4ZA-01

機械学習による Web アプリ脆弱性の検出技術に関する研究‡

陳 含悦† 大久保 隆夫†

Hanyue Chen Takao Okubo

1 はじめに

近年，Web サイトにて商品を販売するオンラインショッピングの EC サイトを始め，銀行業務を取り扱うオンラインバンキングや証券オンライントレード，ソーシャルネットワーキングサービスといった Web アプリケーションは世の中に広く浸透し，現代人の生活になくてはならない存在となっている．

Web 技術の進化とともに，最近では一般企業や行政機関の Web サイトに対して，個人情報，金銭目的を狙った悪質な攻撃による被害が多く見られ，社会的問題にもなっている．

IPA 独立行政法人情報処理推進機構の定期レポート「脆弱性に関する届出状況」によれば，日本国内では 2004 年 7 月から 2018 年 9 月にかけて，Web サイト脆弱性の届出の累計件数が 9,829 件も報告されている．

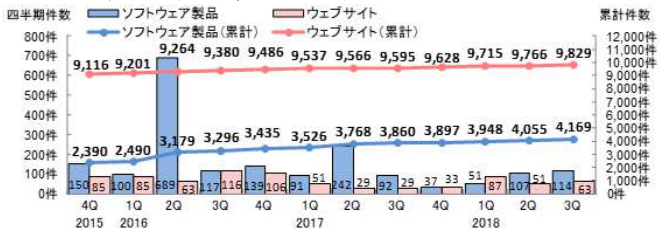

図 1 脆弱性の届出件数［1］

Web システムが侵入される時の被害状況の早期特定や緊急対応が，各開発及び保守現場の技術者がセキュリティ知識持っているか否かに大きく依存される．

2 Web アプリの脆弱性

Web アプリの脆弱性に対する攻撃手法として有名なものには SQL インジェクション，クロスサイトスクリプティング（XSS），クロスサイトリクエストフォージェリ（CSRF）といったものがある．

海外の OWASP(Open Web Application Security Project)組織より「OWASP Top 10」という良く使われるクリティカルな脆弱性の Top10 のセキュリティリスクが公開されている．

OWASP Top 10 - 2013		OWASP Top 10 - 2017
A1 – インジェクション	→	A1:2017-インジェクション
A2 – 認証の不備とセッション管理	→	A2:2017-認証の不備
A3 – クロスサイトスクリプティング (XSS)	↘	A3:2017-機微な情報の露出
A4 – 安全でないオブジェクトへの直接参照 [A7とマージ]	U	A4:2017-XML 外部エンティティ参照 (XXE)[NEW]
A5 – 不適切なセキュリティ設定	↘	A5:2017-アクセス制御の不備 [マージ]
A6 – 機微な情報の露出	↗	A6:2017-不適切なセキュリティ設定
A7 – 機能レベルのアクセス制御の不足 [A4とマージ]	U	A7:2017-クロスサイトスクリプティング (XSS)
A8 – クロスサイトリクエストフォージェリ (CSRF)	☒	A8:2017-安全でないデシリアライゼーション [NEW,コミュニティ]
A9 – 既知の脆弱性のあるコンポーネントの使用	→	A9:2017-既知の脆弱性のあるコンポーネントの使用

図 2 OWASP Top 10［2］

最もリスクが高いのは，SQL インジェクションや、OS コマンドインジェクションなどのインジェクション攻撃になっていることが分かる．

これらの Web サイトの脆弱性が悪用され，改ざん，サービス不正利用，秘密情報漏えいといった被害に遭うと，対処するための費用や Web サイトの閉鎖による機会損失が生じる．

より深刻なケースでは，攻撃者により Web サーバーにマルウェアが仕込まれ，サイト利用者まで感染する場合や，他者への攻撃の踏み台として利用される場合など，被害を拡大させた加害者ともなりかねない．

このような状況に対し，現場の技術者やセキュリティ専門家に依存せず，柔軟な脆弱性対策できる方法が必要である．

そこで，本研究では Web アプリの脆弱性である SQL インジェクションを効率よく検出するために，機械学習のニューラルネットワーク技術を活用した手法を提案し検証した．

2.1 SQL インジェクション

データベースには重要な個人情報を記録するように設計されることが多いため，SQL インジェクション攻撃が成功すれば、データベース上であらゆる操作を行われことが可能である．

もし，データベース層に脆弱性がある場合，データベースに記録されている機密情報の閲覧や盗難，改ざん，そして消去されてしまう被害につながる．

‡ Web Application Vulnerability Detection by Machine Learning Technology

† 情報セキュリティ大学院大学 情報セキュリティ研究科
Graduate School of Information Security INSTITUTE of INFORMATION SECURITY

具体例として，認証処理で以下のような SQL 文が発行される場合

```
SELECT * FROM table WHERE id='ID01' and pw='PW01';
```

「PW01」に該当する入力値を「' or '1'='1」にすると，

```
SELECT * FROM table WHERE id='ID01' and pw='' or '1'='1';
```

の SQL 文が実行され，常に真となるため，認証処理を回避することができてしまう．

3 機械学習

本章では，本研究で利用したニューラルネットワークモデル「LSTM」「Seq2Seq」の概要について説明する．

3.1 LSTM

SQL 文を理解させるため，リカレントニューラルネットワーク（RNN）の一種である LSTM（Long-Short Term Memory）モデルを構築した．

LSTM は長期的・短期的なデータ間の依存関係がある時系列データを学習することが可能であり，音声認識，言語モデリング，翻訳，文書作成などの分野で幅広く利用されている．

そして，忘却ゲートを設けることによって，一般的なリカレントニューラルネットワークが持つ長期依存性の問題を回避するように設計されている．

3.2 Seq2Seq

LSTM モデルでは，固定長の入力時系列データから出力時系列データへと計算することで簡単に言語処理が可能だが，時系列データの系列長が未知で，入出力が可変長の時系列データといった非単調な関係を伴う問題に対しては，対処が困難である．

そこで，Seq2Seq モデルは二つの異なる RNN モデルを用いて，入力側のエンコーダー（Encoder）と出力側のデコーダー（Decoder）によって，可変長の時系列データのペアを学習させることができる．

4 実験内容

本章では，Seq2Seq モデルを用いた Web アプリの脆弱性検出手法を説明する．

4.1 概要

一般的に，攻撃者は SQL インジェクションで攻撃する前に，Web アプリ側に実行される SQL 文を観測または推測する必要がある．

前述の例で挙げた認証処理でいうと，特に「WHERE」句以降の「id=' ID01'」や「pw='PW01'」に注目し，この場合は「id=」の後ろに「'」になっていることから，「' OR '1'='1」を入力すれば，成功する可能性があると判断する．

4.2 環境

OS	Windows 10 Home 64bit
CPU	Intel Core i9-7900X
GPU	NVIDIA GeForce GTX 1080Ti SLI 11GB
Memory	64GB
Framework	Python3.5, TensorFlow, Keras
Server	OWASP bwa,Mysql-5.1.41

表 1 実験環境

4.3 実験詳細

図 3 に示した構成の Seq2Seq モデルを用いて，人間と同じような判断は機械ができないか検証してみた．

```
Layer (type)                 Output Shape              Param #   Connected to
==================================================================================================
input_1 (InputLayer)         (None, None, 33)          0
--------------------------------------------------------------------------------------------------
input_2 (InputLayer)         (None, None, 12)          0
--------------------------------------------------------------------------------------------------
lstm_1 (LSTM)                [(None, 64), (None,  25088     input_1[0][0]
--------------------------------------------------------------------------------------------------
lstm_2 (LSTM)                [(None, None, 64), ( 19712     input_2[0][0]
                                                            lstm_1[0][1]
                                                            lstm_1[0][2]
--------------------------------------------------------------------------------------------------
dense_1 (Dense)              (None, None, 12)          780       lstm_2[0][0]
--------------------------------------------------------------------------------------------------
```

図 3 構築した Seq2Seq モデル

具体に，入力として，正常 SQL 文をエンコーダ側に渡し，デコーダー側の出力となる攻撃コードを写像するというシーケンスのペア

- Encoder：

```
SELECT * FROM table WHERE id='ID01'
SELECT * FROM table WHERE id=('ID01')
… 省略 …
```

- Decoder：

```
' OR '1'='1
') OR ('1'='1
… 省略 …
```

を学習させることによって，SQL インジェクションの攻撃パターンを自動生成する．

5 結果

検証用サーバー「OWASP bwa」の「bWAPP」を用いて， Mysql の Query ログに着目し，リアルタイムで監視する．そして，Web アプリの操作によって，Query ログに吐き出された SQL 文を学習済みの Seq2Seq モデルに入力し，攻撃文を予測させた．

その結果，まだ厳密な数値ではないが，これまでの実験対象の Web 入力画面に対して，およそ 8 割以上は正確に SQL インジェクションの攻撃コード予測されたことが判明した．

今後，さらに学習データと検証サイトを増やし，予測の精度を測る予定である．

参考文献

[1] IPA "ソフトウェア等の脆弱性関連情報に関する届出状況［2018 年第 3 四半期（7 月～9 月）］"
https://www.ipa.go.jp/files/000069580.pdf（最終閲覧日：2019 年 1 月 6 日）

[2] OWASP "OWASP Top 10 - 2017"
https://www.owasp.org/images/2/23/OWASP_Top_10-2017%28ja%29.pdf（最終閲覧日：2019 年 1 月 6 日）