2W-09

JavaScript キーロガーの検知手法の提案と実装

平 俊介† 高須 航‡ 武居 直樹‡ 山田 智隆‡ 石川 貴之‡
細井 理央‡ 安田 昂樹† 高橋 和司† 齋藤 孝道†
明治大学† 明治大学大学院‡

1.はじめに

JavaScript を用いることで利用者のキー入力を秘密裏に採取する，所謂 JavaScript キーロガーがある．攻撃者により JavaScript キーロガーが Web ページに挿入された場合，認証に用いる ID，パスワードやクレジットカード番号等の機密情報が採取されてしまう．近年では，Web サイトの改ざんに伴うスクリプト挿入攻撃による被害は相次いでおり，Web サイト運営者側のみならず，利用者側でも対策を講じる必要がある．そこで本論文では，利用者側での対策として，アクセスした Web ページを解析することで，JavaScript キーロガーの検知を行うブラウザの拡張機能の提案及び実装を行う．

2.JavaScript キーロガー

2.1. 概要

JavaScript キーロガーは，アプリケーション型のキーロガーと違い，ターゲット PC へのインストールを必要とせず，JavaScript キーロガーが挿入されている Web ページに訪れるだけで機能する．JavaScript キーロガーが行う，利用者のキー入力を窃取するための動作は，キー入力に反応するイベント（keydown，keypress，および keyup）を用いて利用者のキー入力を取得し，その取得したキー入力内容を，数秒毎に，XHR や画像等のリソースダウンロードといった，ページ遷移を伴わないリクエストに含んで，秘密裏に任意のサーバへ送信するという特徴がある．JavaScript キーロガーの事例として，AlienVault 社[1]により報告された Scanbox という偵察ツールが挙げられる．Scanbox が仕込まれた Web ページでは，タブが閉じられるまで 5 秒ごとに，その Web ページ内でのキー入力を攻撃者サーバへ送信しており，企業などの Web サイトに挿入されていたことがわかっている．

A Proposal and Implementation method for Detecting JavaScript Keylogger
†Shunsuke TAIRA ‡ Ko TAKASU ‡ Tomotaka YAMADA ‡Naoki TAKEI ‡Yuta NISHIKURA ‡Takayuki ISHIKAWA ‡Rio HOSOI †Koki YASUDA † Kazushi TAKAHASHI †Takamichi SAITO
† Meiji University ‡ Graduate School of Meiji University

2.2. 既存の対策方法

JavaScript キーロガーの既存の対策方法として，ソフトウエアキーボードの利用がある．ソフトウエアキーボードは，画面上に表示されたキーボードをマウスやタッチペンなどで指定し，文字入力を行うものである．キー入力に反応する JavaScript のイベントが発火しないので，キー入力が窃取されることを防ぐことができる．しかし，マウスやタッチペンでの入力は，キーボード入力に比べ煩雑であると言えよう．その他の対策として，NoScriptSuiteLite[2]等のブラウザ拡張機能により JavaScript を無効化する方法もある．しかし，Alexa[3]による Web サイトアクセス数ランキング TOP1,000,00 のうち，JavaScript の利用率は 92.0%[4]と非常に高いため，現実的な対策とは言えない．

3.提案手法

本提案手法では，検査対象ページに対し以下の 3 つの手順を(A)→(B)→(C)の順で行い，JavaScript キーロガーが挿入されているか判断する．また， Chrome は，世界でのブラウザ普及率が 47.26%（2015 年 11 月時点）[5]と最も高いことから，本提案手法を Chrome 拡張機能により実装する．

（A） form タグの検知

キーロガーは ID やパスワードを盗むことを目的とするため，ログインページに挿入される可能性が高い．ログインページでは一般的に，利用者の ID 及びパスワードの入力欄として，form タグが使用される．form タグの検知には，与えられたタグ名を持つ要素のリストを返す getElementsByTagName プロパティを使用し，Web ページ上に form タグが存在するか否かを確認することで検知が可能である．

（B） キー入力に反応するイベントの検知

一般に，2 種類のイベント追加方法がある．キー入力に反応するイベントの検知方法を図 1 にに示す．図 1 の(a)では，ノードオブジェクトの onkeydown イベントハンドラにイベント発火時に行われる関数を代入することで，keydown イベントの追加を行っている．よって，そのノードオブジェクトの onkeydown イベントハンドラ

が，関数を持っているかどうかを調べることで検知が可能である．なお，keypress，keyup イベントに関しても同様に検知可能である．それに対し図 1 の(b)の方法では，検知に使用できるプロパティや検知をするための API が用意されていないので，イベントとして検知できない．よって，Web ページ上の全ての JavaScript コードを取得し，文字列を検知する方法をとる．

```
(a)document.onkeydown=function(e){…};

(b)document.addEventListener("keydown",f
  unction(e){…});
```

図 1. イベント追加方法

（C） 外部ドメインへのページ遷移を伴わないリクエストの検知

JavaScript キーロガーは前述の通り，XHR や画像等のリソースダウンロードといった，ページ遷移を伴わないリクエストにキーログを含んで外部ドメインへの送信を行う．ページ遷移を伴わないリクエストの検知方法として，Chrome 拡張機能で用意されている，webRequestAPI[6]を使用する．図 2にページ遷移を伴わないリクエストの検知方法を示す．

```
chrome.webRequest.onBeforeRequest.addL
istener(
    function(details){/*省略*/},
    types:["sub_frame","stylesheet","s
cript","image","object","xmlhttpreques
t","other"]
);
```

図 2. ページ遷移を伴わないリクエスト検知

図 2 の chrome.webRequest.onBeforeReqest は，webRequestAPI に属するイベントで，リクエストが送信される直前に発火する．第 1 引数に発火後の処理，第 2 引数に発火対象とするリクエストの種類を記述する．図 2 では，内部ドメインへのリクエストである main_frame を除き，types で指定可能な全てのリクエストの種類を検知対象とした．次に，リクエストが外部ドメインへの送信であることを確認する方法を説明する．図 2 の details は，リクエスト先の URL を持っており，これを Web ページのドメインと比較する．一致しない場合，リクエスト先のドメインは外部ドメインであると判断することができる．

以上，3 つ全てを検知した場合，Web ページ上にキーロガーが挿入されていると判断する．

4. 評価

本論文では検知率についての評価を行った．検知率の評価方法として，Alexa による世界の Web サイトアクセス数ランキング TOP200（2015 年 12 月時点）の Web サイトに対し，本提案手法を適用し，入力できる全てのフォームに対して入力を行うことで，JavaScript キーロガーが検知されるかどうか調査した．評価対象とする Web ページはトップページであるが，ログインページを持つ Web サイトについては，ログインページも同様の評価を行った．結果，18 のトップページと 1 つのログインページを含む，合計 19 の Web サイトで検知された．これら 19 の Web サイトについて，実際にリクエストのクエリ内容を確認したところ，13 の Web サイトでは，利用者が入力した文字列を含んだリクエストを送信していた．1 つのログインページを含む残り 6 つの Web サイトでは，リクエスト内に利用者が入力した文字列を確認できなかったが，それら文字列を暗号化してリクエストに含み送信している可能性がある．これら Web サイトは違法なものではなく，企業がマーケティングに活用する等の目的で，顧客情報を収集する内部向けのキーロガーを動作させていることが推測される．

5. 課題

本提案手法の課題として，JavaScript コードが難読化されていた場合，3 節で示したイベント検知方法では，addEventListener で追加されたイベントの検知ができず，キーロガーを検知することができなくなることが挙げられる．

6. まとめ

本論文では，Web ページの form タグの検知，キー入力に反応するイベントの検知，および外部ドメインへのページ遷移を伴わないリクエストの検知の 3 つによって，JavaScript キーロガーを検知する手法を提案した．

7. 参考文献

[1]AlienVault, https://www.alienvault.com/
[2]NoScriptSuiteLite, https://chrome.google.com/webstore/detail/noscript-suite-lite/ahnanjpbkghcdgmlchbcfoiefnifjeni?hl=ja
[3]Alexa, http://www.alexa.com/
[4]w3techs,http://w3techs.com
[5]StatCounter, http://gs.statcounter.com/ats.asp
[6]chrome.webrequest, https://developer.chrome.com/extensions/webRequest