Proposal of Web application attack detection method using machine learning

機械学習を用いたWebアプリケーション攻撃検知手法の提案

清水 大貴*　小高 知宏*　黒岩 丈介*　諏訪いずみ*　白井 治彦**

Proposal of Web application attack detection method using machine learning

Daiki SHIMIZU*,　Tomohiro ODAKA*, Jousuke KUROIWA*,
Izumi SUWA* and Haruhiko SHIRAI**

(Received January 27, 2020)

In this paper, we proposed two effective feature extraction methods for detecting intrusion in HTTP request sequences. We compared the classification accuracy of each proposed method using a machine learning.

Attacks on Web applications are difficult to distinguish between normal and abnormal, and mechanical detection is not easy. Therefore, we focused on the fact that attacks on various Web applications are closely related to special symbols that differ from ordinary characters.

As a result of classification using a method characterized by the number of occurrences of special symbols, the accuracy rate was about 95%. Also F-value and AUC are about 94% each.

Key words : Security, HTTP Request, Special Symbols, Machine Learning

1. 緒言

近年, インターネットの普及に伴い,web アプリケーションの利用者が増加している. Web アプリケーションのサービスは多岐に渡り, 日々, 様々な人々が利用している. 例えば,YouTube や Gmail,Skype 等の様々なサービスが挙げられる. それらのサービスは, 一般的な娯楽や利便性向上が目的の利用だけではなく, 企業の業務に利用されることも多くなっている.

Web アプリケーションサービスは様々な利用者がいる反面, サービスの管理者は高いセキュリティ性が要求される. サービスは常に稼働し, 利用可能な状態となることが当たり前なものが多く, 個人情報や社外秘となる情報も大量に扱う. しかし,Web サイトの改ざんや秘密情報の漏洩等の被害も度々報告されている.[1] Web アプリケーションを狙った XSS(Cross-site Scripting) 攻撃や SQL インジェクション攻撃も後を絶たず, 管理者や企業全体にとって, 深刻な問題となっている [2].[3] そのため, データベースに格納されている個人情報や Web アプリケーション自体を脅威から守るために管理を徹底する必要がある.

サービスの管理者自身で施行するセキュリティ対策として,WAF(Web Application Firewall) がある.[4] WAF とは,Web アプリケーションを含む Web サイトと利用者の間で交わされる HTTP 通信 (HTTPS 通信) を検査, 攻撃等の不正な通信を自動的に遮断するソフトウェアである. WAF は, 現在のセキュリティツールの主流として,Web アプリケーションを対象とした外部からの攻撃の対策として, 実際に運用されている.

しかし, 現状運用されている WAF は, 攻撃の検知方法としてシグネチャ検知を用いている場合が多い. シグネチャ検出は, 攻撃を識別するルール (シグネチャ) を予め記述しておき, そのシグネチャに対してパターン照合を行うことによって, 外部からの攻撃を検知している. シグネチャ検知では, ある程度の検知が可能であるが,XSS 攻撃や SQL インジェクション攻撃に用いられる入力は, 正常な入力との区別が難しく, 機械

* 大学院工学研究科 知能システム工学専攻
** 工学部技術部
* Human and Artificial Intelligence Systems Course, Graduate School of Engineering
** Technical Division

的な検知が容易ではない. また, シグネチャのパターン外の攻撃に対して検知は難しいことが現状である.

本研究では, 様々な Web アプリケーション攻撃の検知に対応可能であり, それらに共通する特徴を用いて網羅的に検知できることを目標とする. そこで, 機械学習アルゴリズムを用いるため, 有効な特徴ベクトルの生成手法による検知を試みる.

本論文では,2 章に Web アプリケーションの攻撃と対策を示し,3 章に提案する特徴抽出手法について述べる. また,4 章では, 検知実験について述べ,5 章で実験の結果について述べる. 6 章では結果について考察し,7 章では本研究で提案した手法について, 総括する.

2. Web アプリケーションにおける攻撃と対策

Web アプリケーションとは, ブラウザから利用可能なアプリケーションサービスのことである. クライアントとサーバ間で HTTP 通信を利用してデータの送受信を行っている. 本章では,Web アプリケーションに対する攻撃と、それに対する既存の対策方法について述べる. また, 次章で, 特徴抽出手法を提案するにあたって, 本論文におけるアプローチ手法について述べる.

2.1 Web アプリケーションによる攻撃

本節では,Web アプリケーションに対する攻撃として,XSS(Cross-Site Scripting) 攻撃と SQL インジェクション攻撃を例として挙げて述べる [6].[7]

XSS 攻撃　通常,Web アプリケーションにおいては, 外部から入力した内容を処理し, その出力結果を Web サイトとして表示をすることが一般的である. しかし, 外部から入力したデータが不適切であり,Web アプリケーションが適切に処理をせず, 間違った HTML 生成を行う問題がある. この問題を Cross-site Scripting の脆弱性と呼び, この問題を悪用した攻撃を XSS 攻撃と呼ぶ.

XSS 脆弱性による引き起こされる影響として, 利用者に偽物の Web サイトが表示されることや, ブラウザが保存してある Cookie を不正に取得される等が挙げられる. 開発者が用意したサーバ側のプログラムが, ブラウザから入力された値を, そのまま出力に悪用するため,XSS 攻撃は,HTML コンテツ内にスクリプトを埋め込むことによって, 引き起こされる.

SQL インジェクション攻撃　データベースと連携した Web アプリケーションでは, 外部から入力された情報に基づいてデータベースへアクセスするための SQL 文が生成される. この生成された SQL 文に問題があると, データベースを不正利用される可能性がある.

SQL インジェクション攻撃により引き起こされる影響として, データベース内の秘密情報の閲覧, データベース内の情報改ざん,ID とパスワードを入力しない認証回避等が挙げられる.

SQL インジェクション攻撃は Owasp Top Ten Project[5] では最も有害な脆弱性のひとつとしてリストのトップになっている.

2.2 現状の対策方法

攻撃全体に対する現状の対策手法として, ホワイトリストまたはブラックリストを用いたパターンマッチングによる入力値検査がある. 入力値検査は, 正規表現などの手法を用いて行うが, 想定される入力値に対してホワイトリストを定義することは難しく, 正常なリクエストを異常リクエストとして検出する可能性がある. また, 異常文字列を定義するブラックリストを用いた場合でも, 未知の攻撃に対して検知漏れを引き起こす可能性がある.

個別の対策方法として,XSS 攻撃の対策は, 攻撃者によって, 入力されたスクリプトの終端記号の解釈処理をされないようにすることである. つまり,HTML が生成される際, 終端記号が HTML のタグとして解釈されないように, 実態参照を用いて置換を行う必要がある. また,HTML データを扱わない場合は HTML 生成の出力時にタグを全て排除することなどが挙げられる.

SQL インジェクション攻撃の対策は, データベースにアクセスする SQL 文に外部から入力したデータを扱う場合, 文字列データであれば SQL エスケープを行い, 数値データであるなら適切であるか確認した後に,SQL 文の生成を行うことである. また,SQL 文を予めプリコンパイルを行い, テンプレートとして使用する方法も挙げれる.

しかし, 現状, これら脆弱性の対策を行うことは困難とされている. 理由として,Web アプリケーションプログラムが入力データをどのように扱うか様々であるからである. そのため, 対策が多岐に渡るため, 容易に対策漏れを引き起こす可能性がある.

2.3 各攻撃と特殊記号の関係性

本研究のアプローチ方法として, 各 Web アプリケーションへの攻撃は特殊記号と深く関係している. 攻撃手法のひとつである XSS 攻撃では外部からの入力は下記である.

表 1 各攻撃手法に出現する特殊記号

攻撃名	特殊記号
XSS	< > = · ;
SQL injection	’ = +
LDAP injection	( ) = *
XPATH injection	’ = + /
OS command injection	; / . -
SSI injection	! # - ”
Directory-Traversal	/ · \

```
<script>document.cookie=’sid=ROOT’
</script>
```

入力されている特殊記号として「<」,「>」,「.」,「=」,「’」,「/」が 6 つ出現している.

SQL インジェクションも同様で特殊記号が多く出現する.

```
Select * from user where username= ’admin’
or 1=1 --and password = ’123’
```

入力されている特殊記号として「*」,「’」,「=」,「-」,「スペース」の 4 つが出現している.

他の Web アプリケーションへの攻撃における出現が多い特殊記号を表 1 に示す. 表 1 では,XSS や SQL インジェクション以外の攻撃に出現する特殊記号が多岐に渡ることを示している. つまり, 特殊記号に着目した侵入検知手法を用いることで, 外部からの攻撃に対してある程度の検知が期待できると考えられる.

この着目方法ならば, 既知の攻撃だけでなく未知の攻撃手法に対応も期待できる. 外部からの入力で内部のシステムを動かす場合, 入力に特殊記号が必要不可欠であることがほとんどである. そのため, 今後, 発生する可能性のある未知の攻撃に対しある程度の対処は可能であると考えられる.

3. 提案する侵入検知手法

本章では, 侵入検知の概要と,2.3 節で示した, 攻撃と特殊記号の関係性から提案した本手法について述べる.

3.1 侵入検知の概要

侵入検知の流れを図 1 に示す. はじめに, 正常リクエストと異常リクエストから学習モデルを構築する. 本提案手法では, 攻撃による異常 HTTP リクエストは特殊記号の出現にある特徴があると予想した. それらの特徴を特徴ベクトルとして定量的に表現を行い, 機械学習アルゴリズムを用いて学習モデルを構築する.

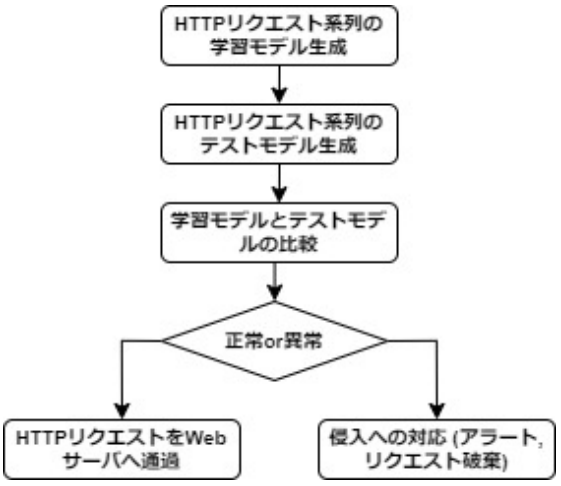

図 1 侵入検知の流れ

特徴ベクトルの作成手法は次節で, 用いる機械学習アルゴリズムに関しては,4 章で述べる.

学習モデルによる十分な学習が終了すると, 次に, テストモデルの構築であり, 同様の手法で構築する. テストモデルは, 学習モデルと異なる点として, ラベル情報を最後の正誤確認に使用されることである.

学習モデルとテストモデルの構築が終了し, 次は, モデルの比較である. テストモデルのひとつが正常リクエストから生成された特徴ベクトルであった場合, 学習モデルの正常リクエストから生成された特徴ベクトルと似た内容になるはずである. 反対に, 検査モデルのひとつが異常リクエストから生成された特徴ベクトルであった場合, 学習モデルは同様に異常リクエストから生成された特徴ベクトルと似た内容になる. つまり, 有効な学習が行われている場合, テストモデルの正常と異常リクエストの分類が可能である.

正常リクエストと異常リクエストを分類したあとの処理として, 正常リクエストであった場合, 当該リクエストを Web サーバへ通過させる. 異常リクエストであった場合, 異常検知として対応を行う.

3.2 モデルの構築

特殊記号は, 表 2 に示している 33 種類であり, これら特殊記号の出現頻度などを特徴とした手法について述べる.

提案した手法は 2 つあり, 特殊記号の出現頻度のみに着目した手法 1 と, 特殊記号に加えて, 英数字の出現頻度にも着目した手法 2 である. また, 提案手法についての説明のため, 特徴抽出対象のリクエストを以下とする.

```
http://local/page.jsp?modo=insertar&precio
=&B1=Pasar+por+caja
```

表 2 特殊記号

sp	!	“	#	$	%	&	’	(	)	*
+	,	-	·	/	:	;	<	=	>	?
@	[	\	]	^	_	‘	{	|	}	~

3.2.1 手法 1

表 3 に手法 1 で抽出する特徴を示す.

抽出する要素として,1 つ目は, 入力項目がいくつあるかを特徴とする (x_1). 特徴抽出対象のリクエストでは, 入力項目は”modo”, ”precio”, ”B1”が対応する. そのため,”3”が最初の特徴量として抽出される.

2 つ目は, 表 2 の特殊記号の出現頻度である (x_2〜x_{32}). ただし, 抽出をする対象は, リクエストの URL 部を除いた部分と, 各パラメタを区別するための記号である”=”, ”&”以外の特殊記号である. つまり, 特徴抽出対象のリクエストでは, 出現してる特殊記号は,”+”が 2 個である. 他の特殊記号は, 出現していないとして,”0”として抽出される. 表 4 に生成された特徴ベクトルを示す. 2 列目が対応している特殊記号を示し,3 行目は, 対象のリクエストから抽出された特徴量を示している.

3.2.2 手法 2

表 5 に手法 2 で抽出する特徴を示す.

手法 2 は, 手法 1 に加えて, 英数字の出現頻度も特徴とする (x_{33}〜x_{95}). 英数字は, 大文字の A〜Z, 小文字の a〜z, 数字 0〜9 の計 62 である. 特殊記号と同様の抽出を行う. 例えば,”a”は,6 個出現,”b”は 0 個で,”B”は 1 個出現している.

表 3 抽出する特徴 (手法 1)

特徴
入力項目数
各特殊記号の出現頻度

表 4 生成された特徴ベクトル (手法 1)

x_1	x_2	x_3	$\cdots$	x_{12}	$\cdots$	x_{32}
	sp	!	$\cdots$	+	$\cdots$	~
3	0	0	$\cdots$	2	$\cdots$	0

表 5 抽出する特徴 (手法 2)

特徴
入力項目数
各特殊記号の出現頻度
英数字の出現頻度

4. 検知実験

本節では, 提案した手法の実験を行う. 実験に伴って, 用いるデータセットと機械学習アルゴリズム, 評価について述べる.

4.1 使用するデータセット

本実験では,ECML/PKDD 2007 Discovery Challenge Dataset[10] を用いる. データセットは XML で定義されており,HTTP リクエストは正常, 攻撃手法とラベル付けで識別される.

データセットは,25000 の HTTP リクエストがあり, 様々な Web アプリケーションのリクエストで構成される. リクエストは 8 つのクラスがあり, 表 6 にクラスの種類を示す. また, リクエストはすべて GET メソッドとして統一して実験を行う. データセットから,GET メソッドのみを抽出しリクエストを 2 つに分割する. 正常リクエスト 7436 と異常リクエスト 3752 を 1 つのセットとして, それぞれ dataset1 と dataset2 とする.

4.2 機械学習アルゴリズム

本実験で用いる機械学習アルゴリズムは, サポートベクタマシン (SVM) とランダムフォレストの 2 つである.

サポートベクタマシンは, 線形可能なデータに対して,「マージン最大化」という概念のもと分類境界を決めることによって, 一般的に高い精度をもつといわ

表 6 構成されているクラス

Class
Normal request
Cross-Site Scripting
SQL Injection
LDAP Injection
XPATH Injection
Directory-Traversal
OScommand Injection
SSI attacks

れている.[8] 線形分類が難しい場合, カーネル法といわれる高次元への写像を行うことによって分類を可能としている.

ランダムフォレストは, アンサンブル学習の一つであり,Leo Breiman によって 2001 年に提案された.[9] アンサンブル学習は複数の弱分類器によって強分類器のような識別を行う学習方法のことであり, ランダムフォレストは複数の決定木を用いて識別を行う機械学習手法である.

上記のアルゴリズムを用いて分類を行う. また,SVM は, 線形 SVM とカーネル法を用いたカーネル SVM の 2 つを用いる.

4.3 評価

提案手法を機械学習アルゴリズムによって生成した分類器の性能は混同行列 (confusion matrix) によって表として要約することが可能である. 混同行列を表 7 に示す. 各行はテスト集合に記録された実際のクラスを示し, 各列は, 分類器によって予測されたクラスを示す. 混同行列から様々な評価指標を導くことができる. 提案した手法を評価するための評価指標を表 8 に示す.

5. 分類結果

本章では,4.1 節で述べたデータセットを用いて, 検知実験を行い, その結果を示す. 表 9 は, 線形 SVM による分類結果, 表 10 は, カーネル SVM による分類結果, 表 11 はランダムフォレストによる分類結果である.

設定したハイパパラメータは, 線形 SVM では, 手法 1 と手法 2 ともに $c = 0.5$ である. カーネル SVM では, 手法 1 では,$c = 5, \gamma = 0.1$ であり, 手法 2 では,$c = 50, \gamma = 0.01$ としている. ランダムフォレストでは, 手法 1 と手法 2 ともに, 決定木の数は 40, 木の深さの最大値は 10 とした.

線形 SVM による分類結果の傾向として, 手法 1 を用いた検知と手法 2 を用いた検知では, ほとんど同じ精度であった. それぞれの平均正解率も 93%程度と同程度であり,F 値や AUC も手法 1 と手法 2 では, ほとんど差異はなかった.

カーネル SVM による分類結果の傾向として, 線形 SVM では精度はほとんど同じであるのに対し, 異なる結果となった. 手法 1 を用いた検知では, 平均正解率は,94.8%程度であり, 手法 2 を用いた検知では,93.3%程度と手法 1 を用いた検知のほうが精度が向上して

表 7 混同行列

	予測 $\oplus$	予測 $\ominus$
実際 $\oplus$	TP : True positive	FN : False negative
実際 $\ominus$	FP : False positive	TN : True negative

表 8 評価指標

尺度	定義・意味
正解率	$acc = \frac{TP+TN}{TP+FP+TN+FN}$
誤り率	$err = 1 - acc$
再現率	$recall = \frac{TP}{TP+FP}$
真陰性率	$tnr = \frac{TN}{FP+TN}$
偽陽性率	$fpr = \frac{FP}{FP+TN}$
偽陰性率	$fnr = \frac{FN}{TP+FN}$
適合率	$precision = \frac{TP}{TP+FP}$
F 値	$Fm = \frac{2recsll \cdot precision}{recsll+precision}$
AUC	ROC 曲線の積分値

表 9 線形 SVM による分類結果

	手法 1		手法 2	
	dataset1	dataset2	dataset1	dataset2
正解率	0.931	0.935	0.933	0.934
誤り率	0.069	0.065	0.067	0.066
再現率	0.812	0.815	0.810	0.810
真陰性率	0.992	0.997	0.996	0.997
偽陽性率	0.008	0.003	0.004	0.003
偽陰性率	0.188	0.185	0.190	0.190
適合率	0.981	0.994	0.990	0.994
F 値	0.889	0.895	0.891	0.893
AUC	0.902	0.906	0.903	0.904

表 10 カーネル SVM による分類結果

	手法 1		手法 2	
	dataset1	dataset2	dataset1	dataset2
正解率	0.946	0.949	0.928	0.937
誤り率	0.054	0.051	0.072	0.063
再現率	0.861	0.870	0.853	0.853
真陰性率	0.991	0.990	0.972	0.980
偽陽性率	0.009	0.010	0.028	0.820
偽陰性率	0.139	0.130	0.157	0.147
適合率	0.979	0.977	0.938	0.956
F 値	0.916	0.921	0.888	0.901
AUC	0.926	0.930	0.907	0.916

表 11 ランダムフォレストによる分類結果

	手法 1		手法 2	
	dataset1	dataset2	dataset1	dataset2
正解率	0.934	0.936	0.944	0.945
誤り率	0.066	0.064	0.056	0.055
再現率	0.817	0.820	0.844	0.847
真陰性率	0.995	0.995	0.996	0.999
偽陽性率	0.005	0.005	0.004	0.001
偽陰性率	0.183	0.182	0.156	0.153
適合率	0.987	0.992	0.991	0.997
F 値	0.894	0.898	0.911	0.916
AUC	0.906	0.908	0.920	0.923

いる. また,F 値と AUC も手法 1 のほうが 2%程度向上してる.

ランダムフォレストによる分類結果の傾向として, 手法 1 を用いた検知より, 手法 2 を用いた検知のほうが精度が向上している結果となった. 手法 1 を用いた検知では, 平均正解率は,93.5%程度であり, 手法 2 を用いた検知では,94.5%程度となっている. F 値と AUC も, どちらも手法 1 より手法 2 を用いた検知のほうが,2%程度の差ができた. つまり, ランダムフォレストを用いた場合, カーネル SVM による分類結果とは, 反対に手法 2 のほうが有効である結果となった.

6. 考察

本章では, 特殊記号に着目した特徴抽出である, 手法 1 と手法 2 を用いた実験の考察を示す.

手法 1 と手法 2 は, ほとんど同様であることが結果からわかる. 少しの差であるが, 手法 1 が手法 2 より,1%程度精度が高くなっている. 全体として,90%以上であることから, 手法 1, 手法 2 とも有効であるといえる. 実際に検出器として運用するという観点から, 偽陰性率は, 手法 1 では 13%程度であり, 手法 2 では 15%程度である. そのため, 異常リクエストを正常リクエストと誤検知を少なくする必要性があるため, 手法 1 を実際に検出器として選択が望ましいと考えられる. しかし,10%以上ある誤検知率を保持した状態で, 運用することは難しいと考えられるため, 偽陰性率を低くする改良が必要である.

本提案手法の結果において, 正解率,F 値,AUC が 90%を超えており, 正常リクエストと異常リクエストを分類するための特徴として, 特殊記号に着目したことは, 有用であるといえる.

異常リクエストを正常リクエストと誤検知してしまった攻撃は, 主に XPATH インジェクション攻撃とディレクトリトラバーサルが多い傾向であった. XPATH インジェクション攻撃は,SQL インジェクション攻撃と同様に, 終端記号を混入させて, その後の文字列構造を変化することが特徴的な攻撃である. そのため,SQL インジェクション攻撃の誤検知が少ないことや, 終端記号などの特徴的な特殊記号が出現する攻撃であるにも関わらず誤検知が多い結果となった.

ディレクトリトラバーサルは, ディレクトリの相対パスを文字列に含ませることで引き起こされる攻撃であり, パスの指定が特徴的である. この特徴は,XPATH インジェクション攻撃にも当てはまり,XPATH の指定を行うため, パス指定でよく利用される「\」が多く出現する. つまり,XPATH インジェクション攻撃とディレクトリトラバーサルの 2 つの共通点から, 特殊記号「\」を含まれる攻撃においては, 分類が完全にできていないといえる. また, 誤検知したディレクトリトラバーサルが含まれているリクエストに関して, 手法 1 よりも手法 2 のほうが多く含まれている傾向であった. 理由として, 誤検知したディレクトリトラバーサルが含まれているリクエストは, 通常の特殊記号ではない文字列が多く出現しているものがいくつかあった. 特殊記号以外の要素である英数字の出現頻度を加えた特徴抽出を行った手法 2 では, 文字の並び順が考慮されていないことや, 正常と異常に関係のない文字列を特徴と捉えてしまっている. そのため, ディレクトリトラバーサルが分類できなかったリクエストが増えて, 手法 1 より手法 2 は少しであるが精度が低くなってしまったと考えられる.

7. 結言

本章では,Web アプリケーションに対する攻撃を検知するために提案した特徴抽出手法について総括し, 今後の課題点について述べる.

本研究では, 様々な Web アプリケーション攻撃手法の検知に対応可能であり, それらに共通する特徴を用いて網羅的に検知できることを目的とした, 手法の提案及び, その実装を行った.

本研究で提案した手法は, 異常リクエストに多く出現する特殊記号に着目したアプローチに沿って,2 つの手法を提案した. 提案手法の有用性を確認するために各手法に対して, 公開されているデータセットである ECML/PKDD 2007 Discovery Challenge Dataset を用いて実験を行った. 実験の結果, 提案した手法 1 が侵入検知として実装できる可能性があることが実証

された.

今後の課題として, 以下の点が挙げられる.

はじめに, 提案した手法の改善が必要である. 本提案手法は, 正常リクエストと異常リクエストのある程度の分類が可能であったが, 完全な分離ができたわけではない. 仮に, 実際の検出器として WAF に導入した場合, 分類が 99%であったとしても, 残り 1%の誤検知によって, 深刻な被害に繋がる可能性もある. そのため, 最終目標として,100%の分類を目指す必要がある.

本手法では, 特殊記号に着目し, ある程度の結果であったことから, 特殊記号への着目は有効であったといえる. 各入力内の出現頻度について, 特殊記号以外の着目する部分を加えることによって, より良い精度が期待できる.

加えて, 特徴抽出後のデータに対して, 加工を行うなどもある. 本手法で得た特徴ベクトルに対して, そのまま加工を加えずに, 機械学習アルゴリズムを用いて学習モデル生成を行っている. 特徴ベクトルに加工を加えることによって, 異なる結果となることがある. よい精度になるか不明であるが, 本手法で得た特徴ベクトルは,"0"である要素が多くあり, 冗長的となっている. この特徴ベクトルに対して, 標準化や均一化といった処理を加えることによって, 精度が上がる可能性がある.

次に, 分類を行う際に用いている機械学習アルゴリズムに関してである. 本実験では, 機械学習アルゴリズムとして,SVM とランダムフォレストを用いたが, これら以外に様々なアルゴリズムが存在する. 他の機械学習アルゴリズムを用いて実験を行うことによって, 劇的な精度の改善とはならないが, 多少の改善が期待される.

上記の課題となる事項を取り入れることにより, 高い精度で正常リクエストと異常リクエストの分類が可能になると期待される.

参考文献

[1] 情報処理推進機構 : ソフトウェア等の脆弱性関連情報に関する届出状況, https://www.ipa.go.jp/files/000073462.pdf

[2] Anley C : Advanced SQL injection in SQL Server applications., http://www.nextgenss.com/papers/advancedsqlinjection.pdf (2002).

[3] OWASP Cross-site Scripting (XSS). (2011).

[4] 情報処理推進機構 : 安全なウェブサイトの作り方改訂第 7 版第 3 刷., https://www.ipa.go.jp/files/000017316.pdf (2016).

[5] OWASP TOP Ten [EB/OL] (2017).

[6] 伊波靖, and 高良富夫 : サポートベクタマシンを用いた WAF への異常検知機能の実装と評価, 情報処理学会論文誌, 7-1, 1-13 (2014).

[7] 園田道夫, and 松田健, : 攻撃特徴記号に基づく WAF 開発, 情報処理学会論文誌, 56-9, 1826-1833 (2015).

[8] 栗田多喜夫 : サポートベクターマシン入門, 産業技術総合研究所 脳神経情報研究部門 (2002).

[9] Breiman, Leo : Statistics Department University of California Berkeley, Machine Learning, Springer, 45, 5-32 (2001).

[10] Analyzing web traffic:Ecml/pkdd 2007 discovery challenge., http://www.lirmm.fr/pkdd2007-challenge/