異常同期性推定に基づくマルウェア活動の早期検知フレームワークの検討

韓 燦洙[1,a)]　竹内 純一[2]　高橋 健志[1]　井上 大介[1]

概要：グローバルにサイバー攻撃が蔓延する中，攻撃傾向を迅速に捉え，対策を講じることが求められている．マルウェアの感染が拡大する際には，ダークネットに時空間パターンの同期性が観測されることが分かっている．そこで本論文では，ダークネットトラフィックの時空間パターンの同期性をリアルタイムかつ自動的に推定し異常検知を行う 3 つの独立した機械学習手法を提案する．各提案手法を一つのフレームワークに統合し，マルウェア活動の検知精度の定量的な評価実験を行った結果，各提案手法の弱みを相互補完できることが分かり，再現率 100%を達成した．また，提案フレームワークは世間にマルウェア活動が明らかになった時期より平均 153.6 日早く検知した．最後に人的分析コスト評価を行い，一人の分析者が約 14.6 時間で日々のオペレーションを遂行できることを明らかにした．本研究成果はマルウェア活動の迅速な対応の取り組みに貢献できるよう，社会実装を進めている．

キーワード：マルウェア活動，時空間パターン，異常同期性推定，ダークネット

Study on the Early Detection Framework for Malware Activity Based on Anomalous Synchronization Estimation

Chansu Han[1,a)]　Jun'ichi Takeuchi[2]　Takeshi Takahashi[1]　Daisuke Inoue[1]

Abstract: As cyberattacks are spreading globally, it is necessary to grasp the attack tendency and devise countermeasures promptly. It is known that the synchronization of spatiotemporal patterns is observed on the darknet when malware infection spreads. In this paper, we propose three independent machine learning methods that automatically estimate the synchronization of spatiotemporal patterns of darknet traffic in real-time and detect anomalies. Quantitative evaluation experiments on the detection accuracy of malware activities reveal that our framework can complement the weaknesses of each proposed model and achieve a recall rate of 100%. In addition, the proposed framework detects malware activities on average 153.6 days earlier than when malware activity became apparent in the world. Finally, we clarify that an analyst can perform a daily operation in about 14.6 hours.

Keywords: malware activity, spatiotemporal pattern, anomalous synchronization estimation, darknet

1. 序論

近年インターネット上で無差別なサイバー攻撃が膨大に観測されており，その解析に多大なコストがかかっている．グローバルなサイバー攻撃傾向を迅速に把握し，原因を特定して対策を練り，世の中に注意喚起することが求められる．第一歩として，ある攻撃が大流行する前段階に行われるマルウェア起因の無差別型スキャン攻撃活動を早期検知することが重要である．

ところで，マルウェアによるスキャン攻撃を通常のネットワークで特定することは非常に難しい．そこで，未使用の IP アドレス空間，ダークネットは正常通信が観測されないため SN 比 (信号雑音比) が高く，グローバルなサイバー攻撃の傾向を容易に把握できる．ただ，ダークネットで観測される通信は年々指数関数的に増加傾向であり，観測されるのは初期通信のみであるため意図のわからない通信が多い．例えば，調査目的のス

[1] 国立研究開発法人情報通信研究機構
National Institute of Information and Communications Technology
[2] 九州大学，Kyushu University
[a)] han@nict.go.jp

キャン活動や突発的に 1 宛先 IP アドレスに集中する原因不明な通信，誤設定通信など，攻撃とは関連のない通信も多く観測される．そのようなノイズ通信を攻撃通信と区別する必要がある．

スキャンモジュールをシェアする同類のマルウェアに感染したホストらは，次の感染ターゲットを探索するために，同様な時空間パターンでスキャンを行う傾向が見られる [1]．このような傾向は当然ダークネット上でも観測される [2]．ここで，ある期間に観測されたパケットに対して，送信元ホストの分布や宛先ポート番号の分布を空間特徴と呼び，その空間特徴の時間的変化に見られる特徴を時空間パターンと呼ぶ．同様の時空間パターンでスキャンを送信したホスト，その時の宛先ポートらを同期したと表現する．我々はこのような同期性に着目し，大規模なダークネット上で時空間パターンに同期性の高い送信元ホストグループを推定することで，潜在的なマルウェア活動の検知を試みた．同期性に着目することで，ダークネットトラフィックにおけるノイズ通信を削ぎ落とす効果が期待できる．また，明示的なスパイクが見えない，または小規模の脅威，複雑な脅威，定常的な脅威など，従来の人手による作業では追うことが困難だったマルウェア活動を，異常に同期した空間特徴を検知することで，マルウェア感染が大流行する前に捉えることができる．

本論文では，ダークネットトラフィックの時空間パターンの同期性をリアルタイムかつ自動的に推定し異常検知を行う 3 つの独立した機械学習手法と，それらを一つのフレームワークに統合した「*Dark-TRACER*」を提案する．単位時間当たりのダークネットトラフィックから時空間特徴を表す観測データ行列 (テンソル) を作成し，次のような 3 つの異なるアルゴリズムをそれぞれ独立して適用する．1 つ目に，スパース構造学習アルゴリズムの *Graphical Lasso* [3] を用いるモジュール「*Dark-GLASSO*」[4,5] を紹介する．空間特徴の単位時間当たりのパケット数を変数として，空間特徴変数間の条件付き独立性を推定し，全変数同士の同期性の程度を測った後，他の時間帯の同期性の程度と比べて異常検知を行う．2 つ目と 3 つ目に，非負値行列因子分解 (*NMF*) [6] を用いて 2 次元データを扱うモジュール「*Dark-NMF*」[7] と，非負値 Tucker 分解 (*NTD*) [8] を用いて 3 次元データを扱うモジュール「*Dark-NTD*」 [9] を紹介する．これらは，潜在的な頻出パターンを複数のグループの重ね合わせに分解し，各グループで同期性の強い空間特徴変数を異常検知する．

本研究において各提案モジュールの性能をそれぞれ評価し，モジュール間の関係性と実用性を示すことは大きな課題の一つである．マルウェア活動の検知精度の定量的な評価のために，手動で作成した 2018 年 10 月の確実なマルウェア活動の正解表を用いて，各モジュールで見逃しと誤検知を最小限にするパラメータチューニングを行った．従来手法の *ChangeFinder* [10] と，提案モジュールの *Dark-GLASSO*，*Dark-NMF* は既に評価結果を発表済みで，*Dark-NTD* は今回初めて同様な基準で評価を行う．その結果，*Dark-GLASSO*，*Dark-NMF*，*Dark-NTD* はそれぞれ 97.1%，100%，97.1%の再現率 (Recall) を達成した．また，各モジュールの強みと弱みを明らかにし，全提案モジュールを一つのフレームワーク *Dark-TRACER* に統合することで，各々の弱みを相互補完できることが分かった．

さらに本論文では，マルウェア活動の感染拡大時期が明確に分かる事象 (2019 年 6 月~2020 年 10) の正解表を手動で新たに作成し，早期検知の実現可能性評価に用いた．その結果，我々のダークネット観測網 NICTER で脅威が観測され始めた時期より平均 126.4 日早く検知でき，世の中に脅威が明らかになった時期より平均 153.6 日早く検知できた．また，人的分析コストの評価を行い，一人の分析者が 1 ポート当たり 15 分の分析時間を要すること仮定すると，約 14.6 時間で日々のオペレーションが可能であることが分かった．

現在，*Dark-TRACER* は実運用に向けた社会実装を行っている．CSIRT や SOC など組織に，検知したグローバルなマルウェア活動情報を提供し，その原因究明や詳細分析など迅速な対策を講じることが期待される．今後は誤検知を減らし再現率・適合率を共に向上させる仕組みや，第 3 者の脅威情報等を自動的に関連付ける [11] など応用面で幅広く考え，*Dark-TRACER* を拡張していきたい．最後に本論文の貢献をまとめる．

- ダークネットトラフィックの時空間パターンの同期性をリアルタイムかつ自動的に推定し異常検知を行う 3 つの独立した機械学習手法と，それらを一つのフレームワークに統合した *Dark-TRACER* を提案する．
- *Dark-NTD* のパラメータチューニングを行い，全モジュールの検知精度の定量的な比較評価を行う．*Dark-TRACER* は各モジュールの弱みを相互補完できることが分かり，再現率 100%を達成した．
- *Dark-TRACER* は世間脅威が明らかになった時期より平均 153.6 日早く検知した．また，一人の分析者が約 14.6 時間で日々のオペレーションを遂行できることを明らかにした．

2. 提案フレームワーク

Dark-TRACER の全体フレームワークを図 1 に示す．時空間特徴の同期性推定には，*Graphical Lasso* [3]，*NMF* [6]，*NTD* [8] の 3 つのアルゴリズムを用いており，それぞれのモジュールを区別するために *Dark-GLASSO* [4,5]，*Dark-NMF* [7]，*Dark-NTD* [9] と呼ぶ．以下に，図 1 の左から順番に説明する．

2.1 データ観測

Dark-TRACER はダークネットトラフィックデータを解析対象とする．序論で言及したように，ダークネットでは正常通信は観測されないため SN 比が高いという利点があるが，ダークネットに届く通信全てがマルウェアによる悪性通信ではない．ダークネットで観測される通信の中には，Shodan や Censys のような調査目的のスキャン活動や原因不明で突発的に 1 宛先 IP アドレスに集中する通信，誤設定通信など攻撃とは関連のない通信などがある．そのようなノイズ通信を削ぎ落として，本質的な攻撃，マルウェア活動を検知するフレームワークが *Dark-TRACER* である．

我々 NICT では，無差別型サイバー攻撃の大局的な動向を把握することを目的とした大規模ダークネット観測システム NICTER

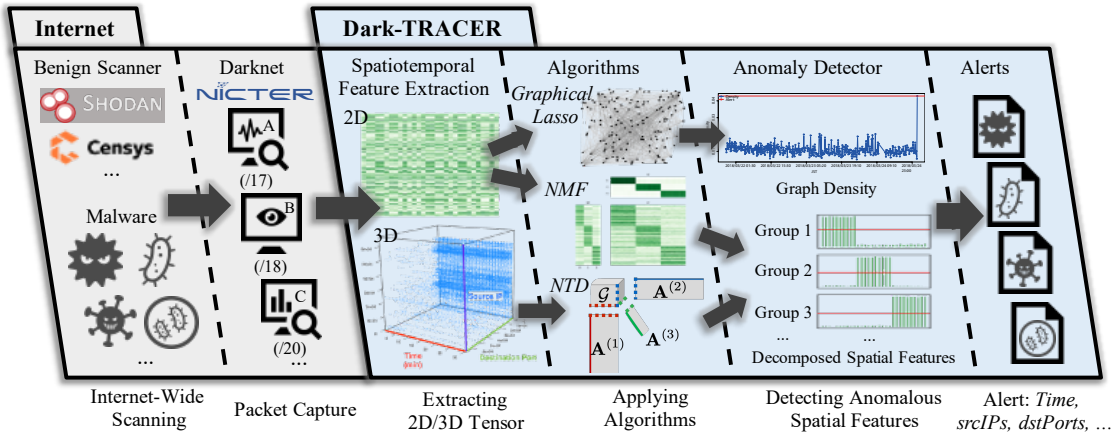

図 1　Dark-TRACER のフレームワーク

プロジェクト[*1]を実施している．様々な国・組織にダークネット観測システム (センサ) を設置し，現在合計で約 30 万 IP アドレスを観測している．ダークネットセンサは，設置される地理的位置や観測規模によって，観測されるデータがやや異なってくる．そのため，*Dark-TRACER* はセンサ別に解析を行う．

次にデータ前処理として，ダークネットに届く TCP-SYN 以外の TCP パケットは，攻撃スキャンとは考えられないため，*Dark-TRACER* では TCP-SYN パケットのみを解析する．また，送信元ホストの単位は，IP アドレスの上位 16bit までを使用する．これはホストを地域や組織レベルでまとめることを意味する．最後に未知のマルウェア活動の観測をより際立てるため，既知で定常的に観測される脅威ポートは除外する．

2.2　時空間特徴抽出

ある期間 (T 秒間) のダークネットトラフィックデータを用意する．そのダークネットトラフィックデータに，N_h 個の送信元ホストのユニーク数と，N_d 個の宛先ポートのユニーク数が観測されたとする．次に，T/M 秒のサンプリング間隔でパケット数を送信元ホストもしくは宛先ポート毎に計数し，これを空間特徴変数と呼ぶ．ここで M はハイパーパラメータである．以上より，観測データから時空間特徴を表す 3 種類のテンソル $V_h \in \mathbb{N}_0^{M\times N_h}$，$V_d \in \mathbb{N}_0^{M\times N_d}$，$V_{hd} \in \mathbb{N}_0^{M\times N_h\times N_d}$ が作成される ($\mathbb{N}_0 = \{0, 1, 2, \cdots\}$)．また，この特徴抽出は t 秒毎にリアルタイムかつ逐次的に処理していく．

2.3　アルゴリズム適用

本節では，スパース構造学習アルゴリズムの *Graphical Lasso* [3]，非負値テンソル分解アルゴリズムの非負値行列因子分解 (Nonnegative Matrix Factorization, *NMF*) [6] と非負値 Tucker 分解 (Nonnegative Tucker Decomposition, *NTD*) [8] の要点を簡略に紹介する．アルゴリズムの詳細は各アルゴリズムの論文，または各モジュール *Dark-GLASSO* [4, 5]，*Dark-NMF* [7]，*Dark-NTD* [9] の論文を参照されたい．

また *Dark-TRACER* では，上記のアルゴリズム以外にも時空間特徴の同期性を推定できるような手法を適用可能であるが，手法に合った異常検知方法を考えなければならない．

2.3.1　*Graphical Lasso*

Graphical Lasso (パッケージ名: *glasso*[*2]) はスパース構造学習手法であり，見かけ上の相関ではなく，変数同士の「本質的な関係」を求めることができる．ここで，2 つの変数間に「本質的な関係はない」とは，他の変数を与えたとき，その 2 つの変数は条件付き独立であることと同値である．多変量正規分布を用いる構造学習モデル，ガウス型グラフィカルモデルにおいて上記の問題は，精度行列 (共分散行列の逆行列) を推定する問題に帰着する．*Graphical Lasso* は，ℓ_1 正則化項付き最尤推定を行い，スパースな精度行列を求めることで変数間の関係性にスパース性を取り入れている．

以上より求められた精度行列を，図 1 の *Graphical Lasso* 箇所で示しているように，無向グラフで表現することができる．ノード集合は変数に，エッジ集合は変数間の「関係の有無」を表す．つまり，ある変数間に関係がないとき，その変数が対応するノード間にエッジは引かれない．逆に，関係がある場合は，エッジが引かれる．

***Dark-GLASSO* モジュール:** *Dark-GLASSO* では，*Graphical Lasso* を用いて時空間特徴行列 (V_h または V_d) から空間特徴変数同士の本質的な関係を求め，グラフ化する．これは，変数同士の同期性を表現していると解釈できる．

2.3.2　非負値テンソル分解

テンソル分解は，行列・テンソルから潜在的な頻出パターンを複数のグループの重ね合わせに分解する手法であり，テンソルの階数や分解方法によって幾つかのモデルが提案されている．第 2.2 節で生成されるテンソルは，負の値を取らないものである．分解結果を現実に即した形で解釈性のあるものにするために，非負制約を加えたテンソル分解手法，*NMF* と *NTD* を用いる．*NMF* は 2 階のテンソル (行列) の分解手法，*NTD* は D 階のテンソル (ここでは $D = 3$) の分解手法である．*NTD* は *NMF* を高次元へ拡張したものと見ることができる．次に手法の適用過程をモジュール別に簡略に説明する．

[*1] https://www.nicter.jp/

[*2] https://cran.r-project.org/web/packages/glasso/

***Dark-NMF* モジュール:** 図 1 の *NMF* 箇所で示しているように，*NMF* は非負値行列 $V \in \mathbb{N}_0^{M \times N}$ を 2 つの非負値行列 $W \in \mathbb{R}^{M \times r}$, $H \in \mathbb{R}^{r \times N}$ の積に近似分解 ($V \approx WH$) する手法である．ここで r は基底ベクトルの数を意味し，分解するパターンの数を指す ($r < N, M$)．*NMF* は次の誤差関数 (フロベニウスノルム) $||V - WH||_F^2$ (s.t. $W \geq 0$, $H \geq 0$) の最小化を行う．*NMF* の近似分解アルゴリズムにはいくつか提案されているが，我々は Lee ら [6] が提案した乗法型更新アルゴリズムを用いた．このアルゴリズムでは，W, H の初期値を適当に与え，誤差関数が最小になるまで W, H を交互に更新し最適化を行う．*Dark-NMF* では特異値分解の値を初期値として用いる．

まとめると，*Dark-NMF* では，*NMF* を用いて時空間特徴行列 (V_h または V_d) を 2 つの行列 W, H に近似分解し，同期したと思われる潜在的な時空間特徴変数グループが各行列で基底の数だけ分かれる．分解される 2 つの行列 W, H の各要素は次のように解釈できる．

W: **時間特徴**．各基底ベクトルは時間に対する通信量の変化パターン．

H: **送信元ホスト空間特徴または宛先ポート空間特徴**．各基底ベクトルは同期して通信を行うと思われる送信元ホストの特徴．または，同時に通信を受ける宛先ポート番号の関係．

***Dark-NTD* モジュール:** 図 1 の *NTD* 箇所で示しているように，*NTD* は D 階のテンソルを，1 つの小さなテンソルと，複数の行列として分解する．*Dark-NTD* では $D = 3$ 階のテンソルを扱い，1 つの小さなテンソル $\mathcal{G}$ と，3 つの行列 $\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \mathbf{A}^{(3)}$ に分解する．分解対象のテンソル $\mathcal{V} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ の分解後の式は $\mathcal{V} \approx \mathcal{G} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \times_3 \mathbf{A}^{(3)}$ のように表せる．ここで，I_1, I_2, I_3 は各軸 (モード) の長さを，$\mathbf{A}^{(n)} \in \mathbb{R}^{I_n \times R_n}$ ($n \in \{1, 2, 3\}$), $\mathcal{G} \in \mathbb{R}^{R_1 \times \cdots \times R_3}$ で，$\times_i$ はモード i 方向の積を表す．$\mathcal{G}$ はコアテンソルと呼ばれ，各モードの基底ベクトルにかかる重み，関係の強さを表す．また，R_1, R_2, R_3 はランクで，それぞれのモードで何本の基底ベクトルを抽出するかを定めており，元のデータにおける各軸の潜在的なグループの数のようなものともいえる．

NTD は次の誤差関数 (フロベニウスノルム) $||\mathcal{V} - \mathcal{G} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \times_3 \mathbf{A}^{(3)}||_F^2$ の最小化を行う．$\mathcal{G}, \mathbf{A}^{(n)}$ を交互に更新し最適化を行う．しかし，分解対象のテンソル $\mathcal{V}$ が大規模になると，厳密な計算を行うためには膨大なメモリと計算量が必要となり，分解が実質不可能である．そのため，*Dark-NTD* では *FSTD* [12] により分解対象のテンソル $\mathcal{V}$ を予め低ランク近似し，その結果をもとに *NTD* を高速に近似計算する *LRA-NTD* [13] を用いることで，分解の精度を落とす代わりに省メモリかつ高速に行っている．この高速化の詳細は *Dark-NTD* の先行論文 [9] を参照されたい．

まとめると，*Dark-NTD* では，高速化を考慮した *NTD* を用いて 3 階の時空間特徴テンソル V_{hd} をコアテンソル $\mathcal{G}$ と 3 つの行列 $\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \mathbf{A}^{(3)}$ に近似分解し，同期したと見られる潜在的な時空間特徴変数グループが各行列で基底の数だけ分かれる．この分解結果の各要素は次のように解釈できる．

$\mathbf{A}^{(1)}$: **時間特徴**．各基底ベクトルは時間に対する通信量の変化パターン．

$\mathbf{A}^{(2)}$: **送信元ホスト空間特徴**．各基底ベクトルは同期して通信を行うとされる送信元ホストの特徴．

$\mathbf{A}^{(3)}$: **宛先ポート空間特徴**．各基底ベクトルは同時に通信を受ける宛先ポート番号の関係．

2.4 異常検知

本節では各アルゴリズムの適用結果から，空間特徴変数の異常検知を行う方法をモジュール別に紹介する．

2.4.1 *Dark-GLASSO* モジュール

求めた精度行列のグラフから，変数間の同期の程度をグラフ密度 $|E|/N(N-1)$ で数値化する．ここで，$|E|$ はエッジ集合の要素数，N は空間特徴変数の数を示す．グラフ密度は 1 に近いほど全変数同士が関係しあっていることを指す．連続した期間で T 秒ごとの観測データからグラフ密度値を算出し，逐次的に記録していく．グラフ密度値の時系列データが決まった窓サイズ (K) の期間分集まると外れ値検知を行う．その方法を簡潔に説明すると，時系列データ中の最大要素を除いた場合の分散と除かない場合の分散を計算し，その割合が閾値を超えると外れ値であると判定し，時系列データから削除する．閾値を超えなくなるまで次の最大要素で逐次的に判定していく．もし，外れ値が出ずに時系列データの数が決まった窓サイズ (K) を超えると，古いデータを順に削除する．以上より，他の時間帯のグラフ密度値と比べて異常なグラフ密度値を持つ時間帯を判定できる．

2.4.2 *Dark-NMF* モジュール

行列 W, H の値は一意に定まっていない．そのため，まず行列 W, H の値のスケールを正規化する．行列 W の各列の総和が 1 になるように，各列の総和の逆数を対角成分として持つ対角行列 $\Lambda \in \mathbb{R}^{r \times r}$ を用いて，$W = W\Lambda$, $H = \Lambda^{-1}H$ で正規化する．正規化した行列 H の要素の値は実際の観測されたパケット数とスケールが揃う．行列 H の値が 1 未満の空間特徴は，該当基底ベクトルにおいて活動していないノイズとみなす．その反面，行列 H の値が 1 以上の空間特徴は，該当基底ベクトルにおいて活動が観測されたとみなす．そのようなアクティブな空間特徴らにおいて，要素の最大値 (最大パケット数) の α (%) を超える空間特徴が β 個以上あるとき，その空間特徴らは異常であると判定する．また，異常と判定する空間特徴を，アクティブな空間特徴ら全てを対象にするか ($f = 0$)，もしくはその中でさらに異常な空間特徴だけを絞るか ($f = 1$) を決める．以上より，ある時間帯における異常な空間特徴らを判定できる．

2.4.3 *Dark-NTD* モジュール

$\mathbf{A}^{(2)}$ について閾値を超える値をもつホストが 2 つ以上存在すれば，そのホスト群 (空間特徴) は同期して活動したとみなし，IP アドレスを記録する．また，このホスト群が通信を行った宛先ポートを特定するために $\mathcal{G}$，$\mathbf{A}^{(3)}$ を用いる．まず $\mathcal{G}$ から，活動ありと判定された $\mathbf{A}^{(2)}$ のホスト群と結ばれている $\mathbf{A}^{(3)}$ のポート群 (空間特徴) を特定する．そのポート群で閾値以上の値を持つ宛先ポート番号を，先程のホスト群の攻撃ポートであると判定する．以上より，ある時間帯における異常なホスト群と

その攻撃ポートを判定できる．

2.5 アラート処理

最終的に各モジュールで異常だと判定された情報を集め，一様な形式のアラートを出力する処理を行う．*Dark-GLASSO* からは異常と判定された時間帯のデータを，*Dark-NMF* からは異常と判定された空間特徴らに関するデータを用いる．そのデータの中で，多数の送信元ホストらが特定の宛先ポートへ多くパケットを送信している場合，時刻と宛先ポート番号，送信元ホスト情報を集約してアラートを出力する．最後に *Drak-NTD* からは異常と判定されたホスト群，攻撃ポート，時刻の情報を用いて，そのままアラートを出力する．

3. 検知精度の定量的な比較評価

各提案モジュールの性能をそれぞれ評価し，モジュール間の関係性と実用性を示すための 2 通りの実験とその結果を述べる．本節の実験では，各モジュールに対してマルウェア活動の検知精度の定量的な評価を行う．第 4 節では，2 つ目の実験としてマルウェア活動の早期検知の実現可能性評価を行う．全ての実験は日本標準時に統一して行った．

2018 年 10 月において，明らかに「マルウェア活動が観測された TCP ポート」の正解表を手動で作成し，合計 35 個の TCP ポートが対象になった．本正解表での評価は，各モジュールで誤検知があっても見逃しは最小限にするハイパーパラメータセットを見つけ，その時の検知精度を評価することを目標にしている．従来手法の *ChangeFinder* と，*Dark-GLASSO*，*Dark-NMF* は既に先行論文 [5,7] で発表済みで，*Dark-NTD* は今回新たに同様な基準で評価する．以下にデータセット，*Dark-NTD* のパラメータチューニング，各モジュールの比較結果の順に述べる．

3.1 データセット

用いたデータセットと正解表は，*Dark-GLASSO* や *Dark-NMF* と同様のものであり，データを公開している[*3]．具体的には，世界各地に設置されていて，観測規模の異なる 8 つのダークネットセンサ A からセンサ H までのデータを用いた．各センサの観測規模は，約 3 万 IP アドレス規模 (/17) から約 2 千 IP アドレス規模 (/21) までのもので，合計で約 8 万 IP アドレス規模のダークネット観測網である．また，正解表となった合計 35 個の TCP ポート番号のみ，以下に紹介する．この正解表の詳細は先行論文 [5,7] を参照されたい．

正解表 (35 TCP ポート): *21, 25, 82, 83, 84, 85, 88, 110, 443, 444, 1701, 2004, 2480, 5358, 5379, 5431, 5900, 5984, 6379, 7379, 7547, 8000, 8001, 8010, 8081, 8088, 8181, 8291, 8443, 8888, 9000, 23023, 37215, 49152, 65000.*

3.2 *Dark-NTD* のパラメータチューニング

ここでは *Dark-NTD* における次のような 5 つのハイパーパラメータのチューニング方法を述べる．

[*3] https://csdataset.nict.go.jp/darknet/

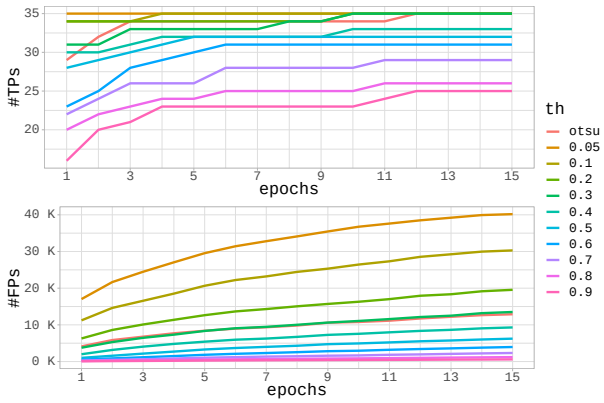

図 2 8 つの全センサを用いたときの th 別の#TPs，#FPs 結果

(1) sensor: どのセンサで観測されたデータを用いるか
(2) $\tilde{R}_n$: 高速化のための低ランク近似手法 *FSTD* の基底数
(3) R_n: *NTD* の基底数
(4) epochs: 同じデータに対して何回計算を繰り返すか
(5) th: アラート判定の閾値

上記の 5 つのハイパーパラメータに対して，グリッドサーチによるチューニングを行った．

続いて，各パラメータのグリッドサーチの範囲を述べる．sensor は，用いた 8 つのダークネットセンサの中で一つ選ぶか，もしくは 8 つ全てを用いるかで試した．次に，*FSTD* の基底数 $\tilde{R}_n$ は，値が大きければ大きいほど元のテンソルの情報を落とさず低ランク近似できる．そして $\tilde{R}_n$ は *NTD* の基底数 R_n より大きく設定すべきである．本実験では，$\tilde{R}_n \in \{25, 49, 81, 121\}$ と $R_n \in \{3, 5, 8\}$ の範囲で試した．さらに，*FSTD* と *NTD* の初期値をランダムに選択しているため，計算結果が一意ではない．そのため，同じデータに対して何度繰り返し計算すれば，十分な精度が安定して得られるのかを知る必要がある．そのための epochs を本実験では，15 回まで繰り返し計算を行う．最後にアラート判定の閾値 th は，動的な方法として「大津の 2 値化」[14] と，固定値として $\{0.05, 0.1, 0.2, \cdots, 0.9\}$ の範囲で試した．

それでは，上記の 5 つのパラメータチューニングを行う．5 つ全てを同時にチューニングすると，その組み合わせ数が膨大となり現実的ではない．戦略として，*NTD* に直接関わる $\tilde{R}_n, R_n$ と，そうでない sensor・epochs・th で，2 手にパラメータを分けてチューニングすることにした．手順として，検知精度に sensor・epochs・th がより大きく影響を与えるため，先にこの 3 つで検知精度の大雑把な調整を行った後，$\tilde{R}_n, R_n$ で微調整を行う．また本実験では，第 2.2 節における時空間特徴抽出は，観測単位 $T = 1800$ 秒，サンプル数 $M = 30$，逐次的処理単位 $t = 600$ 秒に設定し，2018 年 10 月 (1 ヶ月間) のテンソル V_{hd} を作成して用いた．

sensor・epochs・th のチューニング評価: 先行論文で経験的に用いていた値，$\tilde{R}_n = 25, R_n = 5$ に固定した上で，sensor・epochs・th のチューニング評価を行った．評価の結果を図 2, 3 に示す．横軸は epochs，縦軸はポート番号の正解数 (#TPs)，誤検知数 (#FPs) を表す．#TPs が 35 に近いほど，かつ#FPs が少ないほど精度が良い．図 2 は，8 つの全センサを用いたときの th 別の結果を示す．ここで th はどの固定値よりも大津の 2 値

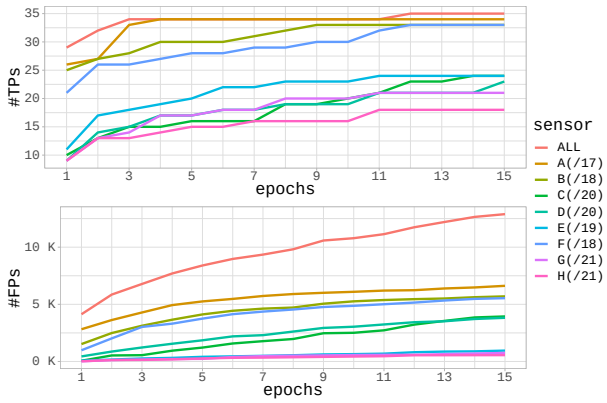

図 3 th が大津の 2 値化のときの sensor 別の#TPs，#FPs 結果

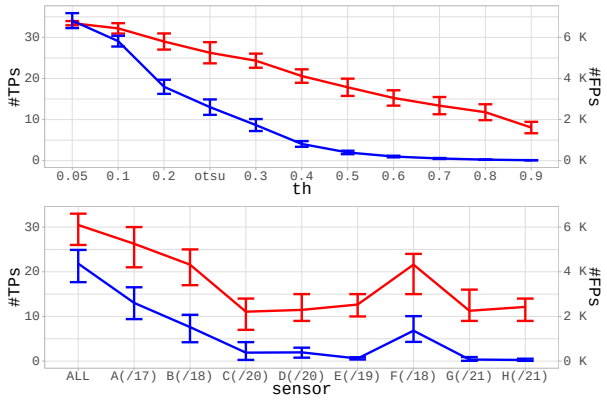

図 4 15 回の epochs ごとの#TPs(赤)，#FPs(青) のバラツキ結果 (上が sensor A のときの th 別の結果．下が th が大津の 2 値化のときの sensor 別の結果．)

化 (otsu) を用いるときに，#FPs を比較的低く抑えながら少ない epochs で高い#TPs を達成していることが分かる．次に図 3 は，th が大津の 2 値化のときの sensor 別の結果を示す．センサ A だけを用いるときに，全センサを用いるときと比べて，#FPs を低く抑えながら少ない epochs で同等な#TPs に達していることが分かる．以上より，sensor は A のみを，epochs は 4 で th は大津の 2 値化が最適であると判断した．

次に，各パラメータに関する素朴な考察を述べる．図 4 は，15 回の epochs ごとの#TPs，#FPs のバラツキを調べたものである．赤線が epochs ごとの#TPs の平均，青線が#FPs の平均，エラーバーはその標準偏差である．この結果から，*FSTD* と *NTD* の初期値選択のランダム性は，毎回同程度の#TPs，#FPs を記録していて，検知精度に劇的な影響は与えていないと判断した．sensor に関しては，*Dark-NTD* は観測規模の大きいセンサで良い#TPs を記録することが分かった．最後に th で大津の 2 値化は，固定値の 0.3 と同等な程度の精度であるが，データから適応的にしきい値を判定する点に利点がある．

$\tilde{R}_n, R_n$ のチューニング評価: sensor・epochs・th を前述で決めた値に設定した上で，$\tilde{R}_n, R_n$ のチューニング評価を行った．その結果を表 1 に示す．期待とは裏腹に，$\tilde{R}_n$ の値を大きくすると，#TPs は減少，#FPs は増加，平均処理時間も増加することが分かった．したがって，$\tilde{R}_n = 25$ が適切であると判断した．

表 1 *Dark-NTD* における $\tilde{R}_n, R_n$ のチューニング評価結果

$\tilde{R}_n$ \ R_n	#TPs			#FPs			Time
	3	5	8	3	5	8	(sec.)
25	31	34	33	3931	4932	4814	61.5
49	33	34	32	4916	5898	6235	148.9
81	32	32	33	6085	7120	6893	328.5
121	32	33	33	6441	7353	7153	638.7

表 2 マルウェア活動の検知精度の定量的な比較評価結果

Modules		#TPs	#FNs	#FPs	Recall
ChangeFinder [10]		24	11	0	68.6%
Dark-GLASSO [5]		34	1	0	97.1%
Dark-NMF [7]	SET1	31	4	9	88.6%
	SET2	35	0	1074	100%
	SET2′	35	0	519	100%
Dark-NTD	従来 [9]	29	6	4131	82.9%
	Tuned	34	1	4932	97.1%
	Tuned′	30	5	3026	85.7%

R_n は平均処理時間に大きな変動はなく，$R_n = 5$ が適切であると判断した．

3.3 検知精度の比較評価

本節では各モジュールのマルウェア活動の検知精度を定量的に比較評価する．その結果を表 2 に示す．*ChangeFinder* [10] は *Dark-TRACER* の各モジュールを提案する以前に NICTER で用いてた既存手法であり，時系列データの変化点を検出するアルゴリズムである．10 分間のパケット数，ユニークな送信元ホスト数の 2 種類の時系列データへ *ChangeFinder* を適用した．本実験で設定した各モジュールのパラメータを述べる．

ChangeFinder : 自己回帰 (AR) モデルの次数=2，忘却パラメータ=0.005，平滑化の範囲 (2 段階)=10, 5，異常検知の閾値=3
Dark-GLASSO : $T = 600$, $M = 12$, $t = 600$, V_h 利用, $K = 432$, $\theta = 0.98$, $\lambda = \{0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$, $\gamma = 1000$
Dark-NMF : $T = 1800$, $M = 30$, $t = 600$, (V_h, V_d) 利用, $\alpha = 30$, $\beta = 2$, $r = \{1, 2, \cdots, 10\}$, SET1 で $f = 0$, SET2 で $f = 1$
***Dark-NTD* (Tuned)** : $T = 1800$, $M = 30$, $t = 600$, V_{hd} 利用, sensor=A, $\tilde{R}_n = 25$, $R_n = 5$, epochs=4, th=大津の 2 値化

ここで，*Dark-GLASSO* における θ は異常検知時の閾値であり，λ は正則化係数である．また，モジュールの中で *Dark-GLASSO* が最も計算量が高いことから，リアルタイム性を保つためにホスト数 N_h が γ 個を超える場合，ランダムサンプリングする．他のパラメータは全て第 2 節で説明済みである．

表 2 の *Dark-NMF* における SET は，f の設定の違いを表す．*Dark-NTD* における「従来」とは，先行論文でのパラメータ設定であり，Tuned が今回のチューニングで決めた設定である．2 つの違いは，Tuned が epochs を設けている点と sensor A だけを用いる点である．ここで，*Dark-NTD* の Tuned のみが sensor A を用い，他は 8 つのセンサ全てを用いていることに注意して欲しい．#FNs は見逃し数で，#TPs+#FNs=35 となる．Recall

表 3 マルウェア活動の早期検知の実現可能性評価のための正解表の一部 (RCE は *Remote Code Execution*，C&C は *Command and Control*，DL は *Download*，DDoS は *Distributed Denial-of-Service*，CVE は *Common Vulnerabilities and Exposures*)

脅威イベント	TCP ポート	NICTER 観測時期と規模	公表時期	観測された脅威の特徴
ECHOBOT [15] (Mirai 亜種)	*1220,6666,9080*	*2019-07-11 16:00, 100→200*	*2019-08-06*	Richard というファイルをダウンロード. 50 以上の *RCE* 等の *exploits* を発見.
MOOBOT [16] (Mirai 亜種)	*60001*	*2019-06-24 09:00, 50→4K*	*2019-09-27*	オリジナル Mirai とは異なる独自の暗号化手法, C&C 通信プロトコル，感染経路を持つ. C&C・DL サーバーを共有する特徴がある. 様々な脆弱性とポートをターゲットに, ゼロデイ攻撃や DDoS 攻撃等を活発に行う.
	9527,34567	*2019-07-11 08:00, 50→8K*		
	81,88,8000	*2019-08-30 00:00, 1K→5K*		
	82,83,85,8081,9090	*2019-09-04 00:00, 1K→10K*		
	1588,8888	*2019-09-19 00:00, 100→16K*		
	84,5984,8181,9200	*2019-09-21 11:00, 1K→12K*		
BlueKeep [17] (CVE-2019-0708)	*3389*	*2019-12-13 10:00, 3.5K→4.5K*	*2019-05-14*	Microsoft により CVE が公開. ウィンドウサイズが *8192* 固定.
...	...	...	...	...

は再現率であり，#TPs / (#TPs+#FNs) である．

最後に，SET2′ や Tuned′ に付いている「′」を説明する．SET2 や Tuned の結果を見ると#FPs がとても多い．主な原因は調査目的スキャナによる同期したスキャンであった．この問題を一時的に解決するために，SET2 と Tuned のアラート結果に，単純なルールを適用して調査目的スキャナによるアラートを除外することを試みた．単純なルールとして，同時刻のアラートで同じ送信元ホストから大量または連番の TCP ポートが見られる場合，そのアラートを除外した．その結果が SET2′，Tuned′ である．*Dark-NMF* は#TPs を保ちつつ#FPs を半減する効果があったが，*Dark-NTD* ではそのような効果がなかった．

表 2 の比較評価の結果から，*Dark-TRACER* は各モジュールの結果を統合することで#FPs は多少あるものの，再現率 100% を達成した．次に，各モジュールで見逃したポートの特徴を見て考察を行う．*ChangeFinder* はホスト規模の小さい，または短期間，または長期定常的なマルウェア活動の検知に弱い傾向がある．*Dark-GLASSO*，*Dark-NMF* はホスト規模の小さいマルウェア活動の検知に弱い傾向がある．さらに SET1 は長期定常的な活動の検知にも弱い傾向がある．*Dark-NTD* は短期間のマルウェア活動の検知に弱い傾向がある．この結果から総合的に見ると，3 つの提案モジュールを統合することで，各エンジンの弱みを相互補完できることが分かった．

4. 早期検知の実現可能性評価

この節では，マルウェア活動の早期検知の実現可能性評価を行う．前節の実験では，正解表に定常的なマルウェア活動が含まれていたため，早期検知の評価が困難だった．本実験では，2019 年 6 月~2020 年 10 月 (17 ヶ月) において観測されたマルウェア活動を対象に，マルウェア活動の感染拡大時期が明確に分かる事象の正解表を新たに手動で作成し，実験を行う．その正解表の内訳の一部を表 3 に示す．本正解表は NICTER で公開しているレポート・ブログ記事を基に，NICTER で観測されたマルウェア活動の中で，活動の原因・特徴が明らか，かつ第 3 者による参考文献が存在するマルウェア活動をピックアップした．その結果，33 個の TCP ポートにおける 12 種類の脅威イベントを収集した．また，NICTER で脅威が観測され始めた初期の時期と 1 時間あたりのユニークホスト数の変化 (規模)，世の中に脅威が明らかになった時期 (公表時期)，脅威の特徴を正確に記録した．本正解表は脅威の種類だけではなく，観測された感染ホスト規模，脅威の持続性・定常性など様々なバリエーションが考慮されている．さらに，本実験では 8 つのセンサ全てではなく，観測規模別に選んで 3 つのセンサ A(/17)，B(/18)，D(/20) のデータを用いた．

表 4 NICTER 観測時期を基準に早期検知，遅れて検知，見逃したポートの数と平均日数結果

Modules	#TPs				#FNs
	早期検知		遅れて検知		
	#Ports	日数	#Ports	日数	
ChangeFinder	15	82.9	10	-5.1	8
Dark-GLASSO	18	89.3	9	-29.5	6
Dark-NMF	28	122.6	3	-79.1	2
Dark-NTD	29	139.9	2	-25.7	2
Dark-TRACER	33	126.4	0	NaN	0

表 5 期間ごとのユニークポート数の平均結果

Modules	ユニークポート数			
	1 日	1 週	1 ヶ月	全期間
Dark-GLASSO	6.21	10.87	17.24	66
Dark-NMF	21.40	85.42	250.53	2042
Dark-NTD	39.12	193.39	565.00	3969
Dark-TRACER	58.49	252.41	718.71	5271

正解表 (33 TCP ポート): *81, 82, 83, 84, 85, 88, 1220, 1588, 3389, 4505, 4506, 4567, 5501, 5984, 6666, 7001, 7002, 8000, 8081, 8089, 8181, 8291, 8728, 8888, 9080, 9090, 9200, 9527, 9530, 9673, 34567, 55555, 60001.*

それでは，第 3.3 節での実験設定と同様に，各モジュールのベストパラメータで処理を行い，単純なルールを適用して調査目的スキャナによるアラートを除外した結果を述べる．ここで *Dark-NMF* は SET1 のパラメータで計算した．表 4 は，NICTER 観測時期を基準に早期検知，遅れて検知，もしくは見逃したポートの数と平均日数結果をモジュール別に示す．この結果から，

モジュール別に見ると見逃しポート (#FNs) や，遅れて検知したポートが多少存在するが，*Dark-TRACER* で統合して見ると 33 個の TCP ポート全てを早期検知できることが分かる．また，NICTER で脅威が観測され始めた時期より平均 126.4 日早く検知でき，世間に脅威が明らかになった時期より平均 153.6 日早く検知できた．

さらに本実験において，各モジュールでどれほどのポートがアラートとして得られたのか調査した．表 5 に期間ごとのユニークポート数の平均結果をモジュール別に示す．17 ヶ月全期間において，*Dark-GLASSO* は 66 個，*Dark-NMF* は 2,042 個，*Dark-NTD* は 3,969 個のユニークなポートに対するアラートを出力し，*Dark-TRACER* で統合してみると 5,271 個だった．これを 1 日，1 週，1 ヶ月ごとに分けてカウントし，その平均を表 5 に述べる．例えば，*Dark-TRACER* では 1 日平均 58.49 個のポートをアラート出力することを意味する．一人の分析者が 1 ポート当たり 15 分の分析時間を要すること仮定すると，約 14.6 時間で日々のオペレーションが可能であることが分かった．1 週間であれば約 63.1 時間，1 ヶ月間は 179.7 時間となる．

5. まとめ

本研究では，ダークネットトラフィックの時空間パターンの同期性をリアルタイムかつ自動的に推定し異常検知を行う 3 つの独立した機械学習手法と，それらを一つのフレームワークに統合したク *Dark-TRACER* を紹介した．*Dark-NTD* のパラメータチューニングを行い，全モジュールの検知精度の定量的な比較評価を行った．*Dark-TRACER* は各モジュールの弱みを相互補完できることが分かり，再現率 100%を達成，かつ実験における全てのマルウェア活動を検知でき，世間に脅威が明らかになった時期より平均 153.6 日早く検知した．さらに，一人の分析者が約 14.6 時間で日々のオペレーションを遂行できることを明らかにした．現在，*Dark-TRACER* は実運用に向けた社会実装を行っている．今後は本論文で深く議論できていない手法の限界やメリット等の考察を行った上で，誤検知を減らし再現率・適合率を共に向上させる仕組みを考えていきたい．

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] Gu, G., Zhang, J. and Lee, W.: BotSniffer: Detecting Botnet Command and Control Channels in Network Traffic, *Proceedings of the Network and Distributed System Security Symposium (NDSS)* (2008).

[2] Bailey, M., Cooke, E., Jahanian, F., Myrick, A. and Sinha, S.: Practical Darknet Measurement, *40th Annual Conference on Information Sciences and Systems*, pp. 1496–1501 (2006).

[3] Friedman, J., Hastie, T. and Tibshirani, R.: Sparse inverse covariance estimation with the graphical lasso, *Biostatistics*, Vol. 9, No. 3, pp. 432–441 (2007).

[4] Han, C., Shimamura, J., Takahashi, T., Inoue, D., Kawakita, M., Takeuchi, J. and Nakao, K.: Real-Time Detection of Malware Activities by Analyzing Darknet Traffic Using Graphical Lasso, *18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications (TrustCom)*, pp. 144–151 (2019).

[5] Han, C., Shimamura, J., Takahashi, T., Inoue, D., Takeuchi, J. and Nakao, K.: Real-Time Detection of Global Cyberthreat Based on Darknet by Estimating Anomalous Synchronization Using Graphical Lasso, *IEICE Transactions on Information and Systems*, Vol. E103.D, No. 10, pp. 2113–2124 (2020).

[6] Lee, D. and Seung, H. S.: Algorithms for Non-Negative Matrix Factorization, *Proceedings of the 13th International Conference on Neural Information Processing Systems (NIPS)*, p. 535–541 (2000).

[7] Han, C., Takeuchi, J., Takahashi, T. and Inoue, D.: Automated Detection of Malware Activities Using Nonnegative Matrix Factorization, *20th IEEE International Conference On Trust, Security And Privacy In Computing And Communications (TrustCom)* (2021).

[8] Kim, Y. and Choi, S.: Nonnegative Tucker Decomposition, *2007 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE (2007).

[9] Kanehara, H., Murakami, Y., Shimamura, J., Takahashi, T., Inoue, D. and Murata, N.: Real-time botnet detection using nonnegative tucker decomposition, *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, ACM (2019).

[10] Takeuchi, J. and Yamanishi, K.: A Unifying Framework for Detecting Outliers and Change Points from Time Series, *IEEE Trans. Knowl. Data Eng.*, Vol. 18, No. 4, pp. 482–492 (2006).

[11] Takahashi, T., Umemura, Y., Han, C., Ban, T., Furumoto, K., Nakamura, O., Yoshioka, K., Takeuchi, J., Murata, N. and Shiraishi, Y.: Designing Comprehensive Cyber Threat Analysis Platform: Can We Orchestrate Analysis Engines?, *2021 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*, IEEE (2021).

[12] Caiafa, C. F. and Cichocki, A.: Generalizing the column–row matrix decomposition to multi-way arrays, *Linear Algebra and its Applications*, Vol. 433, No. 3, pp. 557–573 (2010).

[13] Zhou, G., Cichocki, A., Zhao, Q. and Xie, S.: Efficient Nonnegative Tucker Decompositions: Algorithms and Uniqueness, *IEEE Transactions on Image Processing*, Vol. 24, No. 12, pp. 4990–5003 (2015).

[14] Otsu, N.: A Threshold Selection Method from Gray-Level Histograms, *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 9, No. 1, pp. 62–66 (1979).

[15] Ilascu, I.: New Echobot Botnet Variant Uses Over 50 Exploits to Propagate, https://www.bleepingcomputer.com/news/security/new-echobot-botnet-variant-uses-over-50-exploits-to-propagate/ (2019).

[16] Netlab 360: The Botnet Cluster on the 185.244.25.0/24, https://blog.netlab.360.com/the-botnet-cluster-on-185-244-25-0-24-en/ (2019).

[17] Microsoft: Remote Desktop Services Remote Code Execution Vulnerability, https://msrc.microsoft.com/update-guide/en-US/vulnerability/CVE-2019-0708 (2019).