色付き有限オートマトンの侵入検知システムへの応用

藤本拓海＊1　高橋芳明＊2

An Application of Colored Finite Automata to Intrusion Detection Systems

Takumi FUJIMOTO, Yoshiaki TAKAHASHI

Abstract

In recent years, colored finite automata (CFA), an extension of finite automata, have been proposed, and we are investigating their application to intrusion detection systems (IDS). By assigning “colors” to states, CFAs enable more detailed and multi-layered input identification beyond the binary accept/reject decisions of conventional automata. This research first implements an IDS utilizing these CFA. Subsequently, we aim to evaluate its attack detection accuracy using experimental data and demonstrate the usefulness of CFA-based IDS.

Keywords: *Finite Automata, Regular Expressions, Intrusion Detection System*

1　まえがき

近年、有限オートマトンを拡張した色付き有限オートマトン（Colored Finite Automata：CFA）が提案されている[1)]。CFA を用いることで、従来の有限オートマトンを応用したシステムに対し、より高度な識別能力を付与することが可能となる。その応用例の可能性の一つとして、セキュリティシステムである侵入検知システム（Intrusion Detection System：IDS）が挙げられる。ところで、従来の有限オートマトンを用いた、特にシグネチャベースの IDS では、入力が登録パターンに一致するか否かの二値判定を行う特徴があるため、IDS の識別能力に大きな制約をもたらす。また、IDS において正規表現を決定性有限オートマトン（DFA）に変換してマッチングを行う際、特に複雑なパターンや多数のルールを扱う場合、DFA の状態数が指数関数的に増加する状態爆発問題が発生し、それにより計算量も増加する[2)]。さらに、IDS は検出精度の面でより詳細な情報を必要としているが、既存のシステムではこれを十分に満たせないという課題が指摘されている[3)]。

このような課題を解決するための関連研究として、様々なアプローチが試みられている。例えば、DFA の状態数を削減するための様々な圧縮技術が存在する。Becchi らが提案したハイブリッド有限オートマトン（HFA）[4)]では、非決定性有限オートマトン（NFA）と DFA の長所を組み合わせ、状態爆発が起きるノードは NFA、その他では DFA 変換を使い合わせることでメモリの使用量を削減している。しかし、HFA は巨大な遷移テーブルとそれに伴う検査速度の遅さという課題を抱えている。また、Kumar らが提案した遅延入力 DFA（D²FA）[5)]では、デフォルト遷移を用いて状態数を削減するというアプローチがとられているが、これはスループットの低下を伴う可能性がある。また、複数の正規表現を個別の DFA として構築し並列処理させることでマッチング速度や効率を向上させる DFA のグループ化など[6)]や、Extended Finite Automata[7)]のように、オートマトンに補助メモリを付加することで状態空間爆発を緩和し、効率的なパターンマッチングを目指す研究もその一例である。

このような現状に対し、本研究では、色の概念を導入した CFA を用いることで、既存研究とは異なるアプローチで上記の課題解決に取り組む。CFA は、色という形式的な属性を用いて有限オートマトンの表現力自体を拡張する[1)]。これにより、例えば、ある攻撃パターンの部分一致レベルや攻撃進行度を特定の色で表現することができ、従来の IDS の二値判定では不可能だった多段階的な識別を可能にする。これは、物理的なメモリの使用とは違い、オートマトン自体に意味的な属性を組み込むことで、ルールの

＊1 電子・情報システム工学専攻　＊2 情報工学科

記述性を向上させることにつながる。

そこで本稿では、CFA を実際に IDS に組み込み、攻撃を想定したデータを用いて動作検証及び検知性能の評価を行う。この結果より、CFA を用いた IDS の有用性を示す。

本稿では、まず第 2 章において色付き有限オートマトンを用いた IDS の基盤となる諸定義及び基本概念を整理する。続いて第 3 章では、従来の IDS が抱える課題を明確化し、問題設定を提示する。さらには第 4 章にて、提案手法である CFA を用いた IDS の具体的な実装手法を述べ、最終的に、第 5 章で本研究における評価実験を示し、結論を述べる。

2　諸定義

CFA[1)]は、従来の有限オートマトンの受理状態に色の概念を追加したモデルであり、これにより詳細な入力の分類が可能になる。本章では CFA と、合わせて考慮すべき計算問題に関する定義を与える。

定義 1: [1)]次のように表される 5 項組 $M = (Q, \Sigma, \delta, q_0, \sum_{i=1}^{k} F_i)$を非決定性色付き有限オートマトン（Nondeterministic colored finite automata: NCFA）と呼び、NCFA の 5 項組を以下の通りとする。

1. Q:状態の有限集合
2. Σ:入力文字列の有限集合
3. δ:遷移関数で$Q \times \Sigma$から2^Qへの関数
4. q_0:初期状態の集合
5. $\Sigma_{i=1}^{k} F_i$:色付き受理状態の集合

混色性とは、NCFA から積オートマトンまたは決定性色付き有限オートマトン(DCFA)を生成したとき、1 つの受理状態にその生成元となった NCFA の異なる色の受理状態が 2 つ以上含まれている状況、つまり 1 つの入力に対して 2 つ以上の出力が発生する可能性があることを表す。入力を的確に識別するために、混色状態は望ましくなく、その解決のために非混色性検証問題について考察する必要がある。

定義 2：[1)]NCFA の非混色性検証問題(Unmixedness Verification Problem : UV 問題)は以下のように定義される。

$$\begin{cases} \text{入力：NCFA } M = (Q, \Sigma, \delta, q_0, \sum_{i=1}^{k} F_i), \\ \text{出力：} \bigcup_{i=1}^{k} L_i(M) = \sum_{i=1}^{k} L_i(M)? \end{cases}$$

これは、ある NCFA が与えられたとき、混色しているかどうかを問う問題である。

3　従来の IDS と課題

従来の IDS[2)3)]は、あらかじめ定義されたシグネチャ、ルール、並びに正規表現に基づき、入力を NFA により“受理（攻撃）”または“非受理（正常）”の二値で判定する方式を採用している。この方式は、特定の攻撃パターンを高精度に検出できる一方で、判定結果が二値に限られるため、入力内容の詳細な分類や類似度に基づく柔軟な検知が困難であるという制約を持つ。

例 1：小数点を持つ数値を意味する正規表現[+-]?[0-9]*(¥.[0-9]|[0-9]_¥.)[0-9]*の従来の IDS での判定を例示する。この正規表現を NFA へ変換すると、図 1 のようになる。ここで、dは 0 から 9 の数字を表す。

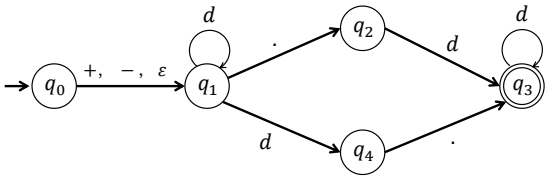

図 1　小数点を持つ数値を受理する NFA

この NFA に対し、「入力なし」「10」「10. 1」の 3 種類の入力を与えると

入力なし：状態q_0に留まり、非受理
「10」　：$q_0 \to q_1 \to q_4$と遷移し、非受理
「10.1」：$q_0 \to q_1 \to q_1 \to q_4 \to q_3 \to q_3$と遷移し、受理

となり、従来の IDS では、この正規表現が表す小数のみ受理するという単一の判定しか行えず、整数と空文字の区別を詳細に行うことはできない。　■

4　提案手法

本章では CFA を用いた IDS という新たな提案手法における基本アイディア及び動作の流れについて記述する。

4. 1　基本アイディア

第 3 章では、従来の NFA と正規表現に基づく IDS の挙動について説明している。これに対して本研究では、IDS に色の概念を加えた NCFA を応用した手法を提案する。この新たな侵入検知システムを CFA-IDS と呼ぶこととする。

例 2：例 1 と同じ正規表現に色付けを施した構文記述[8)]を NCFA に変換した際の検知を例示する。この NCFA では、同じ入力に対して次のような多段階分類が可能となる。

入力なし：q_0に遷移し赤(R)で受理
「10」　：$q_0 \to q_1 \to q_4$と遷移し緑(G)で受理
「10.1」：$q_0 \to q_1 \to q_1 \to q_4 \to q_3 \to q_3$と遷移し、青(B)で受理

つまり、従来であれば非受理として一括処理されていた入力（空文字・整数）を、それぞれ異なる色

で識別し分類できるようになる。以下に、この動作を示す擬似コードをプログラム 1 に、そしてそれを NCFA に変換したものを図 2 に示す。

プログラム 1: 色による識別分類を可能にした擬似コード

```
1 case line =~re"_R[+-]?[0-9]*(¥.[0-9]|[0-9]_G¥.)[0-9]*_B"
2 of R: echo "empty string"
3 of G: echo "an integer number"
4 of B: echo "a fractional number"
5 else: echo "error"
```

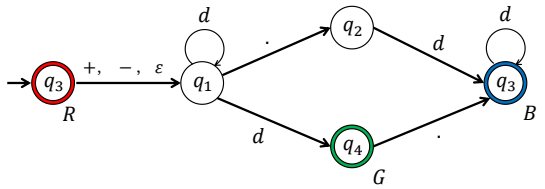

図 2　プログラム 1 で表現される NCFA　■

例 2 に示すように、NCFA を用いることで、複数の判定条件を単一の正規表現へ統合できるという利点がある。この考えを IDS へ応用することで、従来のように類似した条件を複数のルールに分割する必要がなくなり、ルール管理の簡潔化や情報統合が可能になるとともに、プログラム実装時の負荷の減少にもつなげることができる。また、ひとつの入力に対して多面的な判定を同時に行えるため、IDS における詳細な分類と柔軟な検知を実現できると考えられる。

4. 2　CFA-IDS の動作工程

CFA-IDS の実装を行う。実装する CFA-IDS の動作の流れを図 3 に示す。CFA-IDS は前節で記述したように、従来の IDS が持つ二値判定のみでは表現できない情報を状態色として付与することで多段階の識別を可能とする。従来では、より詳細な分類を行うためには、新たなルールの追加または既存ルールの細分化を行う必要がある。一方、CFA-IDS では、単一のルールに対して複数の意味的な状態色を割り当てることにより、同一の有限オートマトン構造を保ったまま、攻撃の種類や文脈に応じた識別が可能となるため、ルール数を増加させることなくルール管理の簡素化、可視化による分析性の向上といった利点が得られる。最初に、実際の通信データなどからデータを抽出し、body 部分など解析対象部分を抽出しデータとして保存する。次に、SAT ソルバー[9]を用いた混色性検証を経て構築されたルール群へ、この抽出されたデータを入力する。各入力は、NCFA の状態遷移を経て処理され、終了時の状態に応じて受理判定及び出力色、例として RED または ORANGE などが決定される。

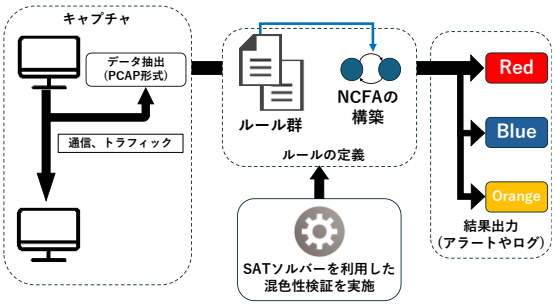

図 3　NCFA を用いた IDS 動作の流れ

最終的に、検査結果は受理の有無、及び「最終色」「マッチしたルール ID」とともに出力された色に応じてアラートなどの処理を行い、その後ログとして出力される。そのログの例を以下に示す。

```
"ts":"2025-10-02T01:52:35",
"flow":"127.0.0.1:8000->127.0.0.1:58510",
"scope":"body",
"rule_id":"CSS-LARGE-MARGIN[YAML-DSL]",
"final_color": "RED"
```

5　評価実験

本章では第 4 章で記述した基本アイディアに則り構築した CFA-IDS に対して評価実験を行う。

5. 1 評価用入力データの用意

本実験では、正規表現的な観点から検知対象となる特徴が適切に含まれているかを確認するため図 4 で示すような手順で評価用入力データの用意及び評価実験を行う。本実験では、CIC-IDS-2017[10] より正常通信データを抽出し、さらに Exploit-DB[11] から本実験で対象とする攻撃に関連する攻撃コードを収集する。これらを対象に、内容の抽出及び Wireshark [12] を用いた通信内容の解析を行い、必要に応じて本実験における検知要素に適合するよう調整した上で、評価用入力データを用意する。検知することを想定した positive データ (pos データ) と、正常通信を模した negative データ (neg データ) を準備し、事前に内容の妥当性を確認したうえで実験に使用する。使用したデータ件数の概要を表 1 に示す。

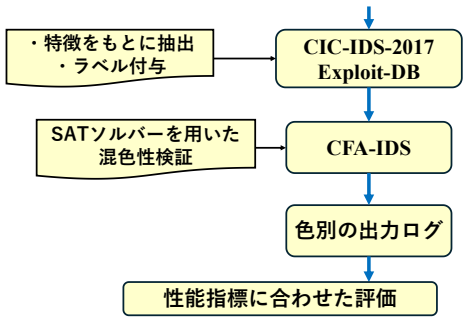

図 4　評価用入力データの用意と評価実験の流れ

表 1　評価実験に用いたデータ総数

Rule	攻撃の種類	pos	neg
1	CSS margin 異常値	99	500
2	動的 JS 実行、リロード悪用	99	
3	外部通信・スクリプト注入	99	
4	DOM/配列長の異常操作	99	
5	DOM 編集 API の悪用	99	
	合計	495	

5. 2 ルール群の定義

本実験では、ルール定義の参考として高い信頼性を有する既存の IDS である Snort[13] を参照する。そのうえで、Snort 内のルールをもとに、そのルールの要素を検知するように拡張する。構築したルールを表 2 に示す。具体的には、Snort ルールをそのまま再利用するのではなく、攻撃を構成する構文要素を分析し、それらを要素レベルで分解してルール設計を行っている。例えば、ルール No.1 に関する攻撃では、異常な桁数の数値指定（例：12 桁以上）を最終的な攻撃条件としつつ、その前段階として現れる 1 桁から 11 桁の数値指定についても攻撃の兆候として扱い、それぞれ異なる色を割り当てることで段階的な検知を可能としている。このルールにおける流れを図 5 に示す。

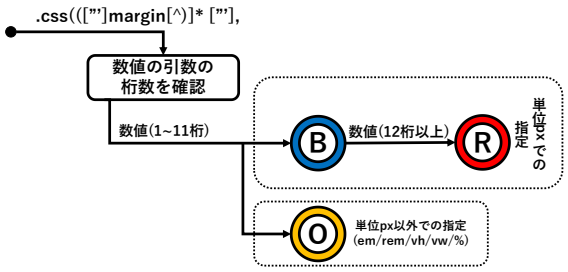

図 5　Rule1 の検知手順

このように、最終的に危険と判定すべき構文に至る前段階の特徴的な構文や値の変化を色として可視化することで、単なる二値判定ではなく、攻撃の進行段階や兆候を表す多段階分類を実現している。また、正規表現中の _R といった色付きの文字は正規表現ではなくこの部分に定義される色を表す。それぞれの色の意味を表 3 に示す。

表 2　CFA-IDS ルール群の概要

Rule	検知対象
1	.*css¥s*¥(¥s*[”’]margin[^)]* [”’]¥s*,¥s*[”’] *([0-9]{1,11}px_B|([0-9]{12,}px_R|[0-9]+(em|rem|vh|vw|%_O) [”’] *¥)
2	.*(onload_M|onload *= *[“’]?location¥.reload_B|onload *= *[“’]?location¥.reload *¥(*<iframe)?_C)
3	.*<script[^>]*(src *=_L|src *=*[”’][^”’]*¥.json[”’] _C |src *=*[”’][^”’]*¥.json[”’][^>]*language=vbs _A)
4	.*((¥.dashstyle¥.array(¥.length_**B**|¥.length= *- [0-9]+_P)|¥.dashstyle *¥.array *¥. *length * *-*¥(? *[1-9]+ *¥)?_C)
5	.*(document *_Y|.*document *¥. *execCommand_G|document *¥. *execCommand*¥(*[”’]superscript[”’] *¥)_T)

表 3　各色による意味

Rule	検知対象
1	R：css()の margin プロパティにて 12 桁以上の数値の指定 B：css()の margin プロパティにて 1~11 桁の数値の指定 O：css()の margin プロパティにて px 以外の単位指定
2	M：onload 属性の読み込み B：再帰的な onload 属性の呼び出し C：再帰的な呼び出しに加え iframe による脆弱性を呼ぶ組み合わせの入力
3	L：<scirpt>内でのファイルの読み込み C：<script>内での外部.json ファイルの読み込み A：.json ファイルを VBSnript として読み込もうとする動き
4	B：dashstyle.array.length によるプロパティの設定 P：dashstyle.array.length への負の値の入力 C：ある程度の空白をゆるした dashstyle.array.length への負の値の入力
5	Y：document 要素の検知 G：execCommand の検知 T：上付き文字設定の脆弱性を検知

5. 3　評価実験

評価実験は以下の指標を用いて評価を行う[10)14)]。TP（True Positive）は正しく pos データを検知した件数、FP（False Positive）は neg データを pos データと認識した件数、FN（False Negative）は pos データを検知できなかった件数、TN（True Negative）は neg データを正しく neg データと判定した件数をそれぞれ意味する。また、その本実験で用いる評価指標を表 4 に、CFA-IDS における各評価指標の値を表 5 にそれぞれ示す。しかし、単なる二値分類だけでは、各攻撃カテゴリがどの程度正しく識別されているかは評価できない。そこで各色について、本来その色で検出されるべき入力が、どの程度見逃されずに正しく検出されているかを検出率（色iとして検出された件数/本来色iとして検出されるべき入力の件数）として算出し評価する。図 6 に、その各色の検出率を示す。その結果、色ごとにばらつきはあるものの、平均して約 80%程度の検出率を達成しており、単純な二値分類に加えて色付き分類を導入しても、識別性能が大きく低下しないことが確認できる。特に、dashstyle 配列や onload の呼び出しなど、構文要素の存在を直接検知するルールにおいては高い値を示し、CFA による構文ベースの識別が安定して機能していることも確認できる。

一方で、数値の桁数の指定などの動的な変化や複雑な記法を含む記述が含まれる入力では、誤分類が発生する傾向が確認される。例えば、<script>タグを用いた外部スクリプトの読み込みの検知ルールでは、属性の記述順序や空白の差異（例：onload="location.reload()"と onload = "location.reload()"）によって、検出状態が変化する例も確認される。これらの結果から、今後は正規表現の境界設計の見直しや、入力正規化などの前処理を導入することで、さらなる精度向上を行うことが必要である。

5. 4　考察

表 5 の結果より、どのルールにおいても、全体を通して高い TPP を実現できている。高い TPP を実現した一例として、Exploit-DB[11)]に掲載されている攻撃コード exploits/windows/remote/46928 に含まれる次の入力例が挙げられる。

```
<script>
  $("#content1").css("margin-left", "50000000px");
</script>
```

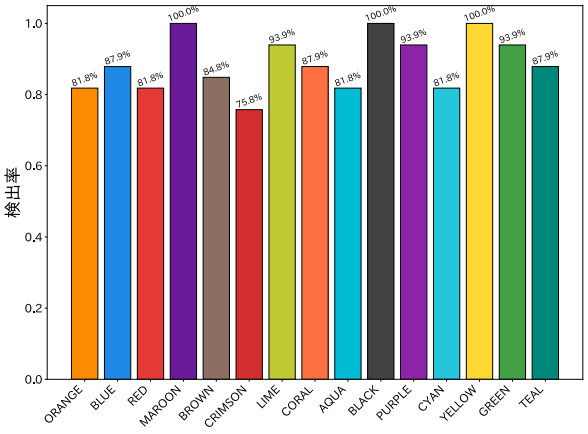

図 6　色別の検出率の割合

表 4　本実験で用いる評価指標

指標名	説明
TPP (True Positive Percentage)	pos データを正しく pos データと判定した割合
TNP (True Negative Percentage)	neg データを正しく neg データと判定した割合
FPP (False Positive Percentage)	neg データを誤って pos データと判定してしまった割合
FNP (False Negative Percentage)	pos データを見逃してしまった割合
Accuracy	全体の中で正しく分類できた割合

表 5　CFA-IDS における各評価指標の値

Rule	TPP	TNP	FPP	FNP	Accuracy	Precision	Recall	F1-score
1	0.8485	1.0000	0.0000	0.1515	0.9849	1.0000	0.8485	0.9180
2	0.9192	0.9821	0.0179	0.0808	0.9759	0.8505	0.9192	0.8835
3	0.9192	0.9989	0.0011	0.0808	0.9910	0.9891	0.9192	0.9529
4	0.9596	0.9911	0.0089	0.0404	0.9879	0.9223	0.9596	0.9406
5	0.9394	0.9955	0.0045	0.0606	0.9899	0.9588	0.9394	0.9490
ALL	0.9192	0.9540	0.0460	0.0808	0.9367	0.9519	0.9192	0.9353

本入力は、CSS の margin-left に対して極端に大きな数値を設定しているが、従来の Snort ルールでは、この挙動が検知条件に完全に一致しない場合、攻撃として検知されないケースが存在する。一方、CFA-IDS では、異常な桁数の数値指定を最終的な検知条件としつつ、その前段階として現れる数値の異常拡大を段階的に識別するようにルール設計を行っている。したがって、本入力のように従来ルールでは検知対象外であったデータについても、異常な構文要素として検知が可能となり、結果として TPP の向上につながったと考えられる。また、本実験において FPP が極端に増加しなかった理由として、使用した negative データの性質が影響していると考えられる。今回の実験では、CIC-IDS-2017[10]において正常通信としてラベル付けされたデータをランダムに抽出して用いているが、これらのデータは通常の業務通信や Web アクセスから構成されているものの、今回の検知対象としている要素を含んでいることが少ない。その結果、CFA によって検知条件を細分化しても検知対象とならない正常データが多く、FPP の増加が顕在化しなかった可能性がある。以上より、本実験結果は、限定された条件下において CFA-IDS の有用性を示すものであるが、評価対象の偏りが結果に影響している可能性も否定できない。今後は、より多様な正常通信データ及び実運用環境に近い混在トラフィックを用いた評価を行い、検出性能及び誤検知率の両面から有用性を検証する必要がある。

6 あとがき

本研究では、色付き有限オートマトンの概念を IDS に応用した CFA-IDS を実装し、検出性能に関する評価実験を行った。その結果、従来の二値判定に基づく手法では検出できなかった入力についても、攻撃の構成要素を段階的に識別することで検知可能であることを確認し、色付き有限オートマトンを IDS に適用する有用性を示すことができた。また、単一のルール内で段階的な要素の検出を色として表現することにより、従来の検知、非検知という判定以上の情報を出力できることを示した。

一方で、本研究で実装した CFA-IDS は、評価実験で想定した入力形式及び攻撃パターンに基づいてルール設計を行っているため、本稿で扱っていない構文構造や記述形式に対しては、十分な検知性能を発揮できない可能性がある。したがって、本研究の結果は限定された条件下における有用性を示すものであり、より一般的な有用性を主張するためには、さらなる検証が必要である。

今後の課題としては、より大規模且つ多様なデータセットを用いた評価実験を行い、実運用環境に近い条件下における検出性能及び誤検知率の検証が挙げられる。また、本研究では既存の NFA に対して色情報を付与することで CFA の動作を再現しているが、今後は色付き正規表現から直接 CFA を構築する専用の処理系や実行エンジンの実装を行い、より広範囲での応用に向けた実装も必要であると考える。

参考文献

1) Y. Takahashi and A. Ito, Finite Automata with Colored Accepting States and Their Unmixedness Problems, IEICE Transactions, Vol. E 105－D (No.3), 491－502, (2022)

2) S. Prithi and S. Sumathi, A survey on recent DFA compression techniques for deep packet inspection in network intrusion detection system, Journal of Electrical Engineering 17.3: 14-14, (2017)

3) K. Kumar and S. Sukumaran, A survey on network intrusion detection system techniques, International Journal of Advanced Technology and Engineering Exploration, Vol 5(47), (2018)

4) M. Becchi, et al., A hybrid finite automaton for practical deep packet inspection, Proceedings of the 2007 ACM CoNEXT conference, (2007)

5) S. Kumar, et al., Curing regular expressions matching algorithms from insomnia, amnesia, and acalculia, Proceedings of the 3rd ACM/IEEE Symposium on Architecture for Networking and Communications Systems, (2007)

6) F. Yu, et al., Fast and memory-efficient regular expression matching for deep packet inspection, Proceedings of the 2006 ACM/IEEE symposium on Architecture for networking and communications systems, (2006)

7) R. Smith, et al., Deflating the big bang: fast and scalable deep packet inspection with extended finite automata, Proceedings of the ACM SIGCOMM 2008 conference on Data communication, (2008)

8) 高橋芳明, 伊藤暁, 色付き有限オートマトンから色付き正規表現への変換アルゴリズム, 第73回電気・情報関連学会中国支部連合大会, R22-18-03, (2022)

9) N. Eén, and N. Sörensson, An extensible SAT-solver, International conference on theory and applications of satisfiability testing. Berlin, Heidelberg: Springer Berlin Heidelberg, (2003).

10) I. Sharafaldin, A. Lashkari, and A. Ghorbani, Toward generating a new intrusion detection dataset and intrusion traffic characterization, 4th International Conference on Information Systems Security and Privacy (ICISSP), 108-116, (2018)

11) Exploit Database,https://www.exploit-db.com/, (2025/9/3)

12) WIRESHARK, https://www.wireshark.org/, (2025/8/11)
13) Snort Project, Snortrules-snapshot-2972, https://github.com/thereisnotime/Snort-Rules, (2024/7/20)
14) N. Khamphakdee, N. Benjamas and S. Saiyod, Improving Intrusion Detection System Based on Snort Rules for Network Probe Attack Detection, 2014 2nd International Conference on Information and Communication Technology IEEE, 69-74, (2014)