東京大学
情報理工学系研究科 電子情報学専攻
修士論文

ダークネットで観測されるパケットの傾向変化の検出

48-206450
古田 陸太
Rikuta Furuta

指導教員　江崎浩 教授

2022 年 1 月

概要

Abstract

インターネットからアクセス可能であるが, 何のサービスも実行されていないアドレス空間はダークネットと呼ばれる. ダークネットに向けられた正規の通信は存在しないため, ここで受信したパケットはすべて不正なパケットであると考えられる. それらを分析することは現在起こっているサイバー攻撃の把握に有用であると考えられるが, ダークネット上では数多くのパケットが観測されるため, キャプチャしたデータから攻撃に関する具体的な情報を得ることは困難である. 我々は, ダークネットで観測されるデータから新たなインシデントの発生を Auto Encoder で自動的に検出することを試みた. 我々の実験では, 専門家が指摘したものを含む多くの攻撃をダークネット上から自動で抽出することに成功した.

Abstract

Abstract

A darknet is an Internet space that is accessible from the Internet, and where no services are running. Since there are no legitimate packet destined for the darknet, any packets observed on the darknet can be assumed to be illegitimate packets. And to analyze those are believed to be useful to know the real trend of cyber attacks today. Though there must be packets related to serious attacks such as 0day threats, it is difficult to pick up important packets from the darknet as so many packets are observed everyday. We attempted to automatically detect the occurrence of a new incident from the data observed in the darknet using Auto Encoder. In our experiments, we succeeded in automatically extracting many attacks from the darknet. Some of attacks we found are not pointed out by experts.

目次

図目次

表目次

第 1 章

序論

1.1 本研究の背景

未知の脆弱性は日々発見され続けており, その数は年々増加している.2013 年の 1 年間には 5000 件程度の Common Vulnerabilities and Exposures (CVE) が報告されていたが, 2021 年には合計 20000 を超える CVE が報告された. この背景にはインターネットの普及や IoT の進歩など様々な要因があるとされているが, 何れにせよこれらの拡大とともにサイバー攻撃の脅威は年々増している [1]. サイバーセキュリティにおいては, 日々変化するサイバー攻撃の手法に適応することが重要であるが, 新しく出現したゼロデイ攻撃に迅速に把握することは困難である. したがって, ネットワークのモニタリング等の手法を用いて現在行われているサイバー攻撃を明らかにする試みが行われてきた. しかしながら, ネットワークの監視においては日々膨大な量のパケットが観測される. また, パケットの意味を解釈し, どのような攻撃を受けているのかを理解するのは専門的な知識がなければ難しい.

1.2 本研究の目的

ダークネットとは, Network Telescope とも呼ばれる, IP アドレスが割り振られ経路広告はなされているがなんらサービスが稼働していないインターネット空間である [2]. 図 1.1 にダークネットの構造の概要を示す. ダークネットは未使用の IP アドレスであるため, 本来はダークネットに宛てたパケットは観測されないはずである. しかし実際には多くのパケットが観測され, その数は 1IP アドレスあたりの年間の観測パケット数が 79 万パケットを超える [3]. ダークネットに届く通信は, 設定ミスにより意図した接続先と異なる接続先に通信を試みた機器が送ってきたものもあるが, ネットワークに接続された脆弱なデバイスを探すためのスキャン行為や, ワームやマルウェアによるスキャン, (D)DoS 攻撃のバックスキャッタもまた観測される. これらのパケットは, 今現在まさに発生しているサイバー攻撃を捉えたものであり, これらのパケットを観測し検知することはインターネット全体におけるサイバー攻撃の現状を把握する上で有用である [4]. 本研究では, ゼロデイ攻撃の検出に特に着目し, ダークネットにおいて未知の攻撃を既知の攻撃から特異的に分離することで ゼロデイ攻撃の検出を助けるような異

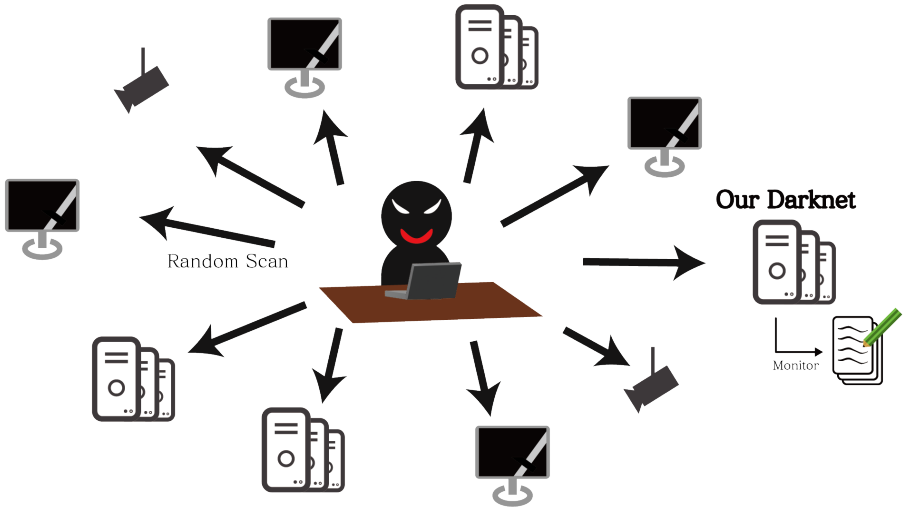

図 1.1: ダークネットの概要

常検知モデルを構築することを目標とする.

1.3　本論文の構成

本論文は以下の内容からなる.
まず第 2 章では, 本研究の前提となるダークネットと, そこで観測されるパケットの概要について述べる. 第 3 章では, ダークネットの分析に関する先行研究やパケットの異常検知に関する先行研究について述べる. 4 章では, 提案手法について述べる. 5 章では実験とその結果を述べ, 第 6 章で結果に関する評価を行う. 第 7 章で提案手法に関する考察を述べ, 第 8 章では本研究のまとめと今後の課題について述べる.

第 2 章

関連技術

本章では, まずどのようにダークネットが構築されてパケットが収集されたかについて述べ, 次に観測されるパケットについての概観を述べる.

2.1 ダークネットの概要

ダークネットを構成するサーバーの挙動を図 2.1 に示す. インターネットに接続された通常のサーバーやデバイスは, サービスが稼働しているポートに対し他のデバイスからアクセスがあった場合, アクセス元に対して SYN-ACK パケットなどを用いて返答を行う. 一方, サービスが稼働していないポートにアクセスがあった場合, TCP パケットに対しては RST パケットや FIN パケットを用いて返答するか, 何もしないのが一般的である. また, 電源が入っていなかったり故障しているなどの理由でサーバー上のサービスが停止している場合も, アクセスされたサーバーはいかなるパケットも返さない. いずれにせよ, サーバーに対しパケットを送信したが, 何も起こらなかったというのはインターネット上で頻繁に発生する事象である.
ダークネットになんらかのデバイスがアクセスを試みた際, ダークネットはアクセス元に対し何のパケットも返送しない. しかしながら, ダークネットは送られてきたパケットを裏で記録している. ダークネットは何のパケットも返送しないので, アクセスを試みたデバイスにとっては, アクセス先がダークネットであるのか, それとも応答を返さない一般的なサーバーであるのか区別がつかない. この攻撃側がアクセスしてくるのを待つというダークネットの特性上, ダークネットを構成する具体的な IP アドレスは公開されない. ダークネットの IP アドレスが攻撃者に知られた場合, 攻撃側がそこを意図的に回避する可能性が高く, サイバー攻撃の把握が困難になるためである. 文献によっては, Tor などを用いてアクセスできる一部のネットワークをダークネットと呼ぶこともあるが, 本論文ではこれは扱わない.

2.1.1 ダークネットのネットワーク構造

ダークネットを運用する上では, サービスが何も稼働していないサーバーに IP アドレスを割り当てる必要がある. ダークネットの運用のために用いる IP アドレスの数が多ければ多い

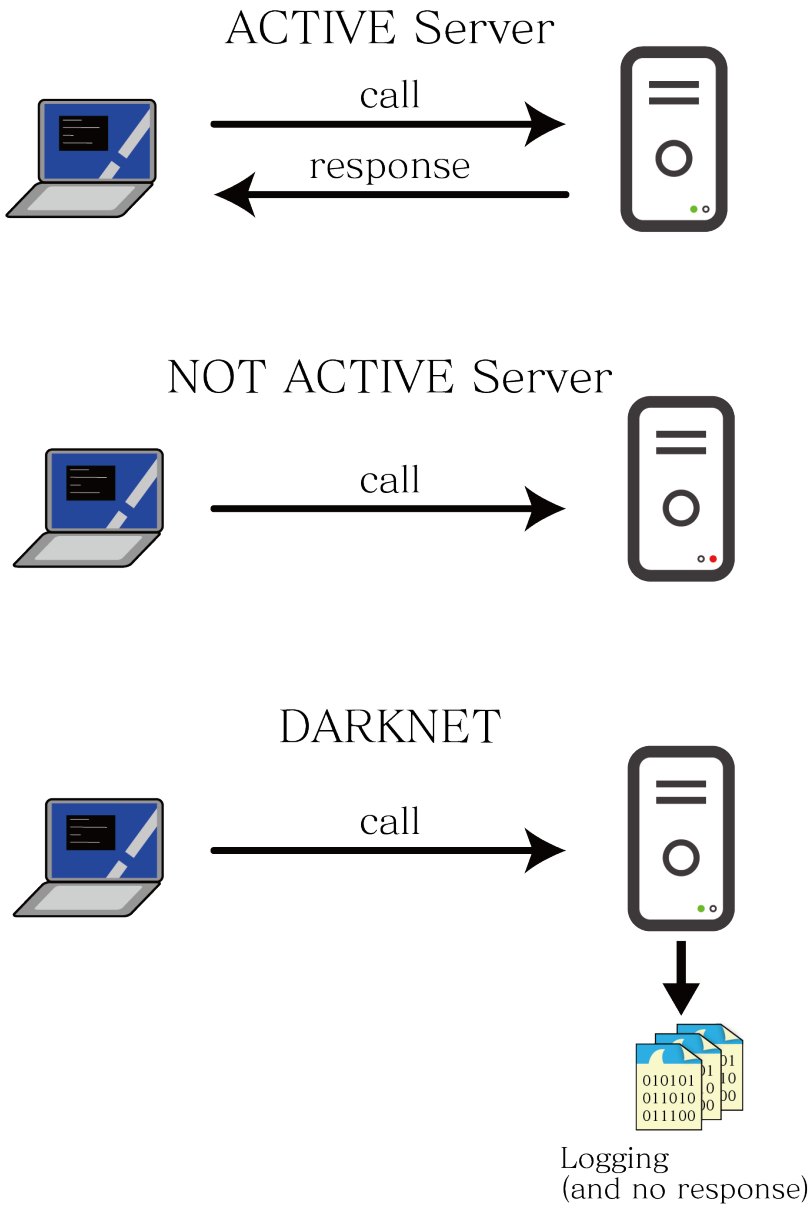

図 2.1: ダークネットの挙動

ほど, より多くのサイバー攻撃の兆候を効率的に観測することが可能であるため [2], なるべく多くの IP アドレスをダークネットに割り当てることが望ましい. 一方、IPv4 アドレスは全世界でわずか 2^{32} = 約 43 億個しか存在せず, 枯渇が問題視されている [5]. また, ダークネットとして使用される IP アドレスの上では何も動いていない必要があるため, 他のウェブサービスなどと IP アドレスを共有して構築することも困難である. ダークネットは, 組織において余っている IP アドレスを用いて構成されることが多いが, IP アドレスが全世界的に枯渇する中で組織内においては余り続けているという状況は少なく, 結果として長期に渡り運用されているダークネットは限られている. 通常の研究においては, ダークネットの運用者から提供されたパケットデータを用いて分析を行うことが多い.

NICT では必ずしも連続していない未使用の IP アドレスに届くパケットを収集している [6]. CAIDA では/8 の IP アドレスをダークネットとして設置している [7].
[8] は, クラウドコンピューティングサービスである Amazon Web Service を用いて分散した IP アドレス郡を用いてダークネットを作成し, 1 つのアドレスブロックを用いたダークネットと観測できるデータを比較した. また, [9] もオランダ, ブラジル, イタリアの 3 ヶ国にダークネットを設置し 1 ヶ月に渡ってデータを収集した.
[10] は, /18 のネットワークブロックを構成する 16,384 個の IP アドレスを用いてダークネットを作成し, そのうち 32 箇所に脆弱な MySQL を模したハニーポットを設置した. この研究では, ハニーポットが脆弱なバージョン応答を返す場合, 応答アドレスを含む/24 のダークネットの IP アドレスをシーケンシャルにスキャンする例が確認された. これは, ダークネットの近傍で稼働するサービスによってダークネットで観測されるデータが変化する可能性を示している.
ダークネットは IPv4 アドレスを用いて構築することが一般的であるが, IPv6 アドレスを用いてダークネットを構築し IPv6 空間上における攻撃活動の把握を目指した研究も存在する [11][12]. IPv4 で構成されたダークネットと異なり, IPv6 で構築されたダークネットで観測されたパケットの大半は ICMP プロトコルによるものであり, 設定ミスに起因するものであった [13].

2.2 ダークネットで観測されるパケット

2.2.1 観測されるパケットの概要

ダークネットでは, 主に TCP, UDP, ICMP のパケットが観測され, その中でも TCP パケットが最も多く観測されることが知られている [14]. ダークネットに届く通信には, 主に 3 種類存在すると考えられている [15].
1 種類目は, マルウェアやボット, あるいは人手による, ネットワーク上のデバイスを探すことを目的としたスキャン行為である. 特に, インターネット経由で不正な任意コード実行を行える脆弱性が知られているシステムや簡便なパスワードでログインできるデバイス, 反射型 DDoS 攻撃に利用できるサービス [16] などが探索されることが多い. スキャン行為は, その第一段階として特定のポートに向けて通信を行い, 応答が返ってくるかどうかを確認することでそのポートが開いていて接続可能であるかどうかを判断する. TCP でこの活動を行う際には, $SYN \cap FIN \cap FIN - ACK \cap NULL$ の Flag が付与される [17]. ダークネットは一切の応答を返さないので, 攻撃側からはポートが空いていないように見える.
2 種類目のトラフィックは, (D)DoS 攻撃のバックスキャッタ呼ばれるパケットである. (D)DoS 攻撃とは、攻撃の対象となるサービスに対して対象の処理 能力の限界を超える量のパケットを送りつけることで対象をシステムダウンさせる攻撃である. この攻撃においては, 攻撃者としては対象がひたすらパケットを処理し続けることが重要なのであり, 処理された結果の戻り値に攻撃者は特に興味がない. したがって, 攻撃者は自身が送ったパケットの返答

を受け取るために自身の IP アドレスを相手に通知する必要がない．一方, 攻撃の発信元を相手に通知すれば, そこからの通信を遮断されたり, 発信源を特定されて最悪の場合は逮捕される可能性もある．そこで, 攻撃者は (D)DoS 攻撃の際に送信元 IP アドレスを偽装したパケットを送信することがある．攻撃者にとって, 偽装する IP アドレスは自分自身のものでなければ何でもよいため, IPv4 アドレスをランダムに選択することが多い．この際, 選ばれた偽装先 IP アドレスが偶然ダークネットを構成している IP アドレスであった場合, (D)DoS 攻撃を受けたサーバーは送られてきたパケットをダークネットから送られてきたパケットであると認識する．すると, そのサーバーはダークネットに対して応答パケットを返送する．これが (D)DoS 攻撃のバックスキャッタと呼ばれるパケットである．ダークネットに到達するパケットは TCP 3-way handshake の 2 個目のパケットであるため, バックスキャッタのパケットには $SYN - ACK \cap ACK \cap RST \cap RST - ACK$ の Flag が付与される [18].
最後に, IP アドレスを手入力する際に打ち間違えるなどの設定ミスによる通信がダークネット上で観測される, ただし, これらはスキャンやバックスキャッタと比較して少数であると考えられる．このように, TCP flags を分析することで, ダークネットで観測された TCP パケットを大別することが可能である．

2.2.2　観測されるパケット数

2008 年から 2021 年にかけて, 日本国内で収集している /18 のダークネット上で観測されたパケットの総数を図 2.2 に示す．ダークネット上で観測されるパケットは年々増加しており, サイバー攻撃が活発化を続けていることがわかる．また, TCP, UDP, ICMP のパケットが観測されており, TCP パケットが一貫して攻撃の主流であることもわかる．

2.2.3　payload

送信されたパケットの意図を推定する上で, 最も確実なのはパケットの payload を観察することである．しかしながら, ダークネット上で観測される TCP パケットにおいて, payload を保持しているパケットは限定的である．図 2.3 になんらかの payload を保持するパケットの割合を示す．payload を保持するパケットが極めて少なく, 多少の変動はあるが概ね全体の 0.1% 程度であることがわかる．ダークネットはあらゆるパケットに対し一切の返信を行わないため, TCP 3-Way Handshake のプロセスにおいても, SYN に対し SYN-ACK を返すことはない．従って, TCP 通信が session が確立されないため, 意味のあるデータのやり取りが行われていないものと考えられる．なお, UDP においては 3-Way ハンドシェイクは存在しないため, 大半のパケットが payload を所持している．

2.2.4　宛先ポート

ダークネットに当てられた TCP パケットの宛先ポートを分析することは, サイバー空間上で発生している攻撃の傾向を捉える上で有用である．図 2.4 は, 2008 年から 2021 年にかけて,

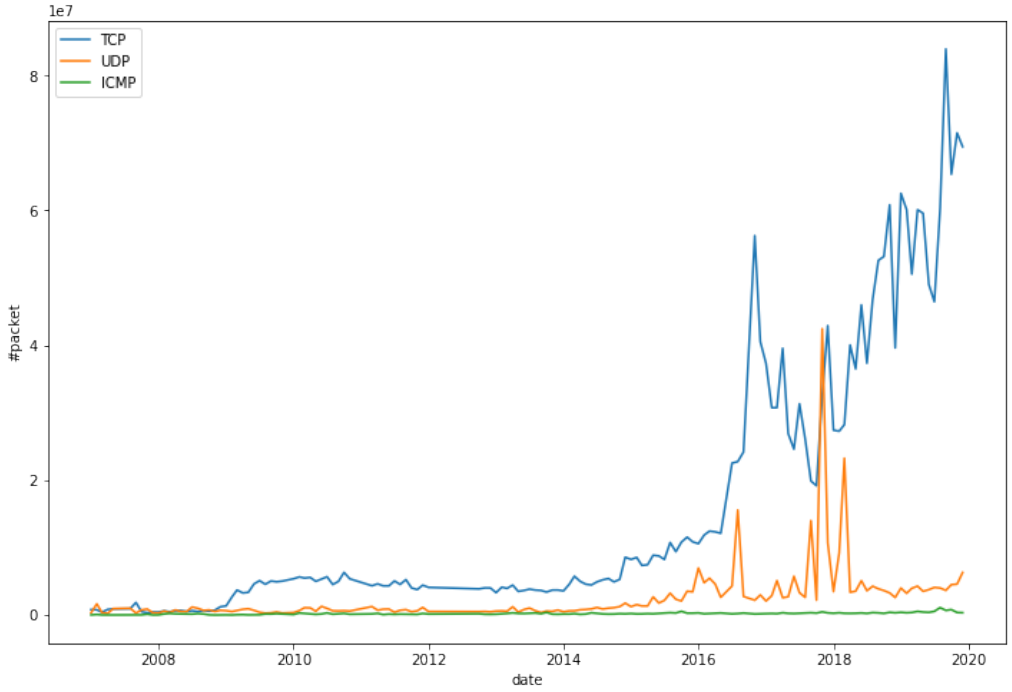

図 2.2: ダークネット上で観測されるパケットの総数

宛先ポートのパケット数が 10 位以内に入ったことがあるポート宛のパケット数の推移である．2008 年から 2014 年頃にかけては SMB などが動作する 445/TCP 宛のパケットが支配的であったが，これは Conficker と呼ばれるマルウェアの感染が全世界で拡大したのが原因であると考えられる [19]．一方、2015 年ごろから Telnet が稼働する 23/TCP 宛のパケットが増加しており，特に 2016 年には 23/TCP 宛のパケットが急増している．これは Mirai と呼ばれるマルウェアの感染が拡大したためである [20]．また，近年になって Android Debug Bridge が稼働する 5555/TCP へのアクセスや、RCE の脆弱性が知られているルーターが動作している 52869/TCP へのアクセス [21] など多様なアクセスが増加している．

図 2.5 に，ダークネット上で観測される TCP パケットの宛先ポートのエントロピーの推移を示す．エントロピーが大きければ大きいほど，多種多様な宛先ポート宛のパケットがダークネット上で観測されることを意味する．グラフから，時間の経過とともにサイバー攻撃が複雑化かつ多様化している様子が見て取れる．

2.2.5　ダークネット上で観測されるパケットの変化の活発さについて

基本的に，マルウェアのスキャンをダークネットで受信することはランダムに発生するイベントであると考えられる．従って，サイバー攻撃の状態が安定しておりインターネット全体が一種の平衡状態にあるとすれば，観測されるパケットの数はポワソン分布に従うはずである．一方，新たなマルウェアが開発され，急速に拡散するような場合は，定常性が成り立たないことからポワソン分布から外れた数のパケットが観測されるはずである．2018 年 1 年間で，なんら

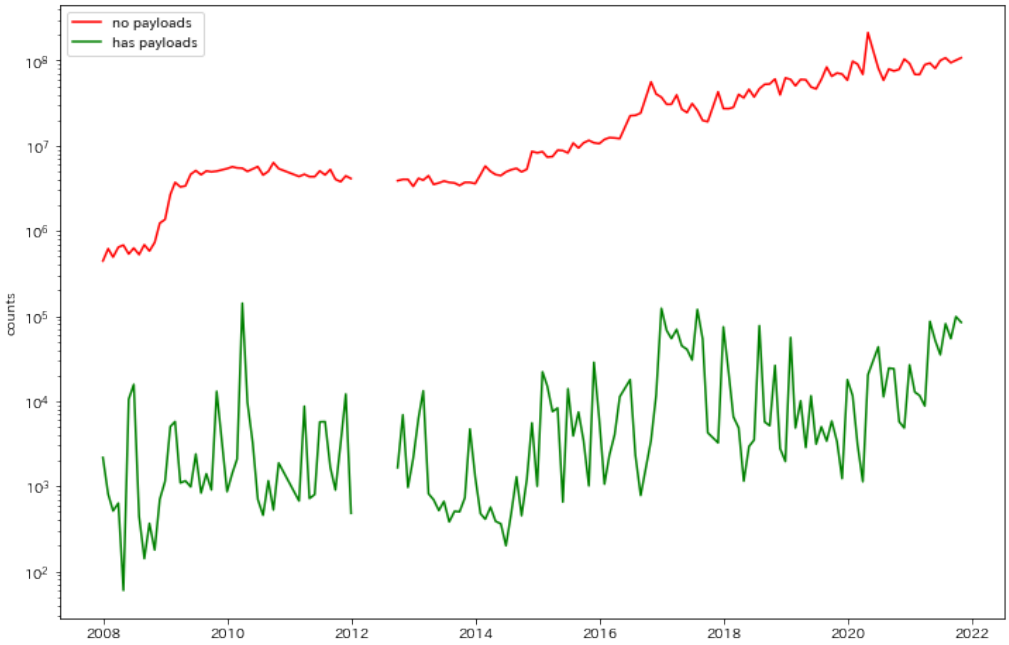

図 2.3: なんらかの payload を保持する TCP パケットの割合

かの宛先ポート宛の TCP パケットの数が 99.99999% 信頼区間より多い日付を探したところ, データが存在している 361 日すべてが, なんらかのポートがポワソン分布に従わないという結果となった. なお, パケットがランダムに来ると仮定した場合, なんらかのポート宛のパケットの数が 99.99999% 信頼区間の外側になる確率は $1-(0.9999999)^{65536}=0.0065$ である. なお, 99.9999% 信頼区間の場合はこの値が 0.063 となるため, 10 から 20 日程度はパケットの発生が偶然であるとしても信頼区間の外に該当する. 信頼区間を 99.999% にまで緩めると, パケットがランダムで発生するとしても 48% の確率で少なくとも 1 つのポートについて分布から外れた数が発生することには注意が必要である. パケットがランダムであれば, 該当する日付はせいぜい 361 日中 1 日か 2 日であるため, 361 日全てが信頼区間の外側にいるということはパケットの到着はまったくもって定常性を持たないということであり, 日々新しい攻撃が発生していると考えられる.

また, 宛先ポートの変化が日々あまりにも激しいことから, 単純に特定のポートあてのパケットを数えるだけでは変動が激しすぎてマルウェアのアウトブレイクを補足することは困難である.

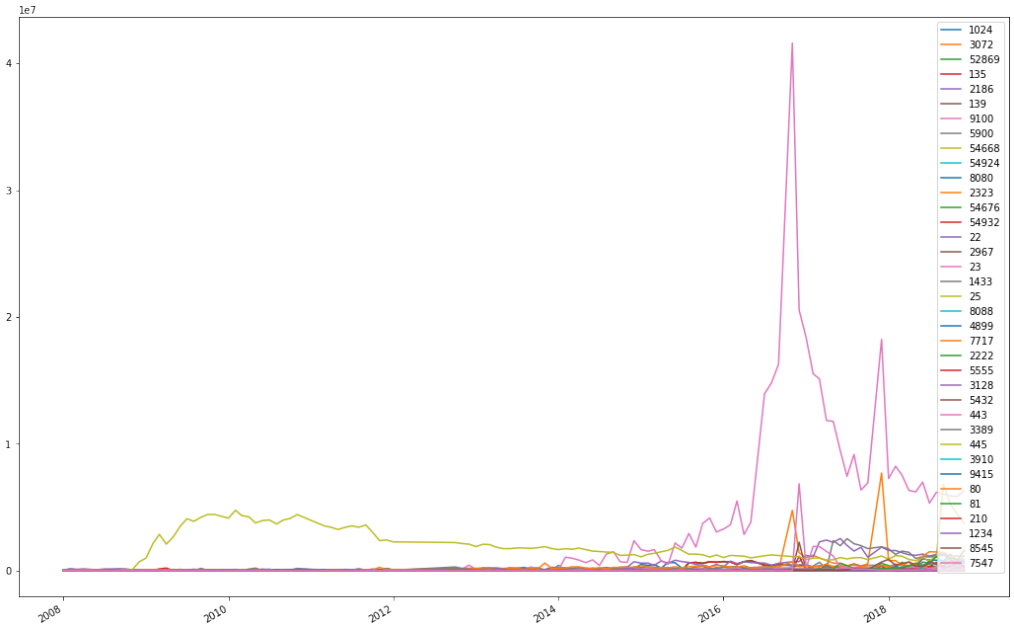

図 2.4: 特定のポート宛のパケット数の推移

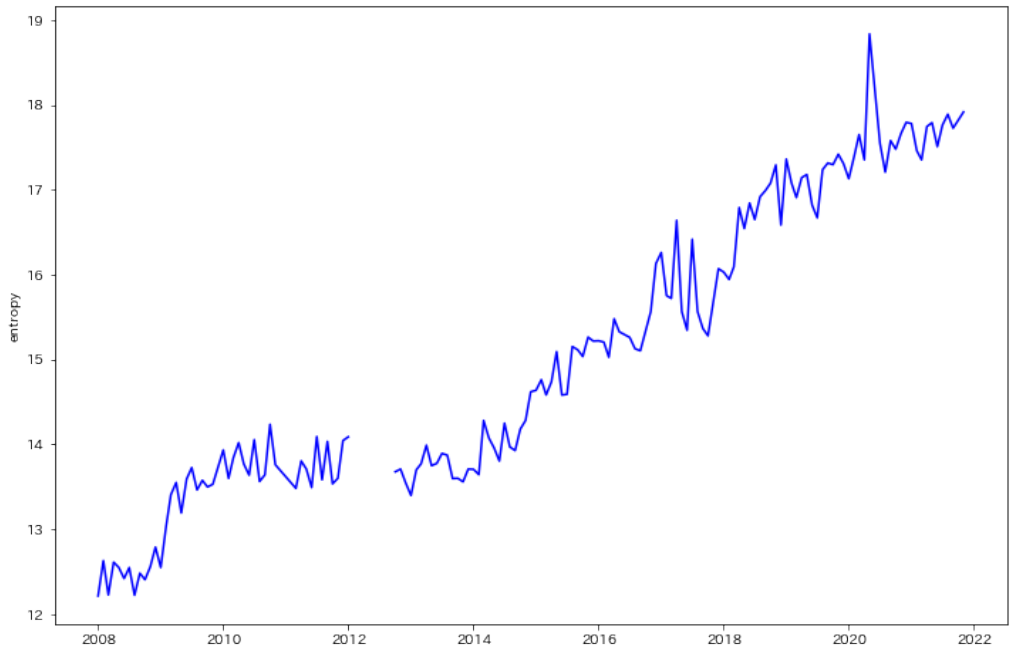

図 2.5: 宛先ポートのエントロピーの推移

第 3 章

関連研究

3.1 パケットの分類, 異常検知

ダークネット上で観測されるパケットは, かつては少数であったが, 近年では人間が 1 つの 1 つのパケットを観察するにはあまりにも多すぎる量のパケットが観測される様になった. したがって, パケットを何らかの形で分類する必要性がある. また, ダークネットではないが, 同じくパケットを分類し悪性のパケットを検知する技術として Newtork Intrusion Detection System (NIDS) が存在し, 研究されている. 本章では, これまでに行われたダークネット及び NIDS に関する研究について述べる. NIDS 等で用いられるパケットの分類のアプローチは大きく分けて 3 種類存在する. それは,

- ルールベース
- 教師あり学習の適用
- 教師なし学習の適用

である.

3.1.1 ルールベースによる異常なパケットの検知

[22] はダークネットで観測されたパケットの宛先ポート番号に注目してパケットを分類した. また, そのルールを 10 年以上分のダークネットのデータに適用し, それぞれのカテゴリに属するパケットの観測量の変化を示した.
[23] はパケットのアクセス先や付与される TCP flag などを基にダークネット上で観測されるパケットの意図を分類するルールを作成し, 観測データに適用した. また, [24] ではこのルールを半年間分のダークネットのデータに適用し, いくつかの分類に属するパケットの量が時間変化していることを発見した. [25] は, このルールを拡張してダークネットで観測されるパケットを分類する Taxonomy のルールを作成し, 10 年以上の観測データに適用した. 表 3.1 にこのルールを示す. また、これにより分類されたパケットの割合の変化を図 3.1 に示す.
[25] では, 2017 年までのダークネットのデータの分析により, Light Network Scan が近年

表 3.1: Taxonomy のルール

Anomaly	Category		Traffic rules
Port Scan	TCP	Heavy	(#ipSrc=1)∩(#ipDst=1)∩(#portDst≥N_2)∩(ScanFlagPktRatio≥R%)∩(Avg #Pkt/portDst>M)
		Light	(#ipSrc=1)∩(#ipDst=1)∩(#portDst≥N_2)∩(ScanFlagPktRatio≥R%)∩(Avg #Pkt/portDst≤M)
	UDP	Heavy	(#ipSrc=1)∩(#ipDst=1)∩(#portDst≥N_2)∩(Avg #Pkt/portDst>M)
		Light	(#ipSrc=1)∩(#ipDst=1)∩(#portDst≥N_2)∩(Avg #Pkt/portDst≤M)
Network Scan	TCP	Heavy	(#ipSrc=1)∩(#portDst=1)∩(#ipDst≥N_1)∩(ScanFlagPktRatio≥R%)∩(Avg #Pkt/portDst>M)
		Light	(#ipSrc=1)∩(#portDst=1)∩(#ipDst≥N_1)∩(ScanFlagPktRatio≥R%)∩(Avg #Pkt/portDst≤M)
	UDP	Heavy	(#ipSrc=1)∩(#portDst=1)∩(#ipDst≥N_1)∩(Avg #Pkt/portDst>M)
		Light	(#ipSrc=1)∩(#portDst=1)∩(#ipDst≥N_1)∩(Avg #Pkt/portDst≤M)
	ICMP	Heavy	(#ipSrc=1)∩(#ipDst≥N_1)∩((Type,Code)=(8,0))∩(Avg #Pkt/ipDst>M)
		Light	(#ipSrc=1)∩(#ipDst≥N_1)∩((Type,Code)=(8,0))∩(Avg #Pkt/ipDst≤M)
One Flow	TCP		(#ipSrc=1)∩(#ipDst=1)∩(#portDst=1)∩(#Pkt>N_3)∩(Protocol=TCP)
	UDP		(#ipSrc=1)∩(#ipDst=1)∩(#portDst=1)∩(#Pkt>N_3)∩(Protocol=UDP)
Backscatter	TCP		(#ipSrc=1)∩(#Pkt≥1)∩(TCP Flags∈{SA∪A∪R∪RA}
	UDP		(#ipSrc=1)∩(#Pkt≥1)∩(#portSrc∈{53∪123∪137∪161})∩(Protocol=UDP)
	ICMP		(#ipSrc=1)∩(#Pkt≥ 1)∩(((Type,Code)=(8,0))∩(Type=3)∩((Type,Code)=(11,0)))
IP Fragment			(#ipSrc=1)∩(#FragmentPkt≥1)
Small SYN			(#ipSrc=1)∩(#ipDst<N_1)∩(#portDst<N_2∩(#Pkt_3∩(TCP Flags=S)
Small UDP			(#ipSrc=1)∩(#ipDst<N_1)∩(#portDst<N_2∩(#Pkt≤N_3∩(Protocol=UDP)
Small Ping			(#ipSrc=1)∩(#ipDst<N_1)∩(#portDst<N_2∩(#Pkt≤N_3∩((Type,Code)=(8,0))
Other			Including "Other TCP", "Other UDP", "Other ICMP" and "Other"

増加していることが指摘されたが, 2018 年以降は寧ろ Light Network Scan に代わり Other TCP が増加していることがわかる.

[26] はダークネット上で観測されるパケットを宛先 IP アドレスとポートに基づいて可視化し, 宛先アドレスと宛先ポートが順に 1 ずつ増えていく step scan が存在することを発見した. [27] では, TCP 初期シーケンスや IP ヘッダの ID 値などに固有値が設定されている通信パケットを抽出することで、ネットワーク上で観測される通信を分類する手法が提案されている. ルールベースに基づくパケットの分類や異常検知は解釈性にすぐれ, サイバー攻撃の傾向が変化した際にルールを改定することで追随することが容易である. 一方、日々複雑さをますサイバー攻撃に対し, ルールベースによるアプローチではルールが複雑化し. 今日の状況に応じて更新を続けることが困難になることも予想される [28]. また, ルールベースによるフィルターを作成する場合, 攻撃者がフィルターを回避するように攻撃のパターンに若干の変化をもたせることも考えられる.

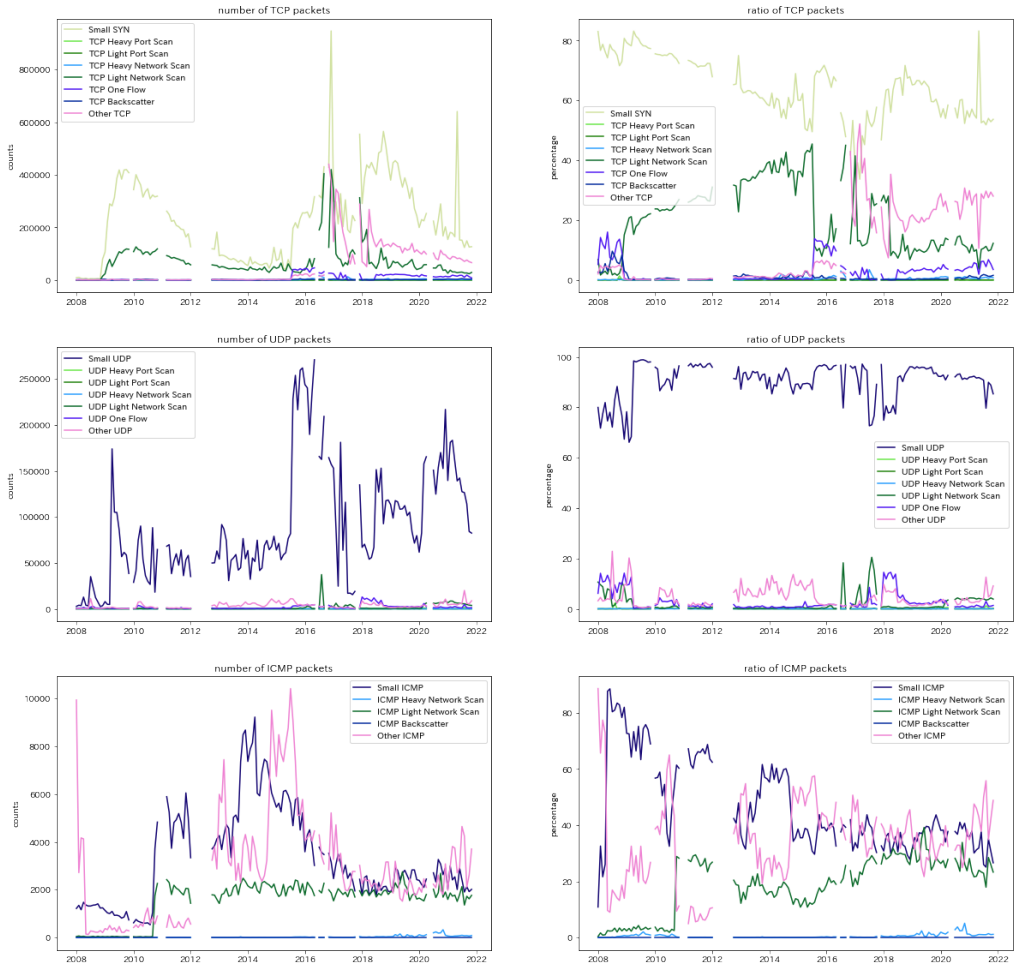

図 3.1: Taxonomy によるパケットの分類

3.1.2　NIDS やダークネットへの教師あり学習の適用

教師あり学習においては, 正常なデータインスタンスと異常なデータインスタンスの両方のラベルを使用して, 教師付きの 2 値または多値の分類機を学習する. [29] では, ダークネットで観測されたパケットから (D)DoS 攻撃のバックスキャッタの特徴を持つ TCP パケットを抽出することによりバックスキャッタのデータセットを作成し, 教師あり学習機である SVM(Support Vector Machine) を用いて (D)DoS 攻撃を検知するモデルを作成した. また, RAN-LSH を用いて同様に (D)DoS 攻撃を検出する検出器を作成することで, 高い精度が出ることが報告されている [30]。

[31] ではネットワークの侵入検知で広く用いられている KDD-99[32] を改良した NSL-KDD データセット [33][34] を RandomForest, J48, SVM, CART, Naive Bayes のモデルに学習させてそれぞれの性能を評価し, Random Forest と SVM が高い性能を出すことを示した. また, NSL-KDD データセットは 41 の特徴量から構成されているが, この研究ではそれらをす

べて用いた場合と比較して, 選択した 15 の特徴量のみを用いた場合にはいずれのモデルでも性能が改良されることも報告されている. [35] は CNN(Convolution Neural Network) を用いたモデルを作成し, Neural Net を用いた NIDS が NSL-KDD データセットにおいて従来の Random Forest や SVM よりも高い性能を出すことを示した。[36] では, CAIDA のダークネットから収集されたデータを元に Shodan を利用してラベルを作成し, 勾配 Boosting でスキャンとバックスキャッタ, 設定ミスを分類するモデルを作成した.
教師あり学習は, 人力によるルールベースでは追いきれない異常を捉える可能性があり, また 3.1.3 で述べる教師なし学習と比べて一般に精度が高いことが知られている. 一方, ネットワークの異常検知において, 異常なクラスのデータを用意することは極めて困難である [37] という問題がある.

3.1.3 NIDS やダークネットへの教師なし学習の適用

教師なし学習は, 教師あり学習と同様にパケットを分類することを目的としているが, 教師あり学習と異なり異常なデータインスタンスのラベルを使用しないアプローチである. 防御側にとって未知の攻撃であるゼロデイ攻撃に対応するラベル付きデータセットを入手することは不可能であるため, 教師あり学習でゼロデイ攻撃を検知することはできないが, 教師なし学習であればゼロデイ攻撃を補足できる可能性がある. なお、文献によってはこれらの手法を半教師あり学習や自己教師あり学習などと呼ぶこともあるが, 本論文ではこれらの手法を, 異常なクラスにラベルが付与されたデータセットを与えていないという観点から教師なし学習と呼ぶことにする. [38] はパケットフローに対して n-gram モデルを適用し, 正常なプロファイルから著しく逸脱したネットワークフローを検知することで異常を検出した. [39] は複数のクラスタリングアルゴリズムを組み合わせた NIDS を作成し, [40] は SVM を用いた NIDS を作成した. 両者の研究は教師あり学習を用いた NIDS と比較して未知の攻撃を捉える可能性が高いことを示した一方, 既知の攻撃の検出においては教師あり学習を用いた NIDS に劣る結果となった. 一方で, Auto Encoder を用いた NIDS で作成された異常検知機は, クラスタリングや SVM などを用いた異常検出と比較して高い性能を出すことが報告されている [41][42][43]. また, [44] も複数の Auto Encoder を用いたアンサンブルによる異常検知を行い, GMM よりも高い性能を達成した. また Auto Encoder の派生として RNN を用いたモデルもネットワークトラフィックにおける異常検知に有効であることが示されている [45].
[46] は, ダークネットで観測されたパケットに Doc2Vec[47] を適用して, 宛先ポート間の類似度をダークネットへのスキャン行為に基づいて計算した. また, [48] もダークネットで観測されるパケットの宛先ポートの関係性を調べることで宛先ポートの類似度を推定した. [49] では, ダークネットにパケットを送信したホストを Word2Vec[50] に適用してベクトル表現を取得し, クラスタリングを行った. そしてこのクラスタリングにおいて, どのクラスタにも当てはまらないホストを探すことでゼロデイ攻撃の検出を試みた. [51] はパケットの送信元アドレスに着目してベクトル表現を取得し, 未知の IP アドレスの活動を効果的に推定した. [52] はダークネット上で観測される活動の時系列の一致を探ることでボットネットを特定する手法を提案し

た.[53] では複数のホストの時系列的な一致を調べることで, 新規のマルウェアの台頭を予測することが提案されている. [54] はダークネットと低対話型ハニーポットのパケットのペイロードに着目してパケットのクラスタリングを行い, 既存のクラスタに分類されないパケット異常として検知した. この研究においては, 異なる種類の攻撃活動を同一のクラスタとして判定する事例が存在したことに加え, 計算に非常に長い時間がかかったことが報告されている.

3.2　ダークネットのモニタリング

ダークネットはインターネット全体から送られてくるトラフィックを収集できるため, インターネット空間で発生した事象を観測するのに適している. [55] では, マルウェアがダークネットに送信してくるパケットを調べることで マルウェアの感染の被害状況を調査し, 既知のマルウェアの動向を調べるとともに, 新規のマルウェアが台頭していることを明らかにした. [56] では, 2011 年にエジプトとリビアでインターネットが停止された際のダークネットのパケットを BGP などの情報と組み合わせて分析することで, インターネットの検閲と遮断がどのように行われたのかを明らかにした.

3.3　本研究の位置づけ

先行研究が示すように, ダークネットではまさに今日インターネット空間内で発生している現象が観測できる. 日々出現する多種多様な攻撃から価値あるシステムを守る上で, 攻撃をリアルタイムで観測できるのはダークネットの極めて大きな意義である. 一方, 新しい攻撃は常に出現しているので, 今日観測されたデータを常に分析し続けることが現実のサイバー攻撃を防ぐ上では重要である. しかしながら, ダークネットのパケットの分析は過去に届いたパケットを分析するものや未知のパケットを既知のクラスに分類するものが主であり, 新規の攻撃に目を向けた研究は少ない. しかしながら, 日々複雑化高度化するサイバー攻撃に対処するためには, サイバー攻撃の現状を迅速に把握し, まず攻撃が発生したときにそれと気がつくことが必要であると考えられる. 本研究では, 今日のダークネットのデータから新しい攻撃を迅速に自動で検出することを目指す. 新たな攻撃が発生した際には, 未知の攻撃に関連するパケットや攻撃の送信元を既知の攻撃から分離してシステム管理者に通知することで新しい攻撃の実態を効率的に把握することを助けるシステムを構築する.

第 4 章

提案手法

4.1 提案手法の要件

先行研究では数多くの異常検知モデルが提案されているが, ダークネットのパケットの分析を行うための検知モデルが満たすべき条件がいくつかある. まず, ダークネットで観測されるパケットデータは高度に非線形な特徴を示し, 従来の機械学習機がその特徴を正しく学習することは困難であると考えられている [57]. 一方, 多層からなる深層学習機は, 不連続なデータを効果的に扱うことができる [58]. これにより, 深層学習機は他の機械学習アルゴリズムと比較して検出性能における大きな優位性がある [59][60] と考えられている. No Free Lunch Theorum[61] が主張するように, あるモデルが他のモデルより本質的に優れていることは無く, 重要なのはそのモデルが適用する対象に適しているかどうかである. パケットデータを分析する NIDS において Neural Net を用いた手法が高精度を示していることは, Neural Net がパケットデータの分析に有用であることの証左であると考えられる.

次に, ダークネットで観測されるパケットに適用するための異常検知モデルは, 高速に推論をすることが求められる. 一定の期間に渡って収集したデータを分析するにあたって, 収集に要する期間以上の時間がかかるのであれば, そのシステムは最新のデータを分析することはできない. 例えば, 1 日分のデータを分析するのに 2 日かかる場合, その分析が有用である場面は限定的であると考えられる. 一方 2.2 で述べたように, ダークネットに届くパケットは日々増大している. 図 4.1 は, ダークネットに届くパケットの総量と, 一般的な CPU のトランジスタ数が 2006 年と比較して何倍に増大したかを示している. CPU のトランジスタ数は大まかに CPU の性能と比例していると考えられる. CPU のトランジスタ数のデータは [62] を使用した. 分析環境の性能が飛躍的に上昇する一方, 同様に観測されるパケットの総数も急上昇していることがわかる. これは, 世間に流通する PC の性能が向上するにつれて, 高速な scan や DDoS 攻撃が実行できるようになり, また侵害できるデバイスが高性能であればあるほど攻撃者にとって魅力的であるため攻撃者が増えるからであると考えられる. したがって, ダークネットの分析にあたっては, PC の性能向上は寧ろ不利に働くと考えられ, ハードウェアの性能向上の恩恵は受けられない. よって, ダークネット上のデータに適用する異常検知のアルゴリズムは高速なものを採用しなければならない. 入力データ数 N に対して, 計算量 $O(N)$ で動

作することが望ましい．先行研究の多くは実行時間について詳細に言及していないが，ネットワークの異常検知のベンチマークによく用いられる NSL-KDD データセットは 1999 年に開発されたデータセットであり，NSL-KDD データセットに対して現実的な時間内に実行が完了する分析手法が今日においても有効であるとは必ずしも限らない点には注意が必要である．

第三に，学習データを載せることができるメモリには限りがある．2.2 で述べたようにダーク

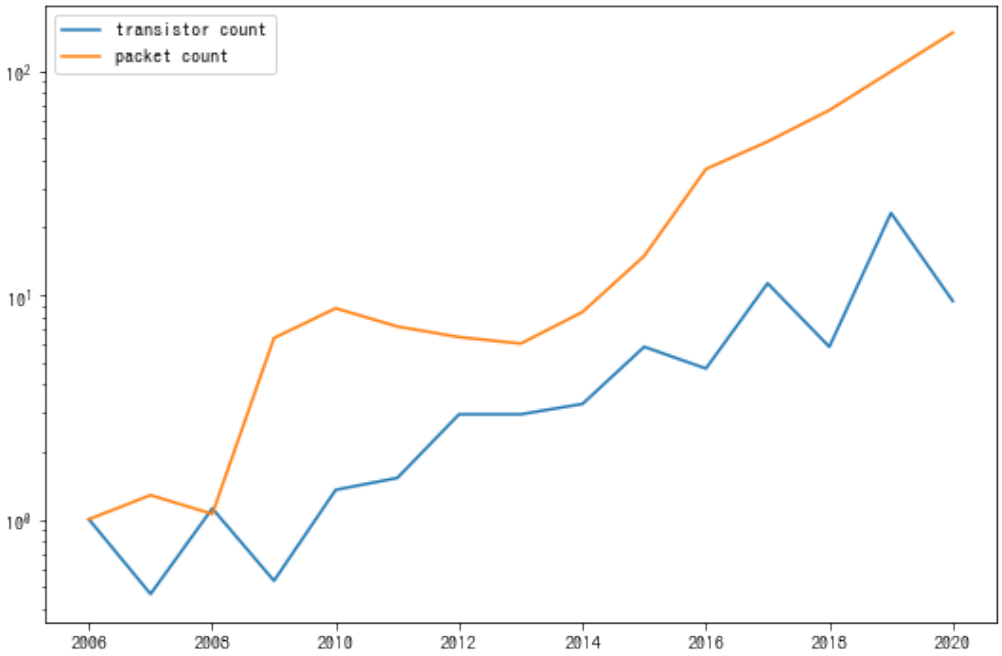

図 4.1: ダークネット上で観測されるパケットの数と，コンピューターの性能向上の比較

ネット上で観測されるパケットの総量は日々増大している．ダークネットの規模や一度に分析する期間にもよるが，一定期間内に観測されるすべてのデータをメモリに載せることは困難であり，たとえ今日可能であったとしても将来的には困難になると考えられる．したがって，ダークネット上で観測されるパケットを分析するにあたっては，いわゆるバッチ学習ではなくオンラインやミニバッチで実行ができるアルゴリズムが望ましい．

最後に，ダークネットに適用される異常検知モデルは常に学習され続ける必要がある．異常検知が有用とされるタスクは，不良品の検出 [63] や病気の診断 [64]，フィッシングの検知 [65] など多岐に渡るが，今回の研究に特有の条件として，異常の性質が一定ではなく常時変化することが挙げられる．検出したい異常である新たな種類の攻撃をダークネットに送信してくるのは，なんらかの意図を持った攻撃者であるが，この攻撃者の攻撃の意図や攻撃の標的は時代に応じて変化する．例えば，ネットワークに接続されていて侵入可能な端末上で不正に仮想通貨を採掘する行為を目的とした攻撃は広く観測されているが，これは 2008 年に初の仮想通貨である bitcoin[66] が提案され，かつ仮想通貨の価格が上昇した後で無いと意味をなさない攻撃である．また，2.2.4 で述べたように，サイバー攻撃の目的だけでなく対象となる標的も日々変化しており，2008 年と 2020 年では標的とされるサービスが大きく異なっている．さらには，防

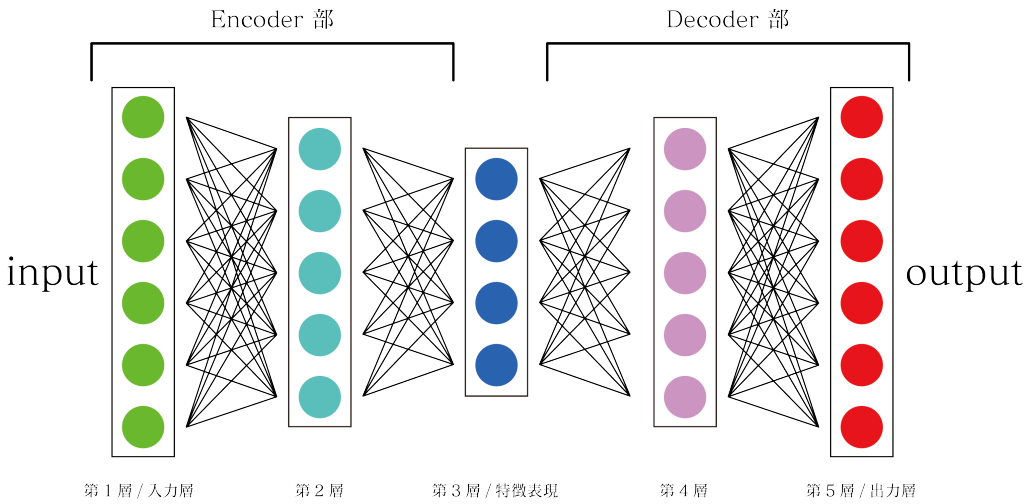

図 4.2: Auto Encoder の構造

御側の検知手法が進化するに従い, 攻撃側はそれまで使っていた攻撃手法にあらたな偽装を施すこともある. これらの事実は, 1 つのモデルを長期に渡って用いることは困難であることを示している. 2008 年のデータを用いて作成したモデルは, 2020 年においては役に立たないであろう. したがって, 異常検知モデルは頻繁に更新されなければならず, 常に学習を行いつづける必要がある. 推論が高速である必要があるのと同様の理由で, 高速に学習を行うことが可能である軽量なモデルを使用する必要がある.

本研究では, 異常検知の手法として深層学習による高速な異常検知モデルである Auto Encoder の 1 種, VAE を用いる. 理由としては 1. 先行研究において ニューラルネットモデルはパケットを分析する上で高い精度が出ることが知られており, 2. モデルの学習, 異常検出がどちらも $O(n)$ で実行できるミニバッチアルゴリズムであり, 3. 同じくニューラルネットベースの GAN や metric learning を用いたモデルと比較して単純な構造であるため学習と実行の速度が高速であるためである.

4.2 Auto Encoder

この節では Auto Encoder について述べる. Auto Encoder は Encoder 部と Decoder 部からなる教師なし学習による深層学習モデルであり, 新しいコンテンツの生成 [67] や画像のノイズ除去 [68] など様々な用途に用いられている. 本研究では Auto Encoder を異常検知のために用いる. 図 4.2 に Auto Encoder の概要を示す. Auto Encoder の学習において, i 番目の入力データを X_i, 対応する出力データを X_i' としたとき, この両者の差がなるべく小さくなるように Auto Encoder を学習する. すなわち, 4.1 で表される二乗誤差 L が最小となるように学習を行う. ここで, N は入力データの総数である.

$$L = \frac{1}{N}\sum_{i=1}^{N}(X_i - X_i')^2 \tag{4.1}$$

L は入力データそれ自身を再構成した結果と自身の差であるので, 再構築ロスとも呼ばれる. Auto Encoder では, 中間での内部表現は入力データよりも表現力が低い. したがって, 入力データと同じデータを出力するにあたっては, 入力を単純にコピーしたものを出力するわけにはいかず, 入力データの中で出力を再構築するのに必要なデータを取捨選択して学ぶことが期待できる.
Auto Encoder は高い汎化性能を持つ. すなわち、訓練データに存在しないデータが入力された場合, Auto Encoder はそれをかなり正しく再構成できる. しかしながら, 中間層での内部表現が入力の持つ情報を全て保持するわけではない以上, この再構成には限界がある. 入力データと同じデータが訓練データに存在したり, 全く同じではなくとも入力データと類似したデータが訓練データに存在する場合には, 入力と出力は比較的近い値を取り, 再構築ロスは小さいと考えられる. 一方, 入力データが訓練データと全く異なる特徴を示す場合, Auto Ecnoder はデータを上手く扱えず, 入力データと出力データは大きく異なり, 再構築ロスが大きくなると考えられる. これを用いて, 図のように 4.3 異常検知を行うことができる.

4.3　VAE

Variational Auto Encoders (VAE)[69] は最も広く使われている Auto Encoder の一種である. VAE は, Auto Encoder 同様に Encoder 部と Decoder 部からなるモデルであるが, Encoder が入力に対する特徴表現を直接生成する代わりに, 平均コーディング μ と標準偏差 σ を計算する点が異なる. 図 4.4 に VAE の概要を示す. VAE では 4.2 で表される誤差 L が最小となるよう学習を行う. ここで, K はコーディング層の次元数であり, μ_i と σ_i はそれぞれ μ, σ の i 番目の要素である.

$$L = -\frac{1}{2}\sum_{i=1}^{K} 1 + log(\sigma_i^2) - \sigma_i^2 - \mu_i^2 \tag{4.2}$$

4.4　提案手法の流れ

ある日が, 前日までと比較した際に異常なことが発生しているかを判断するために, 前日までのダークネットのデータを用いて 4.3 で説明する VAE を訓練する. そして, 前日のパケットの Auto Encoder による再構築ロスの分布と当日のパケットの再構築ロスの分布を比較する. 異常検知の対象である日が前日までと比較して特に大きな差がない場合, 前日のパケットの再構築ロスの分布と当日のパケットの再構築ロスの分布は特に差が無いと考えられる. 一方, その日に観測されたパケットが前日と大きく異なる場合, すなわち新たな攻撃が発生している場合には, 図 4.6 のように両者の誤差分布は大きく異なることになる. したがって, 前日と当日の誤差分布の KL-divergence の大きさが, 前日と当日のサイバー攻撃の変化の大きさとみなせる. 図 4.5 に示すように, この前日までの観測データを用いたモデルの作成と当日の観測データとの比較を毎日行うことで, 異常が起きた際に速やかにそれを検知できる.
また, 前日と当日の誤差分布の KL-divergence が大きく, 何らかの異常が発生していると推定

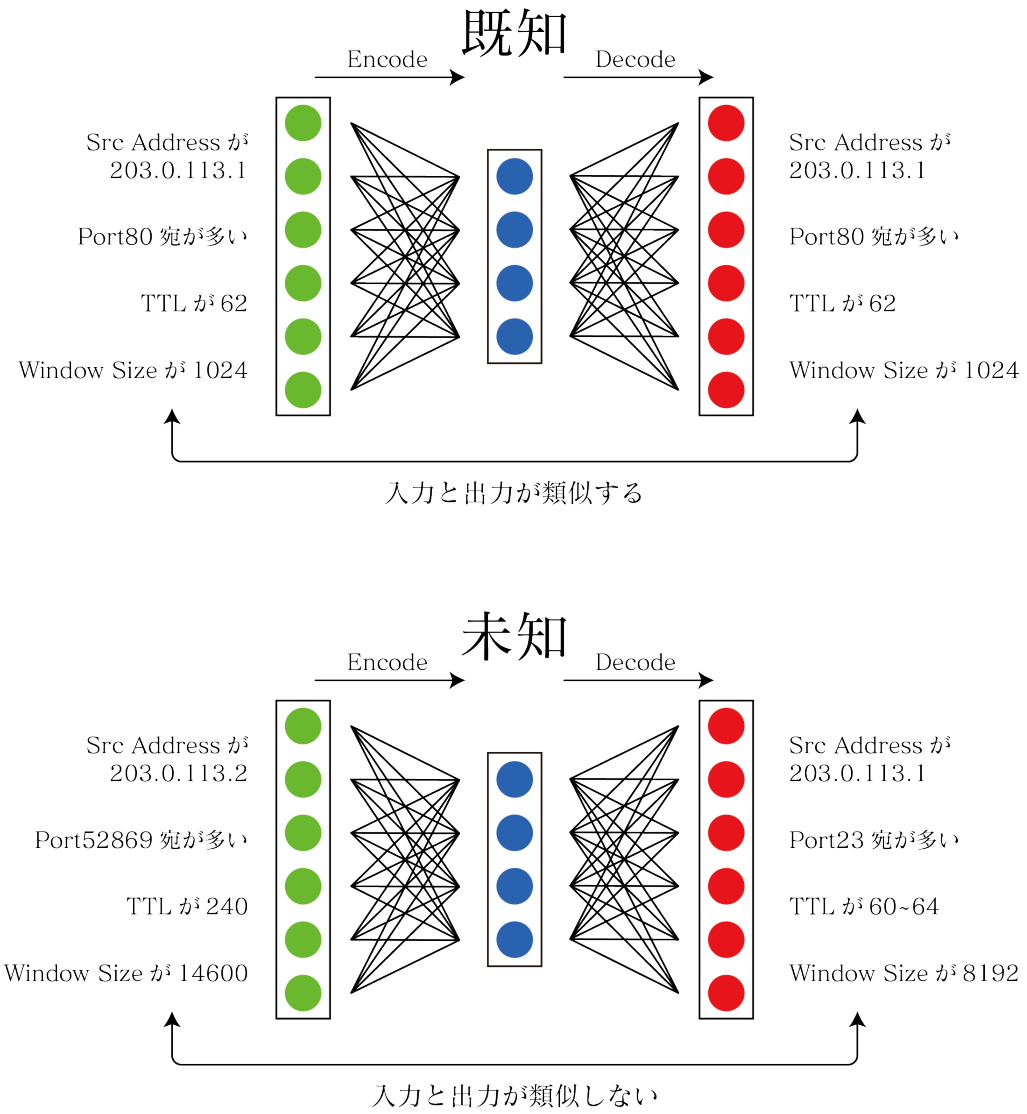

図 4.3: AutoEncoder を用いた異常検知

される日においては, 4.6 の矢印で示したような誤差分布の形状が異なっている点が存在するはずである. この誤差に相当する発信元が送信してきたパケットが前日と当日の違いの原因であるため, 該当するパケットを抽出することで, 新たな攻撃に関連するパケットのみを効果的に抽出することができる.

4.5 特徴量

本節では, Auto Encoder に入力する TCP パケットの特徴量について述べる. 表 4.1 に, Auto Encoder に与える特徴量の一覧を示す. また, すべての数値は最小値が 0, 最大値が 1 となるよう正規化する.

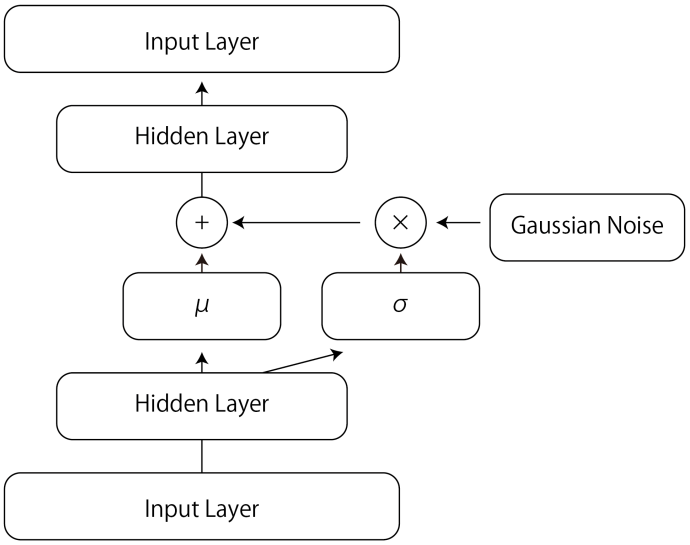

図 4.4: Variational Auto Encoders の構造

4.5.1　利用する特徴量

本節では特徴量として採用する値について述べる．具体的なエンコーディングの方法については 4.5.2 で述べる．

TCP/IP ヘッダ

IP パケット及び TCP パケットにはヘッダが存在し, これらのフィールドを本研究においても特徴量として使用する．インターネット上の通信の多くは IPv4 を用いて TCP で実装されているので, 今後発生する未知のゼロデイ攻撃においても TCP/IP のヘッダの情報は存在すると考えられ, これらはサイバー攻撃の種類を識別する上で有用である．ただし, 我々は IPv4 上の TCP パケットを分析の対象としているため, IP version 及び Protocol のフィールドはそれぞれ 4, 6 で固定されている．そのため, これらのフィールドは有用ではないと考えられる．

アプリケーション層

攻撃を受ける可能性があるアプリケーションは世の中に多くの種類が存在するので, それら全てのプロトコルに対応した異常検知システムを構築するのは現実的ではない．また、2.2.3 に示すとおり, ダークネット上で観測されるパケットの殆どは payload を保持していないため, payload から得られる情報は極めて限定的であると考えられる．以上の理由から, TCP/IP より上位のアプリケーション層の情報は特徴量として採用しない．

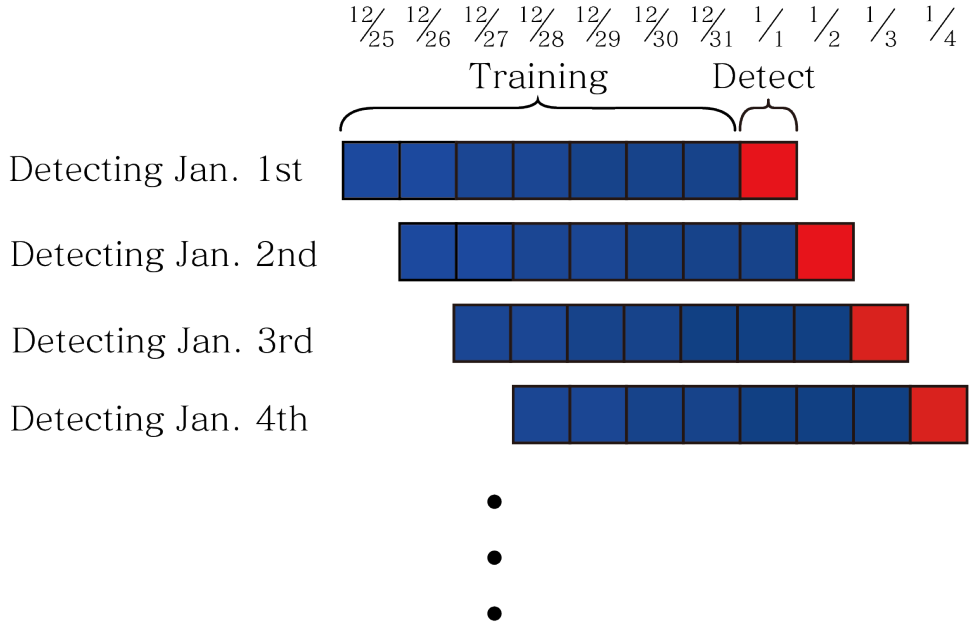

図 4.5: 異常検知の流れ

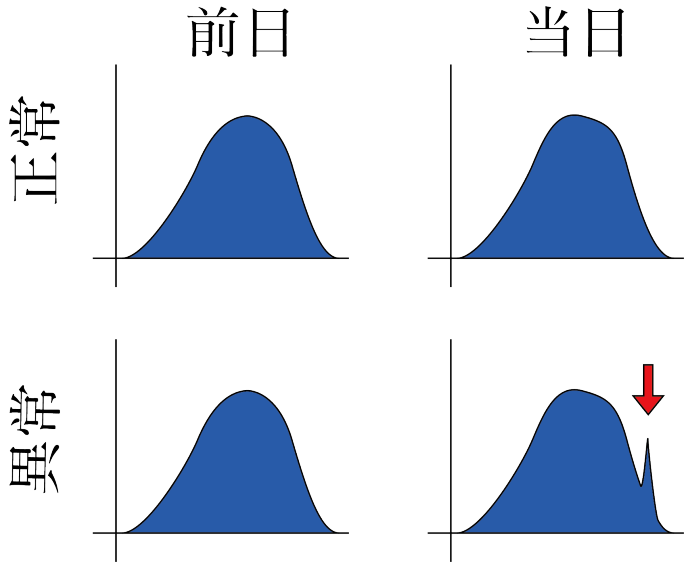

図 4.6: 正常な日と異常な日の誤差分布

パケット数

同一の送信元から 1 日に送られてくるパケットの数を特徴量として用いる．スキャンの意図や悪質性を図る上で重要な指標である．

パケットの到着間隔

攻撃ホストがパケットを送信する頻度は, 不審なスキャンや攻撃を行うマルウェアの実装や攻撃を実際に行っている機器の性能等により大きく異なると考えられる. したがって, 1 日に複数のパケットを送信してくるホストについて, そのパケットの到着の時間間隔を特徴量として用いる.

4.5.2　特徴量のエンコーディング

TCP/IP ヘッダ

IP アドレスやポートなど, ヘッダ内の特徴量は数字で表されるものが多いが, これらは数値に名義以上の意味があるわけでは無いものがある. 例えば, HTTP は通常 80 番ポートで稼働し, HTTPS は 443 番ポートで稼働するが, 262 番ポートがその中間というわけではない. また, 22 番ポートで稼働する SSH は 21 番ポートで稼働する FTP より大きいというわけでもない. TTL は間隔尺度や順序尺度と考える事もできるが, その初期値が OS によって異なることなどから, TTL の大小が単純にホストのなんらかの性質の大小と対応するわけではない. したがってこれらの特徴量は, IP Length を除き順序尺度として扱われるべきである。

これらの値を数字のまま Auto Econder に入力すると, 数値の大小関係などを意味もなく学習してしまう. そこで, 名義尺度を機械学習で扱う場合には, 値をを One-Hot 特徴量として入力することが多い. しかし, これを実施すると 16bit のポートを Auto Encoder に与えるためには $2^{16} = 65536$ 個の特徴量が必要であり、32bit の IP アドレスを Auto Encoder に与えるためには $2^{32} = 4294967296$ 個の特徴量が必要になる. これは極めてメモリ効率が悪い上, 2^{32} 個のフィールドがありながら bit が 1 であるフィールドが 1 つだけというのはデータが疎でありすぎるため, Auto Encoder が情報をうまく捉えられないと考えられる. そこで, 各々の特徴量を bit に変換し, それぞれの桁を特徴量とする. 例として, 16bit で表現される宛先ポートが 54924 番ポートであった場合, 1,1,0,1,0,1,1,0,1,0,0,0,1,1,0,0 の 16 個がエンコーディング後の特徴量である. IP アドレスや TTL, TCP Sequence 番号などの他のフィールドについても同様の処理を行う.

作成した 0 または 1 の特徴量に対して, パケットを送信してくるホストごとに集計して, 各々の平均をとった値が Auto Encoder に与えられる. また, TCPflags などのもともと 1bit である特徴量については, ホストごとに平均と分散を計算して Auto Encoder に与える.

パケット数

同一のホストから送信されてくるパケットは, (D)DoS 攻撃のバックスキャッタを除き概ね 1 ホストあたり 10 パケット未満である. 特に、ダークネット上で観測されるホストのうち, 図 4.7 に示すように 3 割近くが 1 日あたり 1 パケットのみを送信している. 複数パケットを送信するホストはその意図がつかみやすいことが多い一方, それと比べて単独のパケットや単独のパケットを送信するホストを分析することは難しい. 従来の研究ではこれを無視したり, 詳細

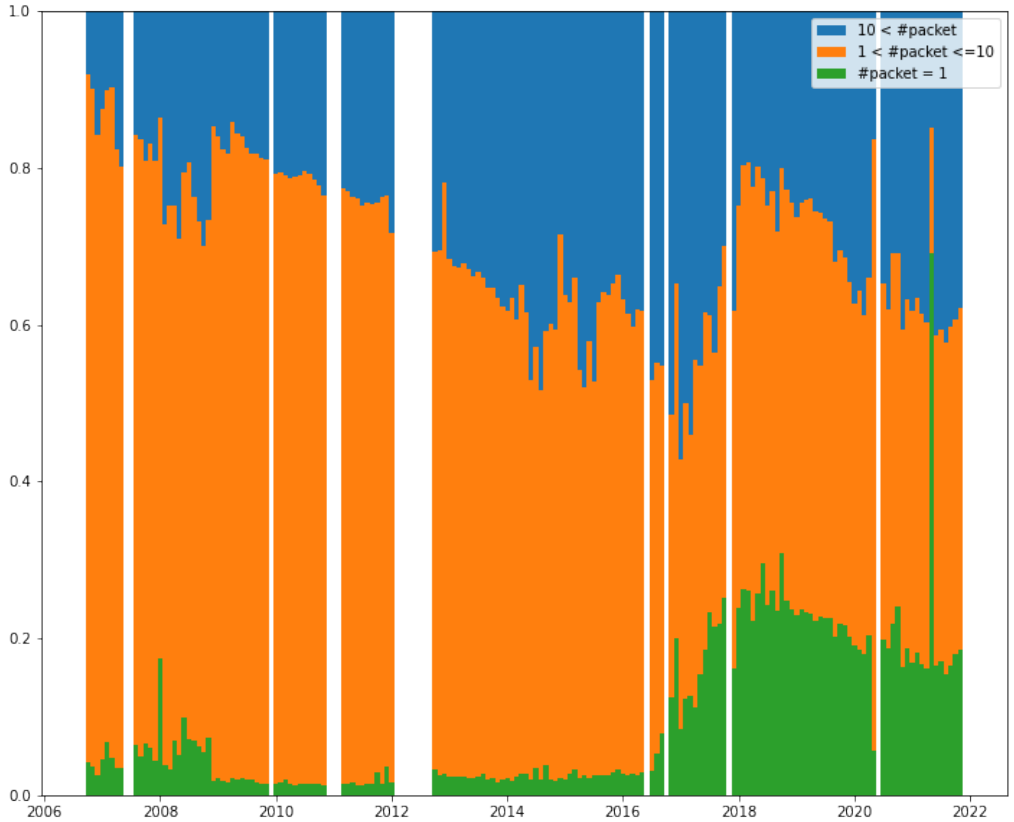

図 4.7: ダークネット上で観測されるホストが 1 日あたりに送信したパケット数

な分析の対象外とすることも多かった．しかしながら, そのようなパケットは年々増加し, 近年では全体の 3 割近くを占めている．そのため, これらを無視することは適切ではない．したがって, 送信されたパケットが 1 つであったかどうかを特徴量として加える．
一方, パケット数そのものには上限が存在しない．これを Auto Encoder に推測させた場合, 特定のホストから送信されるパケットが極端に多い日が存在した場合, Auto Encoder の再構成ロスが巨大になり, 異常検知全体のスコアが実質パケット数のみで決定されてしまう．そして, パケット数が特定の日に極端に増加するのは, (D)DoS 攻撃のバックスキャッタを観測できるダークネットでは頻繁に発生している．したがって, 数そのものを Auto Encoder に与えるのは適切ではない．

パケットの到着間隔

同一ホストから 2 つ以上のパケットが送信された場合, その時間間隔の平均と分散を特徴量とし, 正規化して Auto Encoder に与える．パケットが 1 つのみであった場合はどちらも 0 とする．

表 4.1: 特徴量

	特徴量	特徴量の次元数
IP Header	IP ID	16
	IP checksum	8
	IP offset	16
	TTL	8
	Source IP Address	32
	Destination IP Address	32
	IP Length(mean)	1
	IP Length(var)	1
TCP Header	Source Port	16
	Destination Port	16
	FIN(mean)	1
	SYN(mean)	1
	RST(mean)	1
	PUSH(mean)	1
	ACK(mean)	1
	URG(mean)	1
	ECE(mean)	1
	CWR(mean)	1
	FIN(var)	1
	SYN(var)	1
	RST(var)	1
	PUSH(var)	1
	ACK(var)	1
	URG(var)	1
	ECE(var)	1
	CWR(var)	1
	TCP Sequence	32
	ACK Number	32
	Window Size	16
	TCP checksum	16
	Urp	16
パケット数	Single Packet	1
パケット到着間隔	到着時間の平均	1
	到着時間の分散	1
	計	277

第 5 章

実験

5.1 データセット

我々の研究では, 2006 年 10 月から現在まで日本国内で運用されている, /18 の IP アドレスで構築されたダークネットで収集されたデータを用いる. 観測されたパケットは pcap ファイルの形式で保存される. なお, 一部の日付のデータはダークネットの運用上の問題により欠損している.

5.2 評価ラベル

本研究の目的にはゼロデイ攻撃や未知の攻撃の検出が含まれるため, モデルを運用するにあたって評価ラベルは存在しない. しかしながら, モデルの有用性を図るためには信頼できる評価ラベルが必要である. 本研究では, 実際にダークネットを保有する組織から, 有事が観測された際に専門家の分析を経て公開されたインシデントレポートに記載された事象を正解ラベルとして利用する. ダークネットによって観測されるデータには, IP アドレスが割り当てられている国によって異なる傾向が存在する可能性が指摘されているため, 我々の研究データと同じく日本にあるダークネットに関するインシデントレポートを使用するのが妥当である. 本研究ではインシデントレポートとして NICT が毎年発行している NICTER 観測レポート [70][71][72][73][3], NICT がインシデント発生時にレポートを掲載している NICTER ブログ [74] 及び公式 Twitter[75], 警察庁が発行しているインターネット観測結果, JPCERT/CC が公表しているインターネット定点観測レポート [76], IIJ の SOC チームが公開しているレポートである wizSafe Security Signal[77] を専門家による正解データとして用いる.
レポートにインシデントの発生日が記載されている場合は, その日付を正解ラベルとして用いる. インシデントレポートに具体的な日付ではなく, ○月上旬などの範囲が記載されている場合, その範囲の中で当該パケットの前日からの増加率が最も高かった日を正解ラベルとする.
これらのラベルは, 信頼できる専門家によるラベル付であり, レポートにてインシデントと明言された出来事はインシデントであると考えてよい. 一方, インシデントが発生したと明言されていない日に必ずしもインシデントが発生していないことを意味するわけではないことは注

意が必要である．これは組織のレポート作成や公表のポリシーや, インシデントの単なる見落としなどにより, インシデントとみなせる事象が発生した際にレポートが存在しないことが考えられるためである．

表 5.1 に, 2016 年と 2018 年に報告された TCP に関連するインシデントレポートの概要を示す．宛先ポートは, インシデントに該当するパケットが対象とするポートである．該当するホストがダークネット全体に占める割合は, インシデントとして指摘された特徴を保有するパケットを送信してきたホストが, ダークネット全体で観測されたホストのうちどのくらいを占めるかを表している．Taxonomy は, インシデントに該当するホストが表 3.1 に示す [25] の作成したルールにおいて, 最も多く分類された分類を示している．報告は組織ごとに発行されたインシデントレポートの数を表している．IIJ は 2017 年からレポートを発行しているため,

表 5.1: インシデントレポートを基にした Ground Truth の概要

		2016 年	2018 年
宛先ポート	23	4 件	1 件
	80/443	4 件	11 件
	その他 System Port	3 件	2 件
	公式 User Port	6 件	14 件
	非公式 User Port	8 件	20 件
	動的ポート	0 件	3 件
該当するホストがダークネット全体に占める割合	1% 未満	10 件	37 件
	1% 以上 10% 未満	8 件	6 件
	10% 以上 50% 未満	4 件	8 件
	50% 以上	3 件	1 件
Taxonomy	Heavy Port Scan	0 件	0 件
	Light Port Scan	0 件	0 件
	Heavy Network Scan	0 件	0 件
	Light Network Scan	5 件	8 件
	TCP One Flow	0 件	0 件
	TCP Backscatter	3 件	8 件
	Small Syn	6 件	15 件
	Other TCP	11 件	20 件
報告	NICT	4 件	11 件
	JPCERT/CC	6 件	9 件
	警察庁	22 件	47 件
	IIJ	-件	12 件
計	25 件	計	55 件

2016 年の正解ラベルに IIJ は含まれていない. また, IIJ のレポートには一部「(特定の日付に) 全体的に攻撃の検出件数が増加した」などの記載があるものの攻撃の詳細について言及されていない記述が存在し, これらについては正解ラベルには含めるものの表 5.1 の宛先ポート, 該当パケットが占める割合, Taxonomy には反映されていない. UDP に関する情報やメールの添付ファイルなど, TCP 以外のインシデントがレポートに記載されていた場合, これを除外した. 使用したダークネットでの観測データには停電等の理由により一部欠損があり, その期間に報告されたレポートは正解ラベルに含めていない.
2016 年には計 24 日に 25 件のインシデントが, 2018 年には計 48 日に 55 件のインシデントが存在した.

5.3 実装

実装には Python を用いた. pcap ファイルの解析には dpkt を用い, Auto Encoder の実装には pytorch[78] を使用した. Auto Encoder の学習と推論には NVIDIA GeForce RTX 3090 を用いた.

5.4 パラメーター決定

Auto Encoder を適切に訓練するにあたっては, いくつか決定すべきパラメータが存在する. 表 5.2 に決定すべきパラメータを示す.

表 5.2: Auto Encoder のパラメーター

パラメーター	デフォルト値	候補	最適値
epoch 数	3	1-40	20
訓練日数	7	1-14	1
第 2 層/第 4 層のノード数	60	40-200	100
第 3 層のノード数	20	1-100	40

5.4.1 epoch 数

図 5.1 にモデルを訓練する際の epoch 数を変化させた場合の AUC の変化を示す. 20epoch までは Auto Encoder の性能が上昇していくが, 20epoch 以降は性能が逆に下降していくことがわかる. したがって, 20epoch 程度で AE を訓練するのが妥当である.

5.4.2 訓練日数

4.4 で述べたように, 異常検知機の学習においては前日までに観測されたデータを訓練データとして与える. この際, 前日から何日分のデータを訓練データとして与えるのかは性能を決

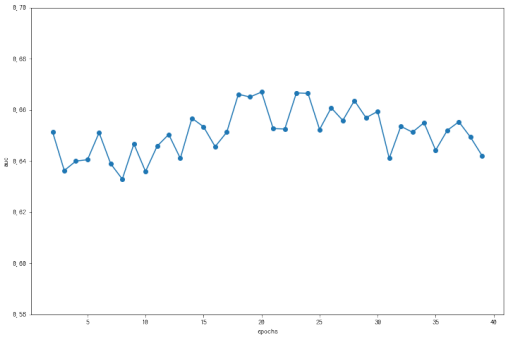

図 5.1: epoch 数

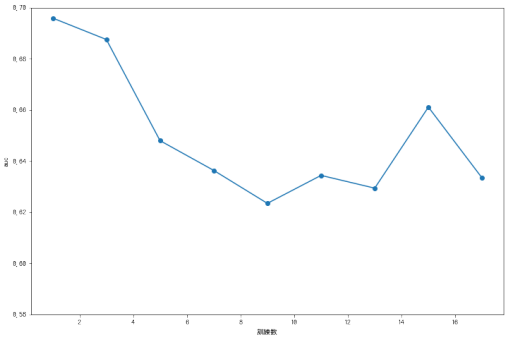

図 5.2: 訓練日数

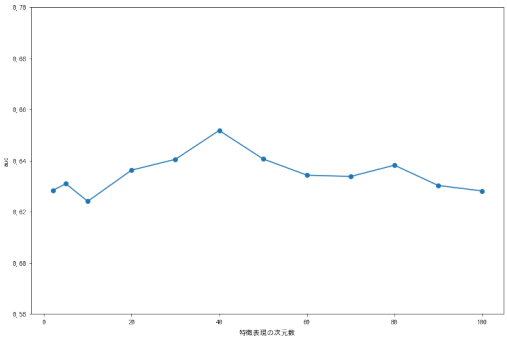

図 5.3: 特徴表現の次元数

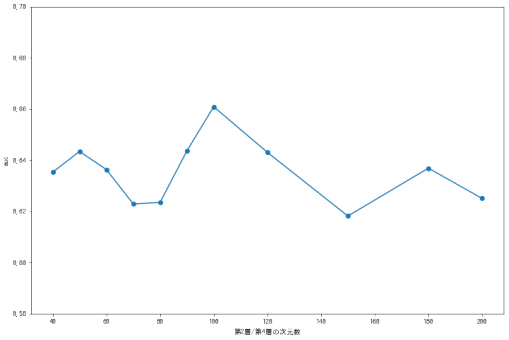

図 5.4: 第 2 層及び第 4 層の次元数

図 5.5: 特徴量とモデルの性能の関係性

める上で重要な要素である．図 5.2 に epoch 数を変化させた際の AUC の変化を示す．Auto Encoder を前日 1 日分のデータを用いて訓練したときが, もっともモデルの精度が良くなることがわかった．これは, 古すぎる過去の情報を過度に学習させると, 現在と異なるデータに引きずられるためであると考えられる．したがって, Auto Encoder の訓練は前日のデータのみを用いて行うのがよい．

5.4.3　コーディングの次元数

4.2 の各レイヤーのうち, input layer のノード数は 277 次元で固定であるが, 残りの中間層のノード数は性能に影響する要素であると考えられる．図 5.3 にコーディングの次元数とモデルの AUC の関係を示す．これより, コーディングの次元数は 40 次元とするのが最も Auto Encoder の性能が高くなることがわかる．

5.4.4　第 2 層/第 4 層の次元数

Auto Encoder が入力と同じものを出力することから, Auto Encoder の Encoder 部と Decoder 部は対称な構造とすべきである．したがって, 第 2 層と第 4 層のノード数は等しくする．また, 第 2 層及び第 4 層のノード数は, 第 3 層であるコーディング層の次元数より多くす

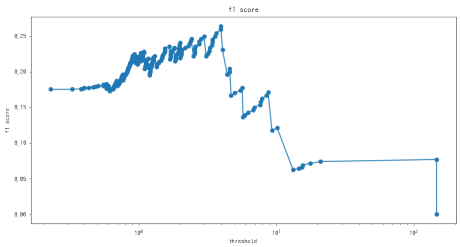

図 5.6: 2016 年における threshold と f 値の関係

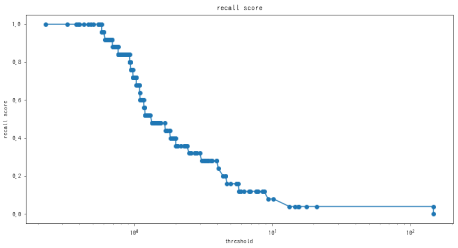

図 5.7: 2016 年における threshold と recall 値の関係

べきである．このノードの次元数と AUC の関係を図 5.4 に示す．これより, 第 2 層及び第 4 層のノード数は 100 次元とするのがよい．

以上をもとに, Auto Encoder を作成するにあたって決定すべきハイパーパラメータを決定し, 異常検知モデルを作成した．

5.5 threshold

Auto Encoder による異常検知では, 大規模なインシデントが発生した場合にモデルの入力と出力の誤差が大きくなる．一方, 具体的に Auto Encoder のスコアがどの値以上であるならば異常と判定し対処すべきなのかという基準値が実用上は重要である．本研究では, 2016 年における異常検知の f1 スコアが最大となるようなしきい値を異常かどうかの境界として用いる．図 5.6 にしきい値と f1 スコアの関係性を示す。Auto Encoder の出力した値が 3.96532 以上である時を異常とみなす時, f1 スコアが 0.23077 となりこれが最大であった。したがって、この値が基準値とみなせる．また, しきい値と racall の関係を図 5.7 に示す．異常検知モデルの誤りには, 存在するゼロデイ攻撃を検出できない検出漏れと, ゼロデイ攻撃が発生していない日を異常と判定する誤検知の 2 通りが存在する．このうち, 後者は専門家が調査すれば誤りが発見できる一方．前者はゼロデイ攻撃を見落とすことを意味する．本研究においては, 見落としをできるだけ避けるべきであると考えられ, 基準値を下げて誤検出を増やしてでも検出漏れを減らすべきであると考えられる．前述のしきい値における recall 値は 0.25 であり, 少なくない数のゼロデイ攻撃を見落としている．したがって, より低いしきい値を採用することを検討する余地がある．一方で NIDS などと異なり, 少数のゼロデイ攻撃の検出に失敗したとしても, それが直ちにこの研究の意味を失うわけではない．そこで, $recall > 0.8$ となるようなしきい値をしきい値として採用する．このとき、しきい値は 0.92322 である。

5.6 2016 年における異常検知の実施結果

2016 年にダークネット上で観測されたデータに対して Auto Encoder が出力した異常スコアを図 5.8 に示す．plot が存在しない部分はデータが欠損している日付である．また, 表 5.3 に 2 種のしきい値において, Auto Encoder が検出した異常と検出しなかった異常の概要を示

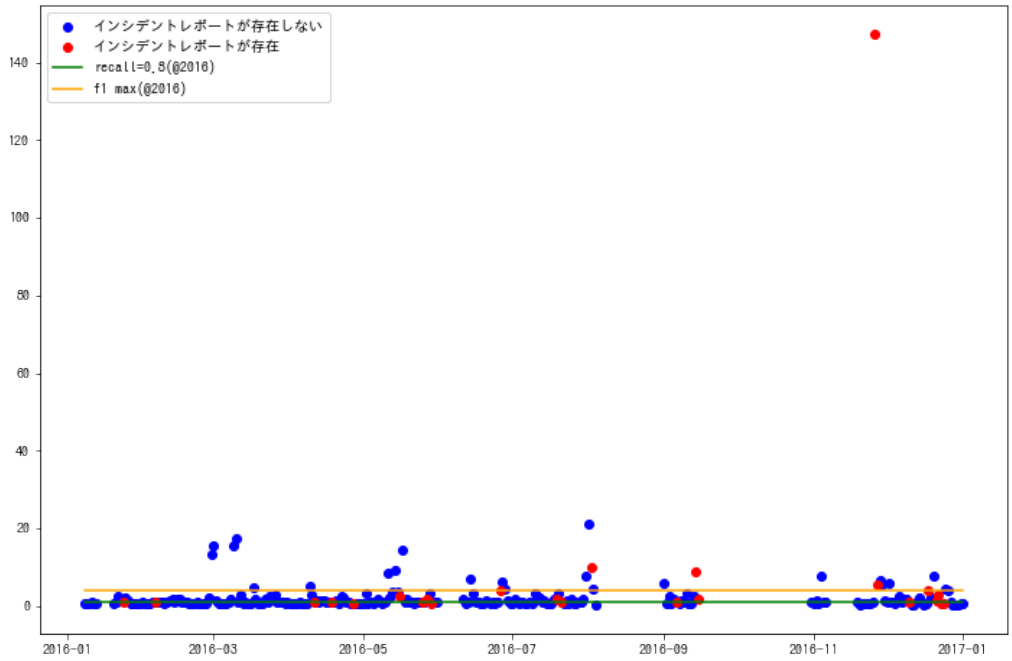

図 5.8: 2016 年における Auto Encoder の異常スコア

す. 2016 年の 1 年間における AUC は 0.62023 であった. また, 実行時間は学習と推論をあわせて 1 日あたり平均 744 秒であった 2016 年における異常検知は教師なし学習で行ったものであるが, ハイパーパラメータを決定するのために一部 2016 年の正解ラベルを参照しているため, 完全な教師なし学習とは言えないことには注意が必要である.

表 5.3: 2016 年における検出結果の概要

f1 値最大 (しきい値が 3.96532)

	Auto Encoder が 異常と判定	Auto Encoder が 正常と判定
レポートが存在する	5 日	19 日
レポートが存在しない	22 日	214 日

recall を重視 (しきい値が 0.92322)

	Auto Encoder が 異常と判定	Auto Encoder が 正常と判定
レポートが存在する	19 日	5 日
レポートが存在しない	142 日	94 日

5.6.1 正しく検出できた異常

表 5.4 にインシデントレポートが存在し, かつ Auto Encoder が異常として検出した事象について記す. f1 スコアが最大となるようしきい値を定めた時, ダークネット全体のうち 50% 近くを占めるような大規模なインシデントは 3 件中 2 件が検出可能であった一方, 全体の 1% にも満たない規模として小規模なインシデントは検出できない結果となった. スキャンの大規模さと攻撃の悪質さや被害の大きさは必ずしも一致しないため, これらの数が少ない攻撃を見落としていることは問題であると言える.

2016 年 11 月 26 日に Auto Encoder が突出して高い異常値を示しているが, これは

表 5.4: Auto Encoder による 2016 年の異常検知

		f1 最大	$recall = 0.8$	total
宛先ポート	23	1 件	4 件	4 件
	80/443	0 件	3 件	4 件
	その他 System Port	1 件	2 件	3 件
	公式 User Port	1 件	5 件	6 件
	非公式 User Port	2 件	6 件	8 件
	動的ポート	0 件	0 件	0 件
該当パケットの割合	1% 未満	0 件	8 件	10 件
	1% 以上 10% 未満	2 件	5 件	8 件
	10% 以上 50% 未満	1 件	4 件	4 件
	50% 以上	2 件	3 件	3 件
Taxonomy	Heavy Port Scan	0 件	0 件	0 件
	Light Port Scan	0 件	0 件	0 件
	Heavy Network Scan	0 件	0 件	0 件
	Light Network Scan	2 件	4 件	5 件
	TCP One Flow	0 件	0 件	0 件
	TCP Backscatter	0 件	3 件	3 件
	Small Syn	2 件	6 件	6 件
	Other	1 件	7 件	11 件
報告	NICT	2 件	4 件	4 件
	JPCERT/CC	2 件	5 件	6 件
	警察庁	4 件	18 件	22 件
	IIJ	-件	-件	-件
計		5 件	21 件	25 件

7547/TCP 宛のパケットが急増したためである。図 5.9 に 11 月 26 日の周辺における 7547

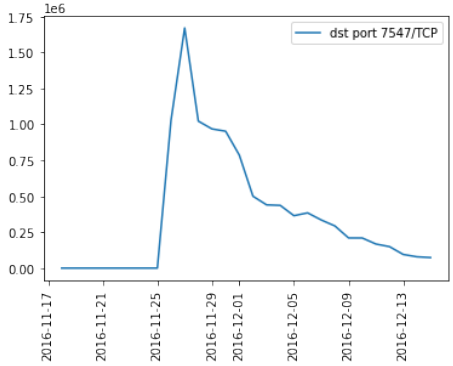

図 5.9: 7547/TCP 宛のパケット数

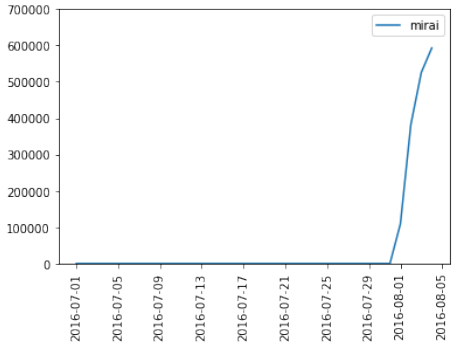

図 5.10: Mirai の特徴を有するパケット数

ポート宛のパケット数の推移を示す．これは、ドイツの電気通信事業者である Deutsche Telekom 社の顧客のルータに脆弱性が存在し, その探索を目的としたスキャンが発生したからであると考えられる [79].

また、8 月 3 日付近にも Auto Encoder が高い異常値を示したが, これは Mirai と呼ばれる IoT 機器に感染するマルウェアが登場し, 無差別なスキャンを開始したことを検知したためであると考えられる [80]. 2016 年上旬に出現した初期の Mirai はスキャン対象の 23 番ポートにアクセスした後, 予め用意されたパスワードリストに基づいて辞書攻撃を行うことで感染を拡大する．Mirai がスキャンのために送信する SYN パケットにはスキャン対象の IP アドレスと TCP パケットの TCP Sequence が一致するという特徴がある．Mirai 以外のプログラムが送信したパケットにおいて, TCP Sequence が偶然 IP アドレスと一致する確率は $(\frac{1}{2})^{32}$ であるため, この特徴をもつパケットが観測された場合は基本的に Mirai によるものだと考えて良い．図 5.10 に Mirai の特徴を有するパケット数の推移を示す．8 月 3 日以降, Mirai の特徴を有するパケットが急増していることがわかる．これらの Mirai の特性は 2016 年 10 月 1 日に Mirai のソースコードがインターネット上に公開されたことで判明したものである．23 番ポートへのスキャンが急増したことは早くから指摘されていたものの, そのパケットに宛先 IP アドレスと TCP Sequence が等しいという特徴があることが指摘されたのは, Mirai のソースコードがインターネット上に公開されて以後のことであった．図 5.12 に 2016 年 8 月 3 日に 23 番ポート宛に送信されたパケットのうちで, Mirai の特徴を有するパケットの割合を示す．23/TCP 宛のパケットのうち, Mirai が占める割合は 35% 程度であり, 23/TCP 宛のパケットを専門家が当時分析したが宛先 IP アドレスと TCP Sequence が等しいことに気が付かなかったものと思われる．一方, 8 月 2 日のデータを用いて訓練した Auto Encoder を, 8 月 2 日と 8 月 3 日に適用した際の Auto Encoder の誤差の分布を図 5.11 に示す．8 月 2 日と比較して, 8 月 3 日には再構成ロスが 5.9 の付近に位置する送信元からのパケットが急増していることが見て取れる．再構成ロスが 5.9 付近に相当する送信元から送信されたパケットを 8 月 3 日に観測されたパケット全体から抽出した所, 図 5.13 に示すように約 56% が Mirai の特徴を有していた．本モデルを用いることで, 新たなゼロデイ攻撃である Mirai を単純に宛先ポートを数える場合と比較して効果的に抽出することが可能であり, Mirai に対してより早期に対処できた可能性が

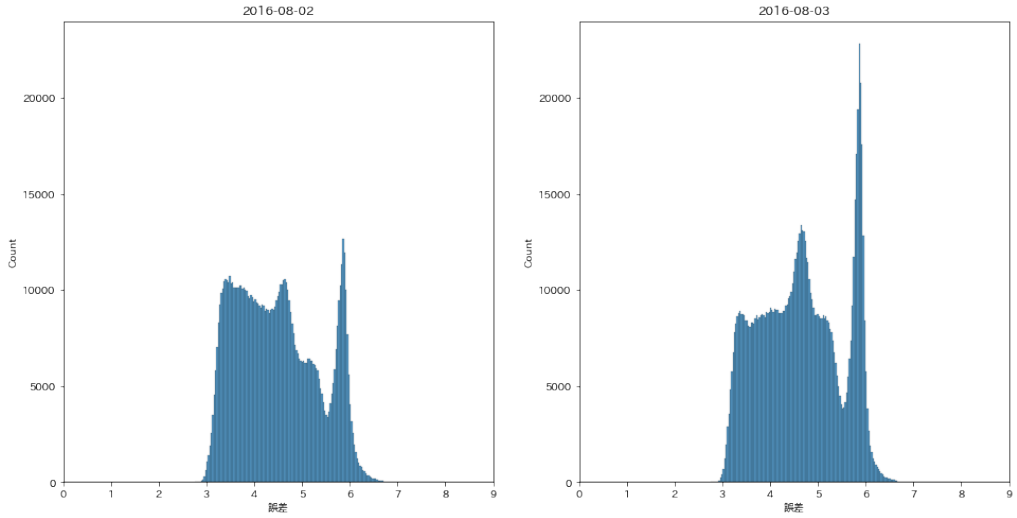

図 5.11: Auto Encoder の出力した誤差の分布

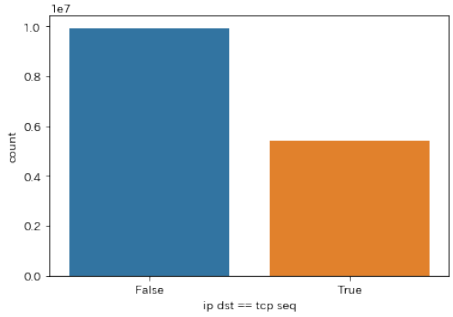

図 5.12: 23 番ポート宛のパケットに含まれる Mirai の割合

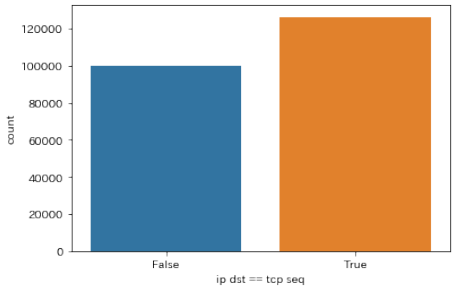

図 5.13: Auto Encoder の誤差分布が大きく変化した点に含まれる Mirai の割合

ある.

5.6.2　インシデントレポートが存在しないが検出された異常

f1 値を最大にするしきい値のとき, インシデントレポートが存在しないにも関わらず Auto Encoder が高いスコアを出力し, 異常と判定した日が 22 日存在した. このうち 12 日は, インシデントレポートが存在した日 ±2 日以内の日付が検出されたものであった. これは, 発生したインシデントが数日ズレて検出されたものと考えられる.

2 月 29 日, 3 月 1 日付近及び 3 月 9 日, 3 月 10 日付近では, 専門家によるインシデントレポートは存在しないものの, Auto Encoder が強く異常と判定している. 3 月 1 日及び 3 月 10 日における, Auto Encoder の前日との誤差が極端に変化した送信元アドレスに対応するパケットの宛先ポートを図 5.14 に示す. また, 比較として 3 月 1 日及び 3 月 10 日で観測されたパケット全体の宛先ポートを図 5.15 に示す. 図 5.15 から, 両日ともに全体としては 23/TCP 宛のパ

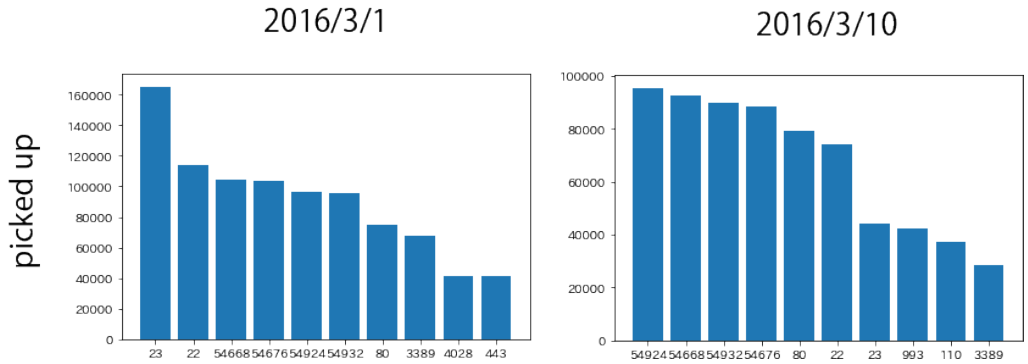

図 5.14: Auto Encoder が検出したパケットの宛先ポート Top 10

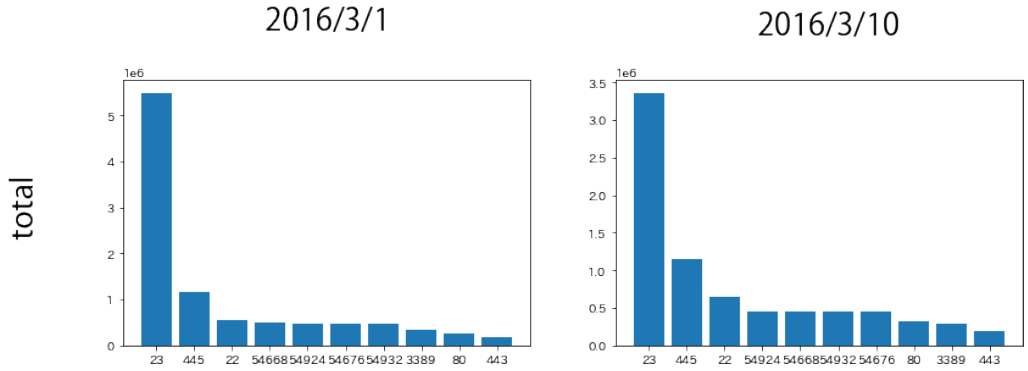

図 5.15: ダークネットで観測されたパケット全体の宛先ポート Top 10

ケットが最も多く, ついで 445/TCP 宛のパケットが多いことがわかる. 一方, Auto Encoder が異常と判定したパケットには, 54668/TCP, 54676/TCP、54924/TCP、54932/TCP 宛のパケットが多いことがわかる. そこで, 図 5.16 にこの 4 つのポート宛のパケット数の遷移を示す. 2 月 29 日と 3 月 9 日の 2 日のみ, 特異的にこの 4 ポート宛のパケット数が激減していることがわかる. したがって、この 4 日において Auto Encoder が異常を検知したのは, このためであると考えられる. この 4 つの宛先ポートは, 2.2.4 で述べたとおり, 2015 年の終盤以降からダークネットで多く観測されるようになったポートである. これらのパケットの原因やスキャンの意図については不明であるが, 何らかの異常が発生していたことは確かなように思われる. インシデントレポートが存在しないものの Auto Encoder が異常と判定した日のうち, 残りの 6 日については異常は明白でなかった.

5.7　実行時間

5.6 で評価した実行時間は 2021 年のハードウェアを用いて 2016 年のデータを分析した結果であり, 今日得られたデータを分析する上で十分な速度を得られているかを評価するのに十分ではない. したがって, 2021 年以降のデータで異常検知を実行し, 速度の評価を行った. 2022 年 1 月前半のデータを用いて実験を行ったところ, モデルの訓練と推論を合わせて平均 50 分/

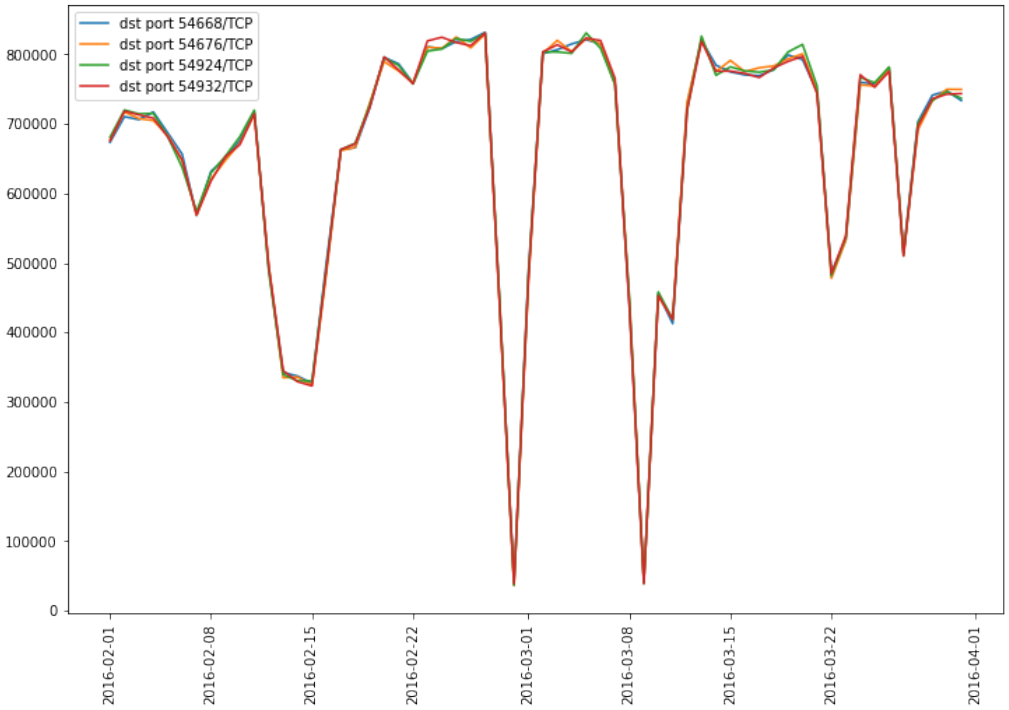

図 5.16: 54668/TCP, 54676/TCP, 54924/TCP, 54932/TCP 宛のパケット数の推移

日で処理が完了した．これは, 1 日分のデータを収集するのに 24 時間が必要であることを考えると十分に高速であると言える．また, このアルゴリズムは全体として $O(n)$ で動作しているため, 最大 24 倍程度の規模の (D)DoS 攻撃のバックスキャッタであれば 24 時間以内に処理できると考えられる．

5.8 まとめ

本章では, 異常検知モデルを実装し, 最適なハイパーパラメータの探索を行った．調整がなされた Auto Encoder を観測データに適用したところ, 専門家によって報告されたインシデントを高速で検出できることが示された．また、観測データやインシデントレポートがモデルにリークを起こしている可能性がある点を考慮する必要はあるものの, 我々の実装した異常検知モデルは異常の原因となるパケットを効果的に選別できることもわかった．

第 6 章

評価

表 6.1: 2018 年における検出結果の概要

f1 値最大 (しきい値が 3.96532)

	Auto Encoder が 正常と判定	Auto Encoder が 異常と判定
レポートが存在する	31 日	17 日
レポートが存在しない	225 日	84 日

recall を重視 (しきい値が 0.92322)

	Auto Encoder が 正常と判定	Auto Encoder が 異常と判定
レポートが存在する	11 日	37 日
レポートが存在しない	97 日	212 日

2018 年にダークネット上で観測されたデータに対して Auto Encoder が出力した異常スコアを図 6.1 に, 検出した異常の概要を表 6.1 に示す. 正解ラベルと比較した際のモデルの AUC は 0.583 であった. また, 2016 年のデータとインシデントレポートをもとに作成した基準値 3.96532, 0.92322 以上のスコアを Auto Encoder が出力した場合にその日付を異常と判定したところ, f 値, recall は表 6.2 に示す値となった。また, 1 日あたりの異常検出に要した時間は平均して 1007 秒であった. f 値, recall ともに, 2018 年の観測データの結果は 2016 年の観測データを上回った. これは, 提案手法がテストデータにおいても有効であり, 観測されるデータの変動が激しくかつ予測不能であるダークネット上のデータから安定して新たな攻撃の発生を検知できることを意味している. 一方, 2016 年と異なり, 2018 年では Auto Encoder の誤差が大きくなる日付が多数存在し, 2016 年と比較して多くの攻撃が新たに発生していることがわかる. これは, 2016 年と 2018 年において, 専門家によるインシデントレポートの数が倍以上に増えていることからも伺える.

表 6.2: 訓練データとテストデータにおけるモデルの性能

		しきい値	
		3.96532	0.92322
2016	f 値	0.21	0.20
	recall	0.12	0.80
2018	f 値	0.26	0.20
	recall	0.15	0.82

6.1 Ground Truth との比較

本節では, 専門家が異常と判定したインシデントに対する Auto Encoder の結果を示す. 表 6.3 に Auto Encoder が検出した異常の概要を示す. インシデントの条件に該当するパケットを送信してきた Host のダークネット全体に占める割合に着目すると, 2016 年で f1 を最大とするようしきい値を定めた時, 1% 未満を占める攻撃 37 件のうち 15 件を検出することに成功した. また、2016 年で recall=0.8 を達成するしきい値を採用した場合は, どの割合においても 8 割以上の検出に成功した. これは, 2016 年において f1 値が最大となるしきい値を採用した際に 10 件中 0 件しか検出できなかったこととは対象的である. 2016 年と比較して 2018 年においては攻撃の手法も対象も多様化しているため, 比較的少数の攻撃であっても検出が可能であったものと思われる.

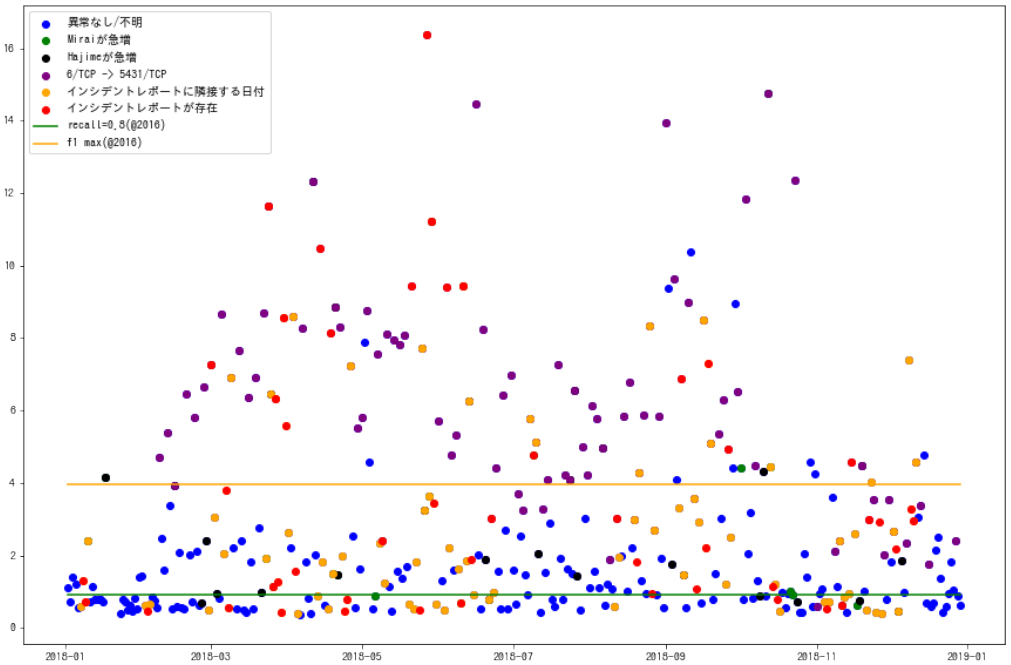

図 6.1: 2018 年における Auto Encoder の異常スコア

表 6.3: Auto Encoder による 2018 年の異常検知

		f1 最大	$recall = 0.8$	total
宛先ポート	23	1 件	1 件	1 件
	80/443	5 件	8 件	11 件
	その他 System Port	1 件	2 件	2 件
	公式 User Port	5 件	13 件	14 件
	非公式 User Port	8 件	15 件	20 件
	動的ポート	1 件	2 件	3 件
該当パケットの割合	1% 未満	15 件	29 件	37 件
	1% 以上 10% 未満	2 件	4 件	6 件
	10% 以上 50% 未満	4 件	8 件	8 件
	50% 以上	1 件	1 件	1 件
Taxonomy	Heavy Port Scan	0 件	0 件	0 件
	Light Port Scan	0 件	0 件	0 件
	Heavy Network Scan	0 件	0 件	0 件
	Light Network Scan	5 件	7 件	8 件
	TCP One Flow	0 件	0 件	0 件
	TCP Backscatter	3 件	7 件	8 件
	Small Syn	5 件	11 件	15 件
	Other	8 件	16 件	20 件
報告	NICT	4 件	8 件	11 件
	JPCERT/CC	1 件	7 件	9 件
	警察庁	19 件	35 件	47 件
	IIJ	6 件	10 件	12 件
計		24 件	45 件	55 件

一方, 2016 年において recall が 0.8 を超えるようにしきい値を定めた時, 2016 年においてはダークネット全体のうち 10% 以上を占めるような大規模な攻撃をすべて検出できていたのに対し, 2018 年においては 1 件のインシデントが検出されなかった. このインシデントは, 2018 年 6 月 10 日に発生した 80/TCP 宛に Mirai の特徴を有するアクセスが急増した事象である [81][82][83]. また, Taxonomy に着目すると, 2016 年においては Small Syn に分類される攻撃をすべて検出できていたのに対し, 2018 年では 15 件中 4 件が検出できず, そのうち 2 件が 80/TCP に関連するインシデントであった. 80/TCP 宛のパケットに関するインシデントで, 該当するパケットがダークネットで観測されたパケット全体に占める割合を表 6.4 に示す. ダークネット上で観測されたパケットの 10% 以上を占める異常で, 80/TCP 宛のパケットが関係するインシデントは 2018 年に 3 件存在したが, そのうち検出できたのはわずか 1 件のみであった. このことから, Auto Encoder は 80/TCP 宛のパケットで発生しているインシデン

表 6.4: 80/TCP 宛のパケットに関するインシデント

src address の割合	f1 最大	recall=0.8	不検出
1% 未満	1 件	1 件	1 件
1% 以上 10% 未満	4 件	6 件	0 件
10% 以上 50% 未満	0 件	1 件	2 件
50% 以上	0 件	0 件	0 件

トの検出は苦手であると考えられる.

6.2 インシデントレポートが存在しないが異常と判断された事象

本節では, Auto Encoder が異常と検出したが. 専門家が異常と示さなかった事象について述べる.

f1 を重視するしきい値においては 84 日が, 誤検知の可能性を上げてでも recall を重視するし

表 6.5: インシデントレポートが存在しないインシデントレポートの概要

異常の概要	日数 (しきい値 3.96532)	日数 (しきい値 0.92322)
インシデントレポート ±1 日以内	16 日	48 日
6/TCP → 5431/TCP	55 日	68 日
Mirai 急増 (レポートが存在するインシデントを除く)	2 日	4 日
Hajime 急増 (レポートが存在するインシデントを除く)	2 日	11 日
不明/誤検知?	9 日	81 日
計	84 日	212 日

きい値を設定した際には 212 日が, インシデントレポートが存在しないにも関わらず異常として判定された. このうち, しきい値が 3.96532 の場合においては 16 日が, 0.92322 においては 48 日が, インシデントレポートが存在する日付 ±1 日の日付であった. これらの日付は, 報告されているインシデントと同様の事象が検出された結果であると考えられる.

2018 年 10 月 12 日には, 2018 年全体で 2 番目に高い異常スコアが Auto Encoder により算出されたが, 該当するインシデントレポートが存在しない. 10 月 11 日のデータを訓練データとして作成した Auto Encoder を 10 月 11 日と 10 月 12 日のデータに対して適用した際の誤

差分布を 6.3 に示す．異常が発生したとされる 10 月 12 日においては, Auto Encoder の誤差が 5.5 の付近と 6.6 の付近に位置するパケットが前の日と比較して多いことがわかる．この誤差に対応するアドレスから送信されたパケットを抽出した所, 宛先ポートが 5431/TCP であり, かつ送信元ポートが 6/TCP であるパケットが大半を占めていた．6/TCP を送信元とし 5431/TCP を宛先ポートとするパケット数の推移を 6.4 に示す．この特徴を示すパケットは, 2 月 8 日頃に出現し, 1 日から 3 日間にかけて 1 日あたり 5 万個程度の IP アドレスからスキャンを行ったあと, 1 日から 3 日間一切の活動を行わないというサイクルを繰り返していることが判明した．この繰り返しにより, ダークネットで観測されるスキャンが 0 から 50000 程度に急増した日で, なんらインシデントレポートが存在しない日は合計 69 日であった．

2013 年に, Universal Plug and Play(UPnP) のオープンソース実装である libupnp に RCE の可能性がある脆弱性が発見された [84]．この脆弱性に影響を受けるデバイスを探索するため, 1900/UDP が空いているデバイスをスキャンにより探す必要があるが, 著名なポートスキャナである nmap において UDP のポートスキャンを行うためには root 権限が必要である．そこで SANS は, root 権限が使用できない環境下で UPnP の脆弱性の影響を受けるデバイスを探すために 5431/TCP をスキャンすることを提案している [85]．多くのマルウェアは辞書攻撃を用いて脆弱なパスワードを使用しているデバイスに侵入するが, これらのログインアカウントの大半は root アカウントとは異なることから, 攻撃者は別途権限昇格を行わない限り root 権限で不正なコードを実行させることができない．従って, 侵害したデバイスで 1900/UDP をスキャンすることはボットネットの管理者にとって手間のかかる作業であり, 代わりに 5431/TCP をスキャンしたものが観測されたのがこの事象ではないかと考えられる．通常はランダムな動的ポートであるはずの送信元ポートが 6/TCP で固定されていることから, なんらかのツールが広まっているものと思われる．スキャンとスキャンの間に数日のインターバルを挟む目的は不明である．ダークネット上で観測される活動には 1 週間単位での周期性が存在することが知られているが [86], このインシデントは明らかに周期が 1 週間単位ではないため, 攻撃者が意図してこのようなインターバルを設けているものと考えられる．理由としては, 防御側による検知を逃れるためなどが考えられる．インシデントレポートが存在しないが Auto Encoder が高い異常値を出力した日について, しきい値を 3.96532 としたときは 84 日中 66 日が, しきい値を 0.92322 としたときは 212 日中 84 日がこの 5431/TCP のスキャンの発生に該当する日付であった．また、5431/TCP のスキャンが出現したにも関わらず検知できなかったのはしきい値 3.96532 のとき 19 日、しきい値 0.92322 のとき 1 日のみであった．我々の知る限りでは, このスキャンについて公に言及されたのは 2018 年 9 月 26 日における NETLAB360 の研究者のツイートが最初であり [87], 半年以上のあいだ探索活動が検知されていなかったことになる．

Mirai は 2016 年に登場したマルウェアであるが, 現在においても多くのデバイスが感染している．Mirai が過去 1 週間と比較して急増した日は 2018 年に, 何らかのインシデントレポートで報告されている他に全部で 7 日存在した．そのうち 2 日が Auto Encoder での異常スコアがしきい値 3.96532 以上であり, さらに 2 日が 0.92322 以上であった．また, 同様に著名なマルウェアである Hajime が急増した日は 2018 年に 15 日あり, そのうち 2 日が Auto Encoder で

の異常スコアがしきい値 3.96532 以上であった．また，さらに 9 日が 0.92322 以上の異常スコアであった．

これらのいずれにも該当しないが Auto Encoder が高い異常を出した日時は，しきい値を 3.96532 としたとき 9 日，しきい値を 0.92322 としたとき 81 日存在したが，これらについては異常の原因は不明であった．これらが誤検知であるのか，それとも何らかの別種の異常を検出しているのかはさらなる研究を行う必要がある．表 6.5 及び図 6.2 に Auto Encoder が異常

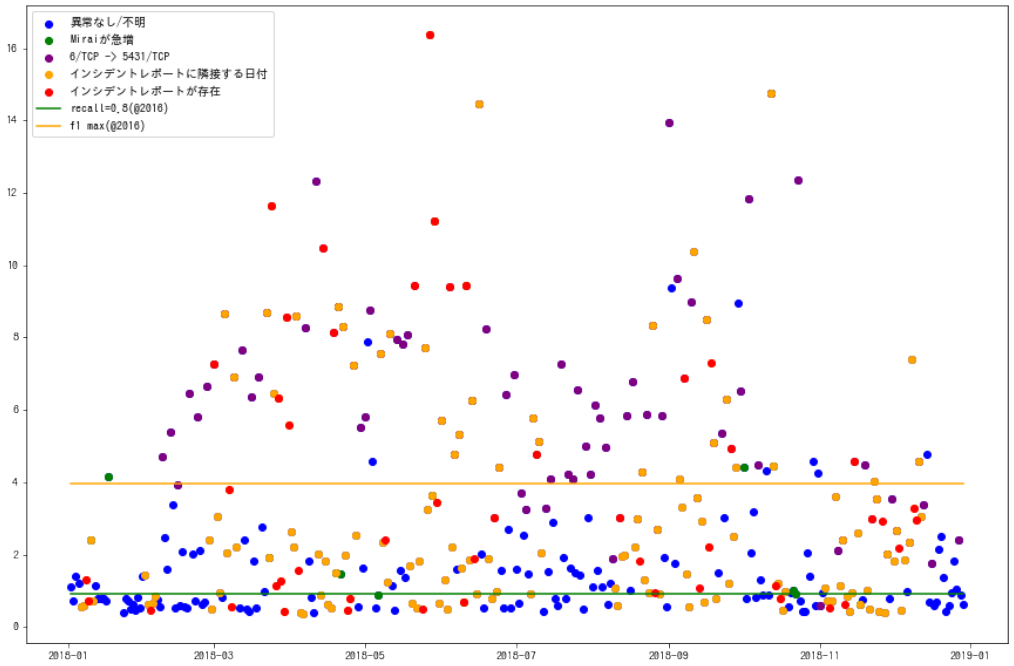

図 6.2: 2018 年の Auto Encoder の出力

として検出し，インシデントレポートが存在しない事象をまとめる．インシデントレポートが存在しないにも関わらず Auto Encoder が異常と判定した日は f1 スコアを重視したしきい値 3.96532 のとき 84 日、検出を重視したしきい値 0.92322 のとき 212 日存在したが，少なくともそれぞれ 75 日/84 日，131 日/212 日は実際に異常が発生していたことがわかる．2018 年における Auto Encoder の AUC は 0.583 と決して高い値では無かったが，これはインシデントが発生したが正解ラベルに含まれていない日付が多数存在したためである．6/TCP を送信元とし，5431/TCP をスキャンする行為を始め，これらのインシデントが専門家により指摘されなかった理由が，専門家が発見できなかった為なのか，それとも発見はしたが何らかの理由により深刻な問題ではないと判断したために公表しなかったのかは不明である．Auto Encoder が発見した異常は，さらなる調査の結果として問題が無かったという結論になる可能性はあるが，いずれにせよ検討が必要な事象を検出したことことは間違いない．

また，本研究の提案手法は，今回観測されたような頻繁に出現と休止を繰り返す攻撃を発生のたびに検知する．これをよしとするかどうかは運用のポリシーによるため，この点についてはさらなる検証が必要である．

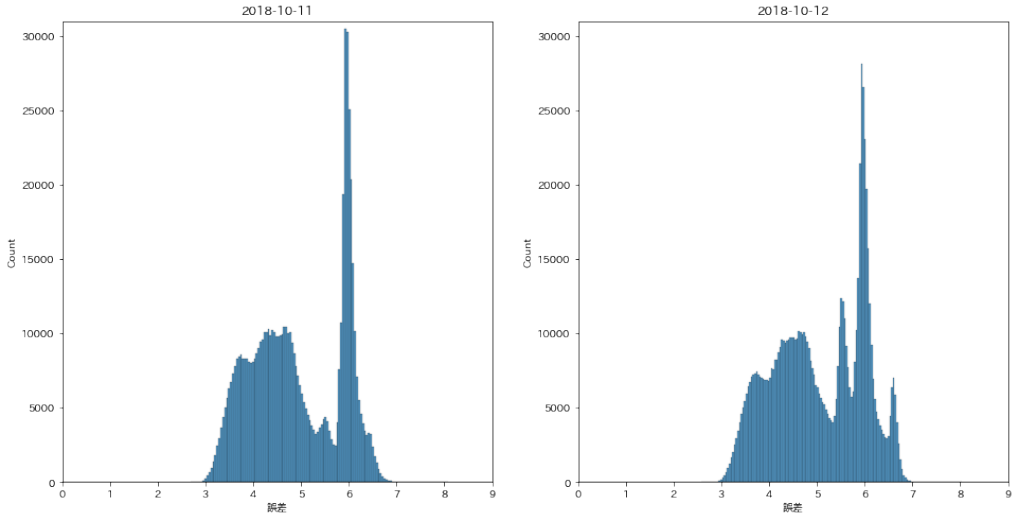

図 6.3: Auto Encoder の出力した誤差の分布

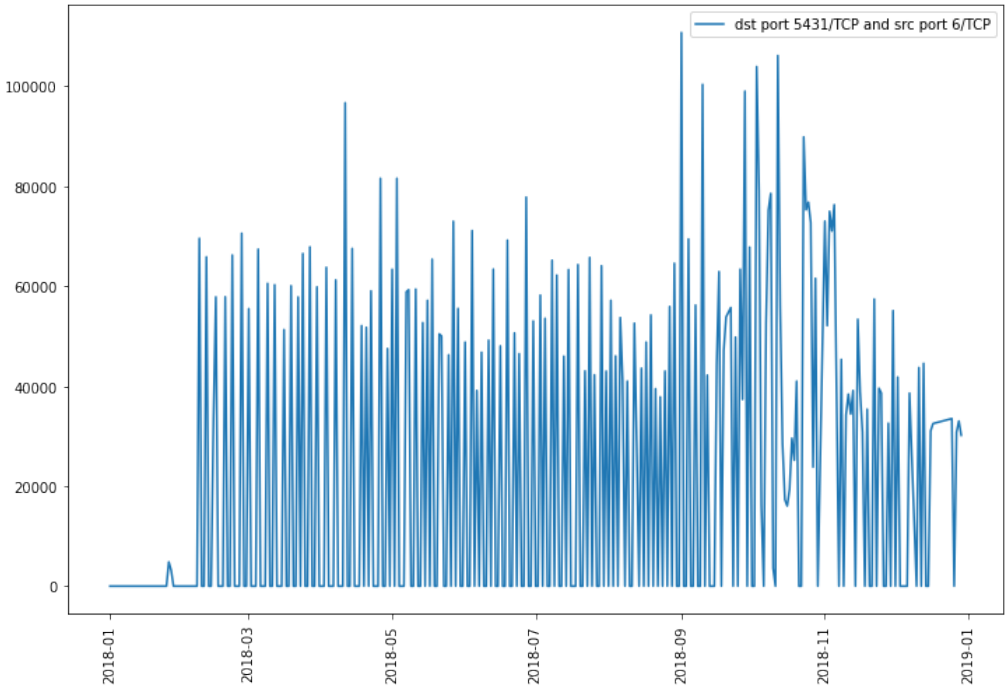

図 6.4: 6/TCP → 5431/TCP のパケットを送信した送信元アドレス数

6.3　まとめ

本章では 2016 年のデータを元に調整した Auto Encoder を用いて, ダークネットで 2018 年に観測されたデータから異常検知を行った. 我々の提案した異常検知モデルは専門家が特定した異常の大半を発見することに成功し, さらに専門家が指摘しなかったインシデントを数多く

発見することに成功した. 観測データの性質が頻繁に変化するダークネットにおいて, 2016 年で調整した Auto Ecnoder を元に 2018 年の異常を検出できたことから, 我々の提案手法は世の中のサイバー攻撃の傾向の変化に強く将来に渡って有効であることが示された.

第 7 章

考察

本章では 2016 年と 2018 年のデータセットの変遷について述べ, Auto Encoder が多くの異常を検知することができた理由について考察する.

7.1 2016 と 2018 の観測データの違い, 共通点

7.1.1 Taxonomy

2016 年と 2018 年にダークネット上で観測された, Taxonomy によるパケットの分類を図 7.1 および図 7.2 に示す. 線が途切れているところは使用したダークネット上でデータが欠損しているところである. 2016 年, 18 年ともに一貫して Small SYN に分類される Host が多

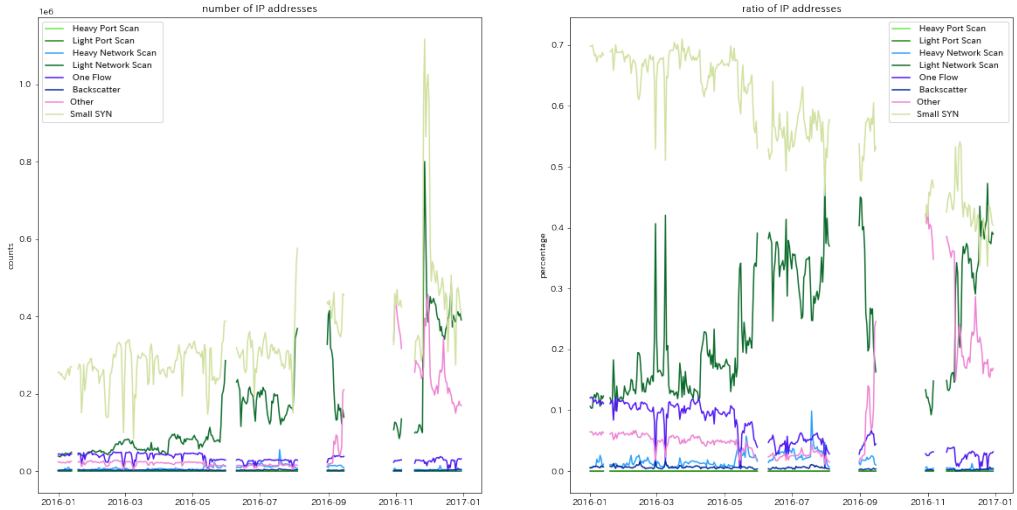

図 7.1: 2016 年度の Taxonomy の実行結果

い. また, 両年を通じて Heavy Scan に該当する Host はほとんど存在しないことがわかる. Scan が Heavy と判定されるためには 1 日・IP アドレスあたり 45 パケット以上のパケットをダークネットが受信する必要があるが, そのような活動を行う Host は極めて少数である.

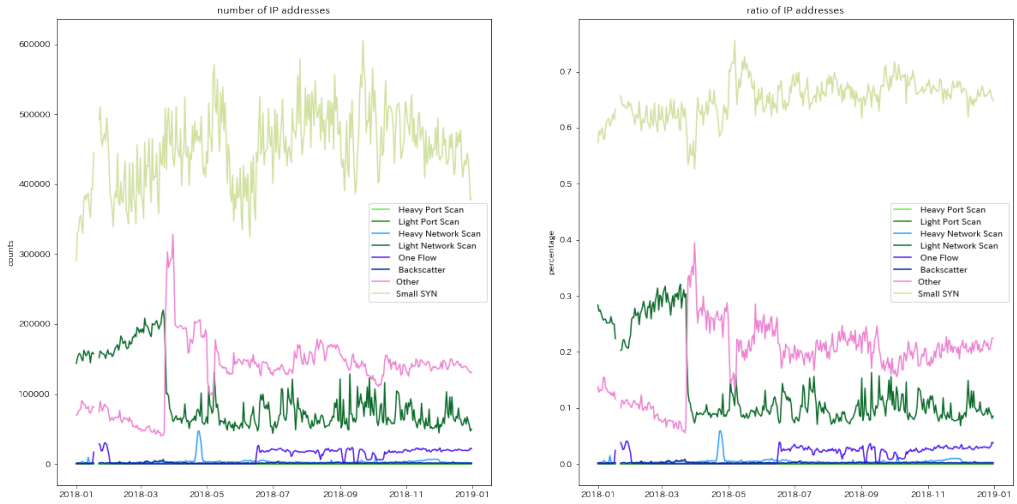

図 7.2: 2018 年度の Taxonomy の実行結果

Backscatter については, (D)DoS 攻撃の一部であるためパケット数は多いが, Backscatter は単一の標的から送られてくるものであるため, Host の数としては少数である. 2 番目に多いクラスは, 2016 年時点では TCP Light Network Scan であったが, 2018 年 3 月頃に Other TCP が急増して以降, Other TCP の割合が TCP Light Network Scan を上回っている. Light Network Scan と Other TCP の主な違いは, 宛先ポートが 1 つかどうかである. Other TCP が TCP Light Network Scan を上回ったのは, ダークネット内の複数のアドレスに対して 2 つまたは 3 つ程度の宛先ポートをスキャンしている Host が増加しているためである. このことから, 特定のポートを観測していても攻撃の傾向を推測することが難しくなっていると考えられ, その時々で適切な特徴量を自動で選ぶ手法が必要であるといえる.

7.2 特徴量選択

本研究においては, TCP/IP ヘッダーに存在する特徴量をすべて使用した異常検知モデルを作成した. しかしながら, 機械学習においては無関係な特徴量をモデルに与えるとモデルの精度が下がることが知られている. したがって, 採用すべき特徴量について議論の余地がある. この点について検証するため, 特徴量を基に Auto Encoder が出力する再構成ロスを勾配 Boosting を用いて教師あり学習で推定し, 決定木の特徴量の重要度を比較した. 図 7.3 に勾配 Boosting を用いて再構成ロスを推定した決定木の feature importance を示す. 誤差項を決定付ける特徴量は, 時期によりかなり変化していることがわかる. これは, 新しいマルウェアやツールが出現した際に, これまでと異なる特徴量の使われ方がされることがあるためであると考えられる. 2016 年 8 月に急拡大したマルウェア Mirai には TCP sequence と宛先 IP アドレスが等しいという特徴があった. 2016 年の 7 月から 8 月にかけて TCP Sequence と宛先 IP アドレスの重要度が急増しているが, それはこのためであると思われる. 他にも 2013 年に開発された高速なスキャナである zmap には IP ID が 54321 で固定されているという特徴がある.

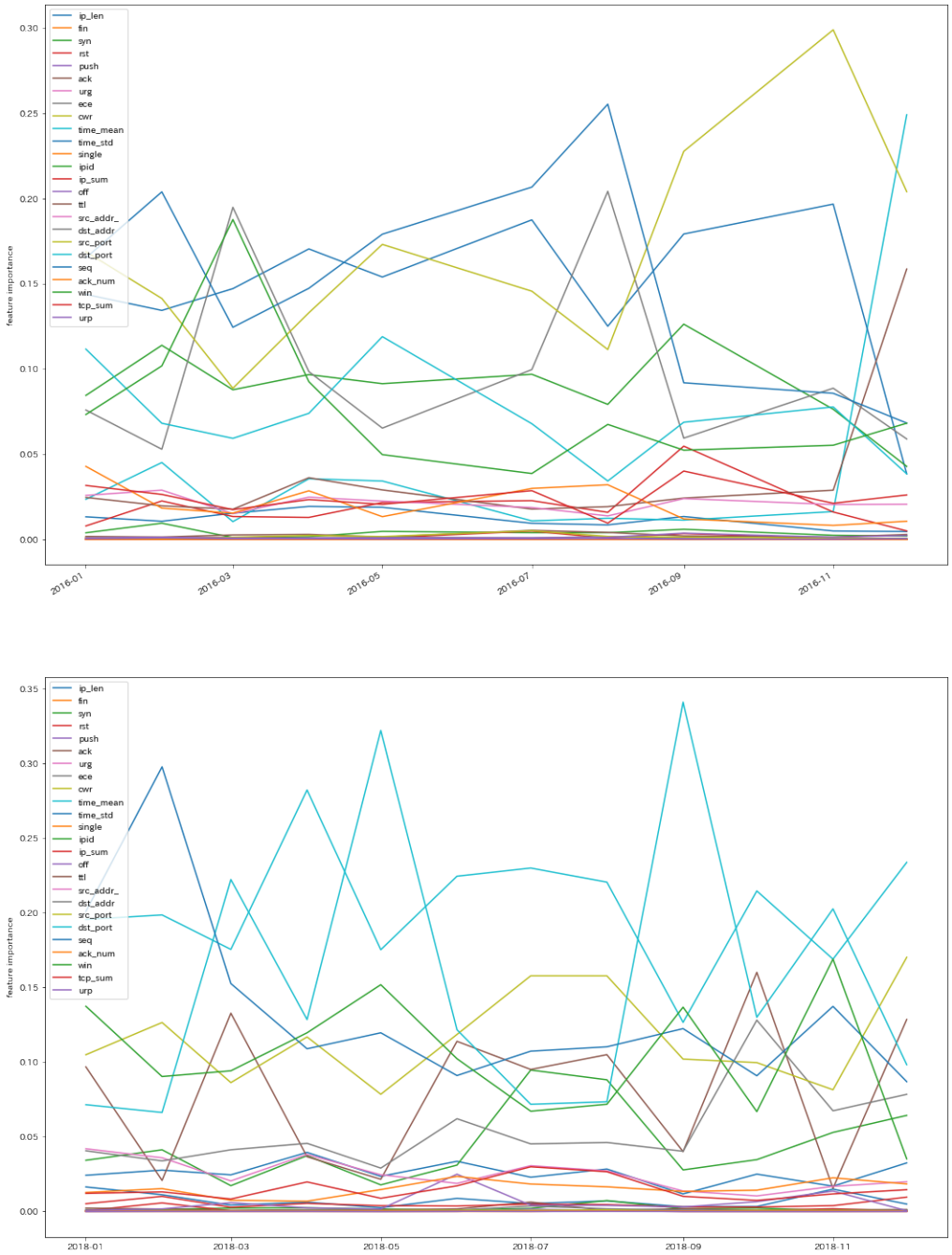

図 7.3: 勾配 Boosting を用いて推定した特徴量の重要度の推移

したがって, 2013 年以前と以後で IP ID の意味合いは大きく変わったものと考えられる. そして, 重要なのは今後もこのような変化が起こりうるということである. 今日重要視されていない特徴量であっても, 将来何らかの固有の特徴として利用できる可能性は常にあり, 今日の結果を基に一部の特徴量を捨てれば将来その特徴量が重要になったときに, 新しい攻撃を見落としかねない. 従って, モデルに与える特徴量を性能評価の結果を基に絞り込むべきではない.

また, 特徴量の意味合いが頻繁に変化するにも関わらず Auto Encoder が新規の攻撃を的確に

検出できていることは, 我々の提案手法がサイバー攻撃の変化に適応できており, 将来に渡っても有効である可能性が高いことを示している.

7.3 まとめ

我々の提案した Auto Encoder は, データセットの特徴に応じて重視すべき特徴量を変えていることが示された. これは, 性質が頻繁に変化するダークネットのデータに適用する上では重要な点である. また, 異常検知を運用するにあたって, 将来のモデルが重視する特徴量を現時点で推定することは将来発生するゼロデイ攻撃を事前に調査することが不可能であるのと同様不可能であるため, TCP/IP ヘッダに存在し利用可能な特徴量はすべて利用するのがよい.

第 8 章

結論

8.1 まとめ

ダークネットではまさに今日インターネット空間で発生している攻撃が観測できるため, ゼロデイ攻撃を含む新規の攻撃が発生した瞬間にそれを収集することが可能である. しかしながら, 観測されるデータが膨大であるため, ゼロデイ攻撃を収集したとしてもそれを認識することは困難であった. そこで本研究では, ダークネットから迅速に新たな攻撃の出現を捉え, 関連するパケットを抽出するシステムを提案した.
そして, このシステムを実装してダークネットのデータに適用し, 同じ時期に専門家がダークネット上のデータを分析した結果と比較した. 我々の提案したシステムは, 専門家が発見したインシデントの多くを見つけ出し, 加えて専門家が長期に渡って見落としていた複数のインシデントを発見することに成功した. また, システムが高速に動くことが確認できた.
ダークネット上で観測されるインシデントの性質は時間とともに変化しており, 単純なルールで攻撃を分析することは困難だが, 我々の提案したシステムはその時々で重要な特徴量を発見し, データの変化に追随できることを示した.

8.2 今後の課題

我々の提案手法は, 新たな異常を検出し, その異常に関係のあるパケットを抽出することに成功した. 一方, その攻撃が具体的に何であったのかの解釈は人間に委ねている. 従って, 重大なインシデントだけでなく, 些細なミスや学術機関などによる悪質でない調査も異常と判定してしまう. また, 提案手法は同一の攻撃が頻繁に活発化と沈静化を繰り返した場合, 活発化のたびに異常と判定した. これもまた, 実用上頻繁にアラートが発生するのは好ましくない. 検出し分離した異常を実用上どのように取り扱うべきかについては今後の課題である.
また, 我々の提案手法は異常検知のために 1 日分のデータを収集する必要があり, このためインシデント発生から異常を検知するまで最低 1 日が必要である. これを短縮することは今後の課題である. 最後に, ダークネットは幅広いサイバー攻撃を観測できる一方, 応答を返さないことから送信元について得られる情報はハニーポットなどと比較して限定的である. ハニーポッ

トや応答型センサをダークネットと組み合わせることでより多くの脅威の情報を得ることができると考えられるが, これらのシステムと我々の提案手法をどのように組み合わせるべきかは今後の課題である. 例えば, ダークネットに適用する異常検知モデルの異常のしきい値を下げることでより多くの異常の可能性がある事象を捉え, それらの事象に対してハニーポットが待受を開始して詳細な情報を取得するようなことが考えられる.

参考文献

[1] Cisco Annual Internet Report (2018-2023) White Paper. Accessed on 2022-1-26.

[2] David Moore, Colleen Shannon, Gm Voelker, and Stefan Savage. Network telescopes: Technical report. *CAIDA, April*, pp. 1–14, 2004.

[3] 国立研究開発法人情報通信研究機構. NICTER 観測レポート 2020. pp. 1–12, 2021.

[4] Claude Fachkha and Mourad Debbabi. *Darknet as a Source of Cyber Intelligence: Survey, Taxonomy, and Characterization*, Vol. 18. 2016.

[5] Noel Chiappa. The ip addressing issue. `https://datatracker.ietf.org/doc/html/draft-chiappa-ipaddressing-00`. Accessed on 2022-1-26.

[6] Daisuke Inoue, Masashi Eto, Katsunari Yoshioka, Shunsuke Baba, Kazuya Suzuki, Junji Nakazato, Kazuhiro Ohtaka, and Koji Nakao. Nicter: An incident analysis system toward binding network monitoring with malware analysis. *Proceedings - WOMBAT Workshop on Information Security Threats Data Collection and Sharing, WISTDCS 2008*, pp. 58–66, 2008.

[7] The CAIDA UCSD Real-Time Network Telescope Data. `https://www.caida.org/data/passive/telescope-near-real-time_dataset.xml`, 2019. Accessed on 2022-1-26.

[8] Joseph O Hara and Supervisor Stefan Weber. *Cloud-based network telescope for Internet background radiation collection.* Ph.d, 2019.

[9] Francesca Soro, Idilio Drago, Martino Trevisan, Marco Mellia, Joao Ceron, and Jose J. Santanna. Are darknets all the same? On darknet visibility for security monitoring. *IEEE Workshop on Local and Metropolitan Area Networks*, Vol. 2019-July, , 2019.

[10] Naoto Sone, Ryouichi Yokota, Ryo Okubo, and Masakatsu Morii. ハニーポットを設置したダークネットのアクセス特性. *Forum on Information Technology*, pp. 197–200, 2013.

[11] Stephen D. Strowes, Emile Aben, René Wilhelm, Florian Obser, Riccardo Stagni, and Agustin Formoso. Debogonising 2a10: : /12: Analysis of one week's visibility of a new /12. In *TMA*, 2020.

[12] Kensuke Fukuda and John Heidemann. Who knocks at the iPv6 door? Detecting IPv6

scanning. *Proceedings of the ACM SIGCOMM Internet Measurement Conference, IMC*, pp. 231–237, 2018.

[13] Jakub Czyz, Kyle Lady, Sam G Miller, Michael Bailey, Michael Kallitsis, and Manish Karir. Understanding IPv6 Internet background radiation. In *Proceedings of the ACM SIGCOMM Internet Measurement Conference, IMC*, pp. 105–118, 2013.

[14] Ruoming Pang, Paul Barford, Vinod Yegneswaran, Vern Paxson, and Larry Peterson. Characteristics of internet background radiation. *Proceedings of the 2004 ACM SIGCOMM Internet Measurement Conference, IMC 2004*, pp. 27–40, 2004.

[15] Karyn Benson, Alberto Dainotti, Kc Claffy, Alex C. Snoeren, and Michael Kallitsis. Leveraging internet background radiation for opportunistic network analysis. *Proceedings of the ACM SIGCOMM Internet Measurement Conference, IMC*, Vol. 2015-Octob, pp. 423–436, 2015.

[16] Claude Fachkha, Elias Bou-Harb, and Mourad Debbabi. Inferring distributed reflection denial of service attacks from darknet. *Computer Communications*, Vol. 62, pp. 59–71, 2015.

[17] Jun Liu and Kensuke Fukuda. Towards a taxonomy of darknet traffic. *IWCMC 2014 - 10th International Wireless Communications and Mobile Computing Conference*, pp. 37–43, 2014.

[18] Eray Balkanli and A. Nur Zincir-Heywood. On the analysis of backscatter traffic. In *39th Annual IEEE Conference on Local Computer Networks Workshops*, pp. 671–678, 2014.

[19] Emile Aben. Conficker/conflicker/downadup as seen from the ucsd network telescope. `https://www.caida.org/archive/ms08-067/conficker/`.

[20] Constantinos Kolias, Georgios Kambourakis, Angelos Stavrou, and Jeffrey Voas. DDoS in the IoT: Mirai and Other Botnets. *Computer*, p. 1, 2017.

[21] 警察庁. 脆弱性が存在するルータを標的とした宛先ポート 52869/tcp に対するアクセス及び日本国内からの telnet による探索を実施するアクセスの観測等について. `https://www.npa.go.jp/cyberpolice/important/2017/201712191.html`, 2017. Accessed on 2022-1-26.

[22] Jun Liu and Kensuke Fukuda. On destination port usage in darknet. *IEICE General Conference 2014*, p. 553, 2014.

[23] Joanne Treurniet. A network activity classification schema and its application to scan detection. *IEEE/ACM Transactions on Networking*, Vol. 19, No. 5, pp. 1396–1404, 2011.

[24] Nevil Brownlee. One-way traffic monitoring with iatmon. In Nina Taft and Fabio Ricciato, editors, *Passive and Active Measurement*, pp. 179–188, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[25] Jun Liu and Kensuke Fukuda. An evaluation of darknet traffic taxonomy. *Journal*

of Information Processing, Vol. 26, pp. 148–157, 2018.

[26] Jean Pierre Van Riel and Barry Irwin. InetVis, a visual tool for network telescope traffic analysis. *ACM International Conference on Computer Graphics, Virtual Reality and Visualisation in Africa*, Vol. 2006, pp. 85–89, 2006.

[27] Takashi Koide, Shogo Suzuki, Daisuke Makita, Kosuke Murakami, Takahiro Kasama, Jumpei Shimamura, Masashi Eto, Daisuke Inoue, Katsunari Yoshioka, and Tsutomu Matsumoto. Detection and Classification Method for Malicious Packets with Characteristic Network Protocol Header. *Css 2014*, pp. 48–55, 2014.

[28] Constantinos Kolias, Georgios Kambourakis, Angelos Stavrou, Jeffrey Voas, and Ieee Fellow. DDoS in the IoT: Mirai and Other botnets. *Computer*, Vol. 50, No. 7, pp. 80–84, 2017.

[29] Nobuaki Furutani, Tao Ban, Junji Nakazato, Jumpei Shimamura, Jun Kitazono, and Seiichi Ozawa. Detection of DDoS backscatter based on traffic features of darknet TCP packets. *Proceedings - 2014 9th Asia Joint Conference on Information Security, AsiaJCIS 2014*, pp. 39–43, 2014.

[30] Siti Hajar Aminah Ali, Seiichi Ozawa, Tao Ban, Junji Nakazato, and Jumpei Shimamura. A neural network model for detecting DDoS attacks using darknet traffic features. *Proceedings of the International Joint Conference on Neural Networks*, Vol. 2016-Octob, No. November 2014, pp. 2979–2985, 2016.

[31] Dr. a. Malathi S. Revathi. A Detailed Analysis on NSL-KDD Dataset Using Various Machine Learning Techniques for Intrusion Detection. *International Jornal of Engineering Research and Technology*, Vol. 2, No. 12, pp. 1848–1853, 2013.

[32] Richard Lippmann, Joshua W. Haines, David J. Fried, Jonathan Korba, and Kumar Das. 1999 DARPA off-line intrusion detection evaluation. *Computer Networks*, Vol. 34, No. 4, pp. 579–595, 2000.

[33] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A. Ghorbani. A detailed analysis of the KDD CUP 99 data set in Computational Intelligence for Security and Defense Applications. *Computational Intelligence in Security and Defense Applications (CISDA)*, No. Cisda, pp. 1–6, 2009.

[34] Raghavendra Chalapathy and Sanjay Chawla. Deep Learning for Anomaly Detection: A Survey. No. January, 2019.

[35] Yalei Ding and Yuqing Zhai. Intrusion detection system for NSL-KDD dataset using convolutional neural networks. *ACM International Conference Proceeding Series*, pp. 81–85, 2018.

[36] Farooq Shaikh, Elias Bou-Harb, Jorge Crichigno, and Nasir Ghani. A Machine Learning Model for Classifying Unsolicited IoT Devices by Observing Network Telescopes. *2018 14th International Wireless Communications and Mobile Computing Conference, IWCMC 2018*, pp. 938–943, 2018.

[37] Robin Sommer and Vern Paxson. Outside the closed world: On using machine learning for network intrusion detection. *Proceedings - IEEE Symposium on Security and Privacy*, pp. 305–316, 2010.

[38] Kajal Rai, M. Syamala Devi, and Ajay Guleria. Packet-based Anomaly Detection using n-gram Approach. *International Journal of Computer Sciences and Engineering*, Vol. 6, No. 5, pp. 366–372, 2018.

[39] Pedro Casas, Johan Mazel, and Philippe Owezarski. Unsupervised Network Intrusion Detection Systems: Detecting the Unknown without Knowledge. *Computer Communications*, Vol. 35, No. 7, pp. 772–783, 2012.

[40] Payam Vahdani Amoli and Timo Hämäläinen. A real time unsupervised NIDS for detecting unknown and encrypted network attacks in high speed network. *Proceedings - M and N 2013: 2013 IEEE International Workshop on Measurements and Networking*, pp. 149–154, 2013.

[41] R. Can Aygun and A. Gokhan Yavuz. Network Anomaly Detection with Stochastically Improved Autoencoder Based Models. *Proceedings - 4th IEEE International Conference on Cyber Security and Cloud Computing, CSCloud 2017 and 3rd IEEE International Conference of Scalable and Smart Cloud, SSC 2017*, pp. 193–198, 2017.

[42] Hyunseung Choi, Mintae Kim, Gyubok Lee, and Wooju Kim. Unsupervised learning approach for network intrusion detection system using autoencoders. *Journal of Supercomputing*, Vol. 75, No. 9, pp. 5597–5621, 2019.

[43] Yang Yu, Jun Long, and Zhiping Cai. Network Intrusion Detection through Stacking Dilated Convolutional Autoencoders. *Security and Communication Networks*, Vol. 2017, , 2017.

[44] Yisroel Mirsky, Tomer Doitshman, Yuval Elovici, and Asaf Shabtai. Kitsune: An Ensemble of Autoencoders for Online Network Intrusion Detection. No. February, pp. 18–21, 2018.

[45] Carlos Garcia Cordero, Sascha Hauke, Max Muhlhauser, and Mathias Fischer. Analyzing flow-based anomaly intrusion detection using Replicator Neural Networks. *2016 14th Annual Conference on Privacy, Security and Trust, PST 2016*, pp. 317–324, 2016.

[46] Sofiane Lagraa, Yutian Chen, and Jérôme François. Deep mining port scans from darknet. *International Journal of Network Management*, Vol. 29, No. 3, pp. 1–20, 2019.

[47] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013 - Workshop Track Proceedings*, pp. 1–12, 2013.

[48] Sofiane Lagraa and JéRome François. Knowledge discovery of port scans from darknet. *Proceedings of the IM 2017 - 2017 IFIP/IEEE International Symposium on*

Integrated Network and Service Management, pp. 935–940, 2017.

[49] Dvir Cohen, Yisroel Mirsky, Manuel Kamp, Tobias Martin, Yuval Elovici, Rami Puzis, and Asaf Shabtai. Dante: A framework for mining and monitoring darknet traffic. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Vol. 12308 LNCS, No. 1, pp. 88–109, 2020.

[50] Yoav Goldberg and Omer Levy. word2vec Explained: deriving Mikolov et al.'s negative-sampling word-embedding method. No. 2, pp. 1–5, 2014.

[51] Luca Gioacchini, Luca Vassio, Marco Mellia, Idilio Drago, Zied Ben Houidi, and Dario Rossi. DarkVec: Automatic Analysis of Darknet Traffic with Word Embeddings. *The 17th International Conference on emerging Networking EXperiments and Technologies (CoNEXT '21), December 7â•fi10, 2021, Virtual Event, Germany*, Vol. 1, pp. 76–89, 2021.

[52] Tao Ban, Lei Zhu, Jumpei Shimamura, Shaoning Pang, Daisuke Inoue, and Koji Nakao. Detection of botnet activities through the lens of a large-scale darknet. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Vol. 10638 LNCS, pp. 442–451, 2017.

[53] Chansu Han, Jumpei Shimamura, Takeshi Takahashi, Daisuke Inoue, Junichi Takeuchi, and Koji Nakao. Real-time detection of global cyberthreat based on darknet by estimating anomalous synchronization using graphical lasso. *IEICE Transactions on Information and Systems*, Vol. E103D, No. 10, pp. 2113–2124, 2020.

[54] Ryoh Akiyoshi, Daisuke Kotani, and Yasuo Okabe. Investigation of A Method for Detecting New Scanning Activities by Correlating Low-Interaction Honeypots with Darknet.

[55] Sadegh Torabi, Elias Bou-Harb, Chadi Assi, El Mouatez Billah Karbab, Amine Boukhtouta, and Mourad Debbabi. Inferring and Investigating IoT-Generated Scanning Campaigns Targeting A Large Network Telescope. *IEEE Transactions on Dependable and Secure Computing*, Vol. XX, No. XX, pp. 1–17, 2020.

[56] Alberto Dainotti, Claudio Squarcella, Emile Aben, Kimberly C. Claffy, Marco Chiesa, Michele Russo, and Antonio Pescape. Analysis of country-wide internet outages caused by censorship. *IEEE/ACM Transactions on Networking*, Vol. 22, No. 6, pp. 1964–1977, 2014.

[57] Ritesh K. Malaiya, Donghwoon Kwon, Sang C. Suh, Hyunjoo Kim, Ikkyun Kim, and Jinoh Kim. An Empirical Evaluation of Deep Learning for Network Anomaly Detection. *IEEE Access*, Vol. 7, pp. 140806–140817, 2019.

[58] 今泉允聡. 深層学習の原理に迫る. 岩波書店, 2021.

[59] Kamesh and N. Sakthi Priya. Evaluation of anomaly ‐ based IDS for mobile devices using machine learning classifiers Dimitrios. *Security and Communication Networks*,

Vol. 5, No. June, pp. 422–437, 2012.

[60] Nadipuram R. Prasad, Salvador Almanza-Garcia, and Thomas T. Lu. Anomaly Detection: A Survey. *Computers, Materials and Continua*, Vol. 14, No. 1, pp. 1–22, 2009.

[61] David H. Wolpert. The Lack of A Priori Distinctions Between Learning Algorithms. *Neural Computation*, Vol. 8, No. 7, pp. 1341–1390, 10 1996.

[62] Wikipedia. Transistor count. `https://en.wikipedia.org/wiki/Transistor_count`. Accessed on 2022-1-26.

[63] Pankaj Mishra, Claudio Piciarelli, and Gian Luca Foresti. A Neural network for image anomaly detection with deep pyramidal representations and dynamic routing. *International Journal of Neural Systems*, Vol. 30, No. 10, 2020.

[64] Ali Narin, Ceren Kaya, and Ziynet Pamuk. Automatic detection of coronavirus disease (COVID-19) using X-ray images and deep convolutional neural networks. *Pattern Analysis and Applications*, Vol. 24, No. 3, pp. 1207–1220, 2021.

[65] Guansong Pang, Ling Chen, Longbing Cao, and Huan Liu. Learning representations of ultrahigh-dimensional data for random distance-based outlier detection. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 2041–2050, 2018.

[66] Nakamoto Satoshi. Bitcoin: A Peer-to-Peer Electronic Cash System. pp. 1–9.

[67] Jacob Walker, Carl Doersch, Abhinav Gupta, and Martial Hebert. An uncertain future: Forecasting from static images using variational autoencoders. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, pp. 835–851, Cham, 2016. Springer International Publishing.

[68] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre Antoine Manzagol. Stacked denoising autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion. *Journal of Machine Learning Research*, Vol. 11, pp. 3371–3408, 2010.

[69] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings*, No. Ml, pp. 1–14, 2014.

[70] 国立研究開発法人情報通信研究機構. Nicter 観測レポート 2016. pp. 1–6, 2017.

[71] 国立研究開発法人情報通信研究機構. Nicter 観測レポート 2017. pp. 1–11, 2018.

[72] 国立研究開発法人情報通信研究機構. Nicter 観測レポート 2018. pp. 1–10, 2019.

[73] 国立研究開発法人情報通信研究機構. Nicter 観測レポート 2019. pp. 1–13, 2020.

[74] 国立研究開発法人情報通信研究機構. Nicter blog observing cybersecurity through darknet. `https://blog.nicter.jp/`. Accessed on 2022-1-26.

[75] 国立研究開発法人情報通信研究機構. Nicter 解析チーム. `https://twitter.com/nicter_jp`. Accessed on 2022-1-26.

[76] JPCERT/CC. Jpcert/cc インターネット定点観測レポート. https://www.jpcert.or.jp/tsubame/.

[77] IIJSOC チーム. wizsafe security signal. https://wizsafe.iij.ad.jp/. Accessed on 2022-1-26.

[78] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019.

[79] Pierluigi Paganini. More than 900k routers of deutsche telekom german users went offline. http://securityaffairs.co/wordpress/53871/iot/deutsche-telekom-hack.html.

[80] Manos Antonakakis, Tim April, Michael Bailey, Matthew Bernhard, Ann Arbor, Elie Bursztein, Jaime Cochran, Zakir Durumeric, J Alex Halderman, Ann Arbor, Luca Invernizzi, Michalis Kallitsis, Merit Network, Zane Ma, Joshua Mason, Damian Menscher, Chad Seaman, Nick Sullivan, Kurt Thomas, Yi Zhou, Manos Antonakakis, Tim April, Michael Bailey, Matthew Bernhard, Elie Bursztein, Jaime Cochran, Zakir Durumeric, J Alex Halderman, Luca Invernizzi, Michalis Kallitsis, Deepak Kumar, Chaz Lever, Zane Ma, Joshua Mason, Damian Menscher, Chad Seaman, Nick Sullivan, Kurt Thomas, and Yi Zhou. Understanding the Mirai Botnet. *USENIX Security*, pp. 1093–1110, 2017.

[81] JPCERT/CC. Jpcert/cc インターネット定点観測レポート [2018 年 4 月 1 日 6 月 30 日], 2018. https://www.jpcert.or.jp/tsubame/report/TSUBAMEReport2018Q1.pdf.

[82] 警察庁. 宛先ポート 80/tcp に対する mirai ボットの特徴を有するアクセスの増加について, 2018. https://www.npa.go.jp/cyberpolice/detect/pdf/20180613.pdf.

[83] NMasaki Kubo Jumpei Shimamura. 80/tcp 宛通信の増加.

[84] HD Moore. Security flaws in universal plug and play: Unplug, don't play. https://www.rapid7.com/blog/post/2013/01/29/security-flaws-in-universal-plug-and-play-unplug-dont-play/, 2013. Accessed on 2022-1-26.

[85] Johannes Ullrich. Exposed upnp devices. https://isc.sans.edu/diary/ExposedUPNPDevices/15040, 2013. Accessed on 2022-1-26.

[86] Akira Saso, Tatsuya Mori, and Shigeki Goto. Darknet Traffic Analysis by Using Source Host Classification. *Graduate*, Vol. 1, No. October, 2013.

[87] LIU Ya. https://twitter.com/liuya0904/status/1044960012072697856. Ac-

cessed on 2022-1-26.

発表文献

(1) 古田 陸太, 福田 健介, 江崎 浩 ”Detecting trend changes in packets observed in darknet by Auto Encoder” WIDE 研究会

謝辞

本論文の執筆をするにあたり, 大変多くの方にご指導ご協力をいただきました. ここに心よりの感謝の意を評します.
指導教官の江崎浩教授には, 学部 4 年生から今までの 3 年間の研究室生活において数多くの的確なご指導をいただきました. 自分が追い詰められていたときに先生に励まして頂いたおかげで, ここまで努力することができました. 心から感謝申し上げます.
国立情報学研究所 福田健介准教授には, 研究に関して細部まで面倒を見ていただきました. お忙しいなかで毎週ミーティングに時間を割いていただき, 実験環境の準備や研究のデータセットの用意から研究の方向性や進め方に至るまで見放すこと無く指導していただきました. 修士論文が完成したのは先生のおかげです. 心から感謝申し上げます.
国立情報学研究所 小林諭博士には, 毎週のミーティングで研究のアドバイスをいただきました. 研究の相談や文章の添削等に何度も付き合っていただき, 大変助けになりました. 心から感謝申し上げます.
東京大学大学院情報理工学系研究科 落合秀也准教授には学部生のころから研究内容から研究室生活まで様々な面倒を見ていただきました. 先生のご指導の元で参加した国際学会は大変貴重な経験となりました. 心から感謝申し上げます.
東京大学大学院情報理工学系研究科 塚田学准教授には研究室内発表等の場で的確なコメントやアドバイスをいただきました. 心から感謝申し上げます.
最後に研究生活や学生生活においてご指導, ご支援下さったすべての皆様に感謝致します.

2022 年 1 月 27 日 古田 陸太

付録 A

宛先ポート

ここでは、2008 年 2021 年の間にダークネット上で頻繁に観測された TCP パケットの宛先ポートについて記す. 宛先ポート上で動く主なアプリケーションは IANA を参考にしたが, 必ずしもターゲットとなるアプリケーションと一致するとは限らない.

表 A.1: ダークネット上で多く観測されるポート一覧

年号	宛先ポート番号	割合	主なアプリケーション
2008	2186	15.2%	Unknown
	445	12.1%	Microsoft-DS SMB file sharing
	9100	6.1%	PDL Data Stream
	1433	3.4%	Microsoft SQL Server
	23	3.4%	Telnet
	80	3.3%	HTTP
	22	3.2%	SSH
	4662	2.2%	OrbitNet Message Service
	135	1.6%	DCE endpoint resolution
	4899	1.4%	Radmin remote administration tool
2009	445	80.4%	Microsoft-DS SMB file sharing
	135	1.3%	DCE endpoint resolution
	1433	1.1%	Microsoft SQL Server
	2186	1.0%	Unknown
	80	0.8%	HTTP
	22	0.7%	SSH
	23	0.7%	Telnet
	8080	0.5%	HTTP alternate
	9100	0.5%	PDL Data Stream

	2967	0.5%	Unknown
2010	445	7.4%	Microsoft-DS SMB file sharing
	5900	0.3%	Virtual Network Computing remote desktop protocol
	22	0.2%	SSH
	2186	0.1%	Unknown
	1433	0.1%	Microsoft SQL Server
	9100	0.1%	PDL Data Stream
	1024	0.1%	Unknown
	3072	0.1%	Unknown
	9415	0.0%	Unkown
	139	0.0%	NetBIOS Session Service
2011	445	5.4%	Microsoft-DS SMB file sharing
	3389	0.4%	Remote Desktop Protocol
	2222	0.4%	Unkown
	22	0.3%	SSH
	80	0.2%	HTTP
	1433	0.1%	Microsoft SQL Server
	9100	0.1%	PDL Data Stream
	2186	0.1%	Unknown
	23	0.1%	Telnet
	3910	0.1%	Unknown
2012	445	12.9%	Microsoft-DS SMB file sharing
	23	2.7%	Telnet
	3389	1.1%	Remote Desktop Protocol
	80	0.8%	HTTP
	22	0.7%	SSH
	2222	0.6%	Unknown
	2186	0.6%	Unknwon
	25	0.3%	SMTP
	210	0.3%	ANSI Z39.50
	1433	0.2%	Microsoft SQL Server
2013	445	3.9%	Microsoft-DS SMB file sharing
	22	0.6%	SSH
	3389	0.4%	Remote Desktop Protocol
	80	0.4%	HTTP
	1433	0.3%	Microsoft SQL Server
	8080	0.3%	HTTP alternate

	23	0.2%	Telnet
	25	0.1%	SMTP
	2186	0.1%	Unknown
	135	0.1%	DCE endpoint resolution
2014	23	3.8%	Telnet
	445	2.0%	Microsoft-DS SMB file sharing
	22	1.1%	SSH
	3389	0.5%	Remote Desktop Protocol
	80	0.4%	HTTP
	443	0.3%	HTTPS
	8080	0.3%	HTTP alternate
	1234	0.2%	Mercurial and git default ports for serving Hyper Text
	1433	0.2%	Microsoft SQL Server
	3128	0.2%	Web cache
2015	23	2.7%	Telnet
	445	1.1%	Microsoft-DS SMB file sharing
	54668	0.6%	Unknown
	54676	0.6%	Unknown
	54924	0.6%	Unknown
	54932	0.6%	Unknown
	22	0.4%	SSH
	80	0.4%	HTTP
	8080	0.2%	HTTP alternate
	3389	0.2%	Remote Desktop Protocol
2016	23	9.1%	Telnet
	7547	3.0%	CPE WAN Management Protocol
	5432	1.0%	PostgreSQL
	2323	0.6%	Unknown(マルウェア Mirai が 23/TCP とともに 2323/TCP をスキャンした影響と考えられる)
	5555	0.5%	Android Debug Bridge
	445	0.5%	Microsoft-DS SMB file sharing
	22	0.3%	SSH
	3389	0.2%	Remote Desktop Protocol
	54676	0.2%	Unknown
	54932	0.2%	Unknown
2017	23	5.5%	Telnet
	2323	2.3%	Unknown(Mirai?)

	1433	0.6%	Microsoft SQL Server
	22	0.6%	SSH
	445	0.3%	Microsoft-DS SMB file sharing
	8545	0.2%	Unknown
	52869	0.2%	Huawei HG532[21]
	3389	0.1%	Remote Desktop Protocol
	80	0.1%	HTTP
	54924	0.1%	Unknown
2018	23	1.3%	Telnet
	445	0.8%	Microsoft-DS SMB file sharing
	81	0.3%	Tor
	1433	0.2%	Microsoft SQL Server
	80	0.2%	HTTP
	22	0.2%	SSH
	8080	0.1%	HTTP alternate
	3389	0.1%	Remote Desktop Protocol
	2323	0.1%	Unknown(Mirai?)
	8088	0.1%	Unknown
2019	23	1.0%	Telnet
	1433	0.4%	Microsoft SQL Server
	445	0.3%	Microsoft-DS SMB file sharing
	26	0.2%	Unassigned
	80	0.2%	HTTP
	8545	0.2%	Unknown
	22	0.1%	SSH
	8080	0.1%	HTTP alternate
	5555	0.1%	Android Debug Bridge
	3389	0.1%	Remote Desktop Protocol
2020	23	0.7%	Telnet
	445	0.2%	Microsoft-DS SMB file sharing
	1433	0.1%	Microsoft SQL Server
	80	0.1%	HTTP
	81	0.1%	Tor
	10443	0.1%	Unknown
	22	0.1%	SSH
	3389	0.1%	Remote Desktop Protocol
	8080	0.1%	HTTP alternate

	5555	0.1%	Android Debug Bridge
2021	23	0.6%	Telnet
	6379	0.5%	Redis
	10443	0.2%	Unknown
	22	0.2%	SSH
	80	0.2%	HTTP
	445	0.2%	Microsoft-DS SMB file sharing
	443	0.1%	HTTPs
	1433	0.1%	Microsoft SQL Server
	81	0.1%	Tor
	5555	0.1%	Android Debug Bridge