報ヒット	脆弱性情報ヒット	成功
/cgi-bin/file_transfer.cgi?cmd=	情報なし	情報なし	脆弱性情報ヒット	情報なし	成功
/cgi-bin/file_transfer.cgi?file_transfer=new&dir=	情報なし	情報なし	脆弱性情報ヒット	情報なし	成功
/wanipcn.xml	情報なし	脆弱性情報ヒット	脆弱性情報ヒット	情報なし	成功
protocol.csp?function=set&fname=security&opt=mac_	脆弱性情報ヒット	脆弱性情報ヒット	脆弱性情報ヒット	情報なし	成功
cgi-bin/ViewLog.asp	情報なし	情報なし	脆弱性情報ヒット	情報なし	成功
invokefunction&function=call_user_func_array	脆弱性情報ヒット	脆弱性情報ヒット	脆弱性情報ヒット	脆弱性情報ヒット	成功
/picsdesc.xml	情報なし	脆弱性情報ヒット	脆弱性情報ヒット	情報なし	成功
/GponForm/diag_Form?style/	脆弱性情報ヒット	情報なし	脆弱性情報ヒット	情報なし	成功
/GponForm/diag_Form?images/	脆弱性情報ヒット	情報なし	脆弱性情報ヒット	脆弱性情報ヒット	成功
/ctrlt/DeviceUpgrade_1	情報なし	脆弱性情報ヒット	脆弱性情報ヒット	情報なし	成功
/card_scan_decoder.php	脆弱性情報ヒット	情報なし	脆弱性情報ヒット	情報なし	成功
/maker/snwrite.cgi	情報なし	脆弱性情報ヒット	脆弱性情報ヒット	情報なし	成功
soap.cgi?service=WANIPConn1	脆弱性情報ヒット	脆弱性情報ヒット	脆弱性情報ヒット	脆弱性情報ヒット	成功
/cgi-bin-igd/netcore_set.cgi	情報なし	情報なし	脆弱性情報ヒット	情報なし	成功
/simplecfg.xml	情報なし	情報なし	情報なし	情報なし	情報なし
/Main_Analysis_Content.asp	脆弱性情報ヒット	情報なし	脆弱性情報ヒット	脆弱性情報ヒット	成功
/cgi-bin/script?	脆弱性情報ヒット	情報なし	脆弱性情報ヒット	情報なし	成功
/cgi-bin/luci/	誤検知	情報なし	誤検知	誤検知	誤検知
/stainfo.cgi?	脆弱性情報ヒット	脆弱性情報ヒット	脆弱性情報ヒット	情報なし	成功
/setup.cgi?	脆弱性情報ヒット	情報なし	誤検知	誤検知	成功
/cgi action=sendPasswordEmail&user_name=		情報なし	脆弱性情報ヒット	情報なし	成功
boaform/admin/formPing	誤検知	脆弱性情報ヒット	脆弱性情報ヒット	情報なし	成功
/boaform/admin/formLogin?	情報なし	脆弱性情報ヒット	脆弱性情報ヒット	情報なし	成功
/wp-content/plugins/dzs-videogallery/img.php	情報なし	情報なし	脆弱性情報ヒット	情報なし	成功
/softnas/snserver/snserv.php	情報なし	脆弱性情報ヒット	脆弱性情報ヒット	情報なし	成功
mg3500\nmerlin	脆弱性情報ヒット	脆弱性情報ヒット	脆弱性情報ヒット	情報なし	成功
krashrpt.php	脆弱性情報ヒット	脆弱性情報ヒット	脆弱性情報ヒット	脆弱性情報ヒット	成功
AsyncResponseServiceHttps	情報なし	情報なし	脆弱性情報ヒット	情報なし	成功
set_ftp.cgi	脆弱性情報ヒット	情報なし	脆弱性情報ヒット	脆弱性情報ヒット	成功
testaction.cgi	脆弱性情報ヒット	脆弱性情報ヒット	脆弱性情報ヒット	情報なし	成功
aggrecate_js.cgi	脆弱性情報ヒット	情報なし	脆弱性情報ヒット	脆弱性情報ヒット	成功
getThumbnail	脆弱性情報ヒット	脆弱性情報ヒット	脆弱性情報ヒット	脆弱性情報ヒット	成功
upgrade_handle.php	脆弱性情報ヒット	情報なし	脆弱性情報ヒット	情報なし	成功
diagnostic.cgi	脆弱性情報ヒット	脆弱性情報ヒット	脆弱性情報ヒット	情報なし	成功
kerbynet	脆弱性情報ヒット	情報なし	脆弱性情報ヒット	脆弱性情報ヒット	成功
track_import_export.php	脆弱性情報ヒット	情報なし	脆弱性情報ヒット	脆弱性情報ヒット	成功
ajaxPortal.lua	情報なし	情報なし	脆弱性情報ヒット	情報なし	成功
web_shell_cmd.gch	脆弱性情報ヒット	情報なし	情報なし	情報なし	成功
/cgi-bin/;	誤検知	誤検知	誤検知	誤検知	誤検知
/HNAP1/	脆弱性情報ヒット	脆弱性情報ヒット	脆弱性情報ヒット	脆弱性情報ヒット	成功
/login.cgi?cli=	情報なし	情報なし	脆弱性情報ヒット	情報なし	成功
goform/mp	情報なし	脆弱性情報ヒット	脆弱性情報ヒット	情報なし	成功
/UD/act	情報なし	脆弱性情報ヒット	脆弱性情報ヒット	情報なし	成功
/UD/?	情報なし	脆弱性情報ヒット	情報なし	情報なし	成功
usbinteract.cgi	情報なし	脆弱性情報ヒット	脆弱性情報ヒット	情報なし	成功
/ws/v1/cluster/apps	情報なし	脆弱性情報ヒット	脆弱性情報ヒット	情報なし	成功
/api/backup/logout.cgi?sid=	情報なし	情報なし	脆弱性情報ヒット	情報なし	成功
tmUnblock.cgi	情報なし	情報なし	脆弱性情報ヒット	情報なし	成功
tmBlock.cgi	情報なし	情報なし	情報なし	情報なし	情報なし
hndBlock.cgi	情報なし	情報なし	情報なし	情報なし	情報なし
hndUnblock.cgi	情報なし	情報なし	脆弱性情報ヒット	情報なし	成功
/shell?	誤検知	誤検知	誤検知	誤検知	誤検知
/setup.cgi?next_file=	情報なし	情報なし	脆弱性情報ヒット	情報なし	成功
/upgrade_check.cgi	情報なし	情報なし	脆弱性情報ヒット	情報なし	成功
/cgi-bin/cgi_system?cmd	脆弱性情報ヒット	情報なし	脆弱性情報ヒット	脆弱性情報ヒット	成功
/webadmin/script command	情報なし	脆弱性情報ヒット	脆弱性情報ヒット	情報なし	成功
/boaform/admin/formTracert	情報なし	脆弱性情報ヒット	脆弱性情報ヒット	情報なし	成功
/picdesc.xml	情報なし	情報なし	情報なし	情報なし	情報なし
/wanipcn.xml	情報なし	脆弱性情報ヒット	脆弱性情報ヒット	情報なし	成功
/index.php?plot	情報なし	脆弱性情報ヒット	脆弱性情報ヒット	情報なし	成功
<methodName>set_time_config</methodName>	情報なし	情報なし	脆弱性情報ヒット	情報なし	成功
/spywall/timeConfig.php	情報なし	情報なし	脆弱性情報ヒット	情報なし	成功
/v1.24/containers/create	情報なし	情報なし	情報なし	情報なし	情報なし
/manager_dev_ping_t.gch	情報なし	情報なし	情報なし	情報なし	情報なし
/board.cgi?cmd=	情報なし	情報なし	脆弱性情報ヒット	情報なし	成功
rdfs.cgi	情報なし	脆弱性情報ヒット	脆弱性情報ヒット	情報なし	成功
/actionHandler/ajax_network_diagnostic_tools.php	情報なし	情報なし	脆弱性情報ヒット	情報なし	成功
/login.gch	情報なし	情報なし	情報なし	情報なし	情報なし
/getpage.gch	情報なし	脆弱性情報ヒット	脆弱性情報ヒット	情報なし	成功
	32	37	75	16	
	33.68%	38.95%	78.95%	16.84%	