Communications of the CCISA
Vol. 25 No. 3 Aug. 2019

基於人工智慧的自動化網頁安全測試

林聖翔[1]、卓信宏[2*]
資訊工程系, 國立宜蘭大學
[1]p8989p72@gmail.com、 [2]hhcho@niu.edu.tw

摘要

以網頁安全為例，資安工程師在做測試時，通常會準備一個大量的攻擊語法列表，一些著名的免費漏洞掃描工具除了使用現成的攻擊語法列表外，有些是基於已知的攻擊語法格式來生成攻擊語法，這樣的方式能夠節省時間與人力成本，但這樣的結果僅能測試出已經被發現的問題，甚至有時候成功率也不高。

為了增加安全測試的效率，我們靠增加攻擊語法的可變性，希望能挖掘出更多漏洞，因此我們提出了一套基於人工智慧的自動化網頁安全測試系統，藉由人工智慧強大的未知搜索能力生成偽資料的特性，來學習和產生攻擊語法，讓安全測試人員在測試上有新的選擇。

關鍵字: 網頁安全、人工智慧、網頁安全測試系統、網頁安全測試

Regular Paper

Communications of the CCISA
Vol. 25 No. 3 Aug. 2019

Automatic Web Security Testing based on Artificial Intelligence

Sheng-Xiang Lin[1], Hsin-Hung Cho[2*]
[12]Department of Computer Science and Information Engineering, National Ilan University
[1] p8989p72@gmail.com, [2] Hsin-Hung@ieee.org

Abstract

In the case of web security, the administrator usually prepares a large list of attack vectors when perform security testing. Some well-known free vulnerability scanning tools use a list of out-of-the-box attack vectors, while others generate attack vectors based on a known attack syntax format. Although this approach can save a lot of time and labor costs, it is able to only test problems which have been identified, and sometimes the success rate is not high.

To increase the efficiency of security testing, we will try to increase the diversity of attack vectors to uncover more vulnerabilities. Therefore, we proposed an automatic security testing system by using artificial intelligence techniques. With the powerful search ability of artificial intelligence for unknown areas, it has the ability to generate pseudo-data features out of thin air so that the attack vectors can be learned and generated. We can take advantage of that to make a second choice to test the website for the administrator.

Keywords: Web Security, Artificial Intelligence, Web security test system, Web security testing

1. 簡介

安全評估涵蓋許多不同的做法，評估的類型不同，做法也大不相同，白箱的安全評估有原始碼分析，黑箱則有模糊測試，儘管已經透過這些方法發現許多漏洞，但是如何百分之百徹底評估測試目標是否已被發現所有的漏洞，一直是個大問題。

在安全測試中，測試目標非預期的輸入值有無限多筆，需要測試的攻擊語法便有無限多種，所以要百分之百確定所有攻擊語法是否能觸發漏洞，基本上是不可能的，不過我們可以盡可能的增加安全評估的效果，提前駭客一步發現安全漏洞，避免這些安全漏洞被這些有心人士利用，網頁的安全問題更是首當其衝，因為許多的駭客攻擊都是以企業的網站當作開始，開放式 Web 應用程式安全項目(Open Web Application Security Project，OWASP)便整理出十種最常被找到的網頁攻擊手法，在 2013 年跨網站腳本漏洞(Cross-Site Scripting，XSS)更是排名第三，現今大部分的評估方法，除了使用現成已知的攻擊語法清單外，有些只是基於已知的攻擊格式來生成攻擊語法，這樣的方式極不合理，且將付出極大的人力與時間成本。

綜合以上的問題，在現今安全評估技術的基礎上，我們要如何改進舊有的方法，並創造新的方法來發現新的問題？因此我們提出了一套結合生成對抗式網路(Generative Adversarial Network，GAN)的自動化網頁安全測試系統，來針對網頁進行安全評估，我們導入生成對抗式網路來學習和產生跨網站腳本的攻擊語法，藉由這種方式增加攻擊語法的變化性，並且能夠加強安全評估的效果。

在本文的第一章，我們介紹目前網頁安全中，跨網站腳本漏洞(Cross-Site Scripting，XSS)的現況，也描述了研究動機與解決方法。第二章節會介紹有關安全測試和深度學習的發展與應用。第三章節則介紹我們提出來的自動化網頁安全測試系統。第四章節會以實際的案例，來展示我們所提出的自動化網頁安全測試系統。最後，第五章節會針對本文進行總結。

2. 背景及相關研究

2.1 網頁安全測試

測試應用程式是多樣的，可以測試它的功能有沒有符合要求，也可以測試運行的過程是否有符合邏輯，隨著駭客在網路背後的惡意行為，測試應用程式安全也變成必要的。安全性根據應用程式的特性不同，測試的方向也大不相同，在測試前我們必須先釐清測試目標的攻擊面為何，以及如何將攻擊向量傳輸至攻擊面，才能夠開發有效益的測試系統，當測試目標為網路協定時 [1]，攻擊面即為被連接的網路伺服器應用程式，攻擊向量則需要構造網路協定封包來做測試；當測試目標為某種格式的檔案 [2]，攻擊面即為檔案解析器的介面，攻擊向量則需要依據特定格式，構造出檔案來作為攻擊向量。

於 1990 年時，Barton P. Miller 等人提出一種測試方法為模糊測試 [3]，模糊測試是透過各種非預期的資料輸入到測試目標中，誘使測試目標發生例外狀況或崩潰，然後將這些狀況加以記錄並分析，期望能從這些紀錄中找到尚未被發現的安全漏洞，最後程式

設計師會修補這些問題來增加系統的安全性。該篇論文便以隨機生成的測試資料來測試 UNIX 程式 [3]，並發現許多潛藏的安全問題，雖然他們展示了模糊測試在漏洞挖掘上的有效性，但構造測試資料應該會有一定的規則，因為如果我們從數學的角度來看，測試目標非預期的輸入值將有無限多種組合 [4]，我們很難去枚舉所有組合來做測試，所以我應該將無限多種組合拆分成各種測試方向。

根據安全人員對測試目標的掌握程度，又可以分為白箱模糊測試與黑箱模糊測試。在白箱模糊測試中，測試人員擁有完整測試目標的原始碼，透過分析原始碼的過程中，可以有效的列舉程式碼中各個有可能發生安全問題的函式，同時也能從開發環境中的除錯功能，確定能夠測試的深度，這在漏洞挖掘中及其重要，Fabian Yamaguchi 等人的研究中 [5]，便將各個程式發現漏洞的函式導入機器學習模型中學習，拿來判斷測試目標是否有相似的函式，相較於完全未知測試目標結構的黑箱模糊測試，這樣的做法確實能夠找到新的安全問題，測試涵蓋率也最有效的遍歷整個測試目標結構。在黑箱模糊測試中，測試人員必須站在駭客的角度，在完全不了解目標系統的情況下，只能不斷的輸入非預期的測試資料，盡可能的製造目標系統的例外情況與崩潰，案例越多發現問題的機率也越大，雖然黑箱模糊測試是最沒有效率的，但只要是非開源的應用程式通常都不會將原始碼細節公布出來，只能以經驗大概猜測實作的內容，因此黑箱模糊測試是大部分漏洞挖掘最好的測試方案。

其中網頁更是經常以黑箱的方式在做測試，它並不像軟體可以透過逆向分析手法，來分析出應用程式的運作流程，在伺服器端與客戶端之間隔著超文本傳輸協定(Hyper Text Transfer Protocol，HTTP)。在測試網頁安全時，更是需要將範圍再縮小一點，於 [4] 還提到另一個名詞是安全類，從無限多種安全問題，單以個別的網頁安全問題來做測試，以最具代表性的開放式 Web 應用程式安全項目(Open Web Application Security Project，OWASP)整理的十種最常被找到的網頁攻擊手法為參考 [6]。

2013 年跨網站腳本漏洞(Cross-Site Scripting，XSS)在 OWASP Top10 中排名第三，漏洞發生原因在於網頁將使用者輸入當作 Javascript 語法在執行，因此攻擊者可以將惡意腳本注入到網頁上，使瀏覽網頁者受到攻擊，因此是一種以使用者作為攻擊目標的安全問題，根據 Amit Klein 的研究 [19]以及 OWASP 的資料 [7]，跨網站腳本漏洞可以分成反射式(Reflected)、儲存式(Stored)、DOM-based 三種類別(圖 1)，反射式發生在使用者的輸入能夠立即性的將執行結果從網頁回傳，因此它是非持續性的，一般攻擊語法會附帶於網址的參數後面，而儲存式則發生在使用者資料會儲存於後台或者資料庫中，它是持續性的直到被管理者清除，DOM-based 跟反射式很像，只是發生在 DOM 元素的資料中，你也可以說它是反射式的。

	Where untrusted data is used		
Data Persistence	XSS	Server	Client
	Stored	Stored Server XSS	Stored Client XSS
	Reflected	Reflected Server XSS	Reflected Client XSS

圖 1、跨網站腳本漏洞的類型[7]

在跨網站腳本漏洞的測試方法上，OWASP 也有明確說明 [8]，黑箱測試至少包括三個階段，首先要檢測輸入參數，對於測試目標的每一個頁面，測試人員必須確定所有被網頁所使用的變數以及網頁接收的 HTTP 方法，例如 HTTP 參數、POST 資料、隱藏欄位的變數，以及開發者預先定義好的變數等等。第二階段分析每個輸入參數確認潛在漏洞，將事先構造好的攻擊語法注入後，將會從瀏覽器觸發響應，第三階段則確認結果，檢查返回的網頁超文本標記語言(HyperText Markup Language，HTML)，搜索測試輸入是否涵蓋在其中，特殊符號是否被測試目標做轉譯處理。

2.2 深度學習

近年來人工智慧的議題非常火熱，不斷的被拿出來討論，甚至將人工智慧應用於各個領域中，人工智慧代表的僅僅是一種模式，最簡單純粹以 if-else 語法來組成的系統，我們也可以說它具備人工智慧的特質，直到後來有了深度學習，最早我們可以追溯到 1957 年 Frank Rosenblatt 提出的感知器模型，它與我們大腦內的神經細胞有些相似，現今人工智慧專家也希望能夠越發掘大腦神經細胞的奧秘，同時也能借鑒生物科學好研發出更具智能且更趨近於人類大腦的模型。

最簡單的神經網路僅有一個神經元，有一個輸入，一個輸出，它基本上就是一個普通的函式，而後 Frank Rosenblatt 提出的感知器，一個神經元可以接收多筆輸入，並經過計算得到多筆輸出，學習過程透過模型的監督式學習，通過期望值不斷的修正權重，直到獲得一個最佳的權重，就能拿它來做判別了，只不過它的構造簡單，以至於只能拿它來學習像 AND 或 OR 運算這種簡單的問題，複雜一點的如 XOR 運算就沒辦法了。特徵值是機器學習訓練時必備的要素之一，當特徵值精確度越高，選擇越能代表這項資料的特徵，判別的成功率也會越高，當我們在訓練一張圖像時，我們可以將一張圖像擷取出好幾個不同的特徵如圖 2 所示，擴大資料量，來達到訓練的有效性，但是當一張圖像的

特徵被擷取的顆粒度越細，細到看不出與原圖像的關聯性，這時候所需的訓練次數就大大增加。人為提取特徵是非常耗時耗力的，因此嘗試讓機器自動學習特徵，是深度學習的特性之一，深度學習中每個模型各有它擅長之處，所以在應用時，應該挑選適合的模型來使用，其中卷積神經網路(Convolutional Neural Networks，CNN)擅長做圖像學習，訓練時將圖片上所有相同大小的區塊做卷積操作，進行特徵的提取。除此之外還有擅長學習序列資料的遞歸神經網路(Recurrent Neural Network，RNN)，訓練時會將前面訓練的訊息的一部分加入下一層的輸入一起訓練，因此比較大的特點在於能夠學習文章的前後文，許多深度學習研究者會在深度學習模型上做一些改動，並且研究新的改動能為該模型帶來時麼樣的變化，其中就有長短期記憶網路(Long Short Term Memory Network，LSTM)源自於遞歸神經網路的改動，在原始架構的基礎下，加入了遺忘閥(Foget Gate)來記憶前面的字詞，一般的遞歸神經網路學習，能夠考慮前一個字詞，來預測下一個字詞為何，但隨著距離該字詞越遠，權重將會逐漸減低，越不能影響下一個字詞的生成，而長短期記憶網路則不一樣，能由遺忘閥來決定記憶的儲存及字詞的權重，它可以選擇是否要將距離較遠的字詞加入遺忘閥，繼續影響其他字詞的預測，也可以選擇就直接丟棄掉，譬如當下的字詞是新主題，或者是完全相反意思的字詞，就能夠丟棄該字詞，好騰出空間給下一個主題記憶新的字詞。

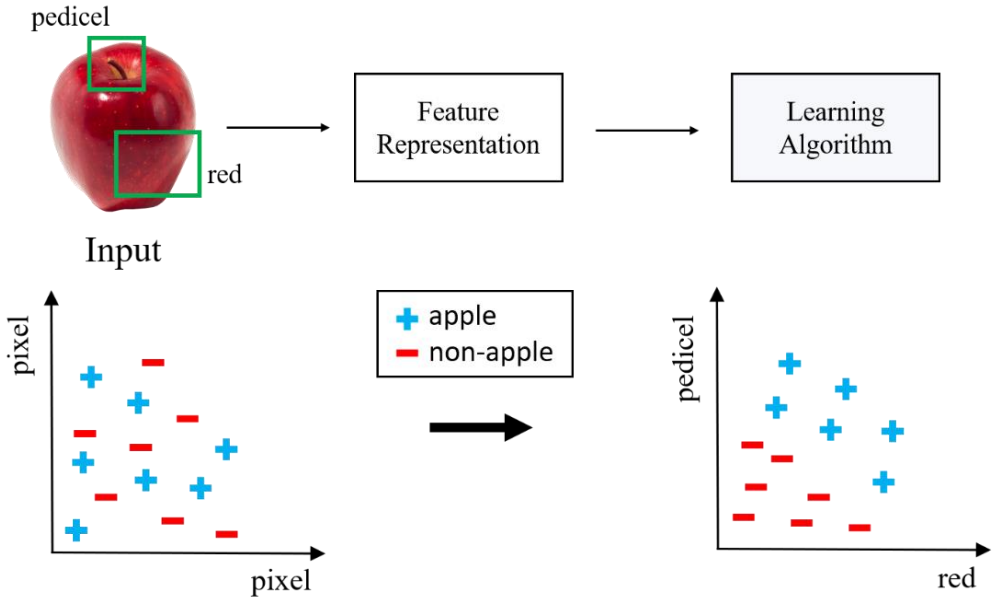

圖 2、特徵提取

2.3 基於深度學習的模糊測試

將資安與深度學習結合的應用非常多，不過大部分都比較偏防禦或者分析，譬如惡意程式分析 [11]、入侵偵測 [12]或者惡意網址分類 [13]，這是因為深度學習模型最主要作用還是以分類或辨識為主。

近年來，黑帽駭客大會才開始出現比較多深度學習與攻擊結合的相關研究，Isao Takaesu1 便開發了一套結合機器學習的全自動化的滲透測試工具 [14]，在訓練階段，對攻擊目標做網路掃描，偵測測試目標開的通訊埠，來判斷測試目標的作業系統及啟動的服務，將這些資訊交由深度學習模型學習，判斷要使用哪一種攻擊模組，接下來對自己架設的靶機進行嘗試性攻擊，以強化學習的方式學習到正確的攻擊流程，這種讓深度學習來規劃攻擊策略，對於不懂滲透測試的一般人員是有幫助的，可以減少測試的學習門檻，但對專業的資安人員來講，頂多是作為輔助而已，畢竟在攻擊時，還是得依靠經驗以及對網路環境架構足夠的了解，除了這種策略型的之外，還有一種是讓深度學習學習攻擊語法。

我們經常能在資安新聞中，看到駭客以網頁釣魚技術促使目標使用者資料大量的外洩，大部分惡意病毒也都是從惡意網站感染到使用者電腦，因此 A.C. Bahnsen 等人為了研究攻擊者如何進行有效的攻擊，分析惡意網址以了解攻擊者構造釣魚網址的不同 [15]，利用遞歸神經網路學習大量已辨識的惡意網址，用來突變惡意網址使它能夠繞過 AI 釣魚檢測系統。類似的應用還有 2014 年 Fabien Duchene 等人利用機器學習來加強跨網站腳本漏洞黑箱模糊測試的偵測效果 [16]，先逆向分析出整個網站流程，根據跨網站腳本漏洞的攻擊語法產生模糊測試資料，並評估每一筆測試資料與跨網站腳本漏洞的相關度，當相關度越高，基因演算法的進化過程越有可能選擇該個體的基因來創造新的一批測試資料。

類似的應用還有 2014 年 Fabien Duchene 等人利用機器學習來加強跨網站腳本漏洞黑箱模糊測試的偵測效果 [16]，先逆向分析出整個網站流程，根據跨網站腳本漏洞的攻擊語法產生模糊測試資料如圖 3 所示，並評估每一筆測試資料與跨網站腳本漏洞的相關度，當相關度越高，基因演算法的進化過程越有可能選擇該個體的基因來創造新的一批測試資料。

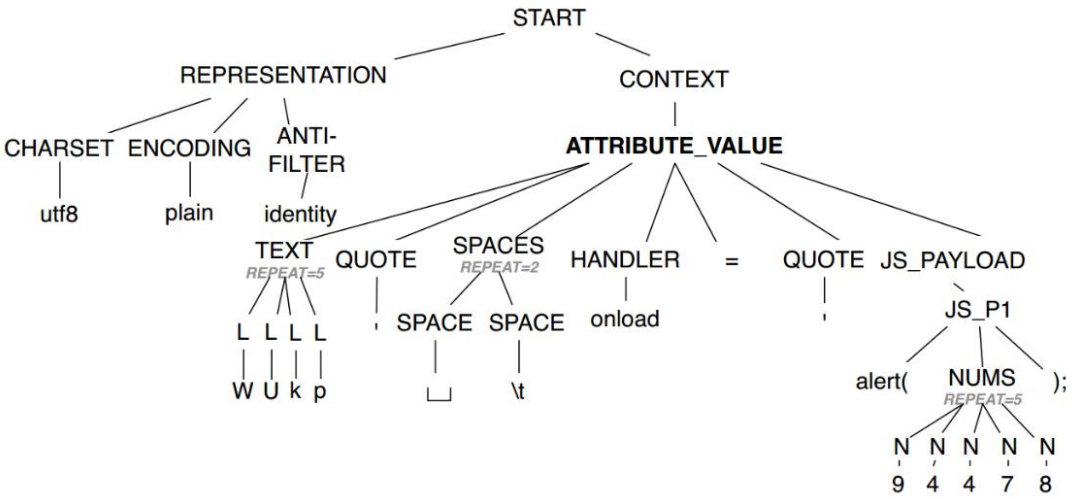

圖 3、測試資料語法生成樹狀圖 [16]

3. 解決方案

3.1 基於深度學習的全自動安全性測試系統系統架構

現有的開源工具中幾乎都是依照特定的格式去做生成，即時有相關研究使用了機器學習的方式來幫助測試資料的生成 [16]，但仍然避開不了需要設定攻擊語法，這樣的方式不但無法增加測試資料的多變性，甚至無法將此框架應用於其他攻擊手法，因此根據[20]的作法，以及前述提到其他的相關背景，我們結合各項研究的優點，並提出一套結合生成對抗式網路的自動化網頁安全測試系統，實現了跨網站腳本漏洞的自動化測試，利用生成對抗式網路的「憑空」生成偽資料的特性，希望能夠生成出新的攻擊手法。系統架構如圖 4 所示，從圖中可以分成兩大部分：測試資料的產生，以及測試腳本的自動化。測試資料的產生由生成對抗式網路負責，訓練資料再交由對抗式生成網路訓練前，必須要先經過預處理階段，從收集資料開始，有些資料會將特定字符經過編碼，為了方便生成對抗式網路可以更好地做訓練學習，訓練資料必須為序列資料，且長度必須一致，所以我們將語法分類如表 1 之測試資料 Tokenization 的轉換，，首先將這些資料分成三等份，其中一份用來作為訓練，另一份作為測試，剩下的拿來驗證練，當處理完後開始訓練資料階段，先對生成對抗式網路的生成器與判別器做預訓練，好讓兩個網路有一些基本的生成資料能力以及分類二元資料的能力，接下來生成器會生成一些資料讓判別器去判斷，當資料訓練結束後，接下來就能夠生成資料了，最後生成的資料就能交由測試腳本對目標網站進行測試。

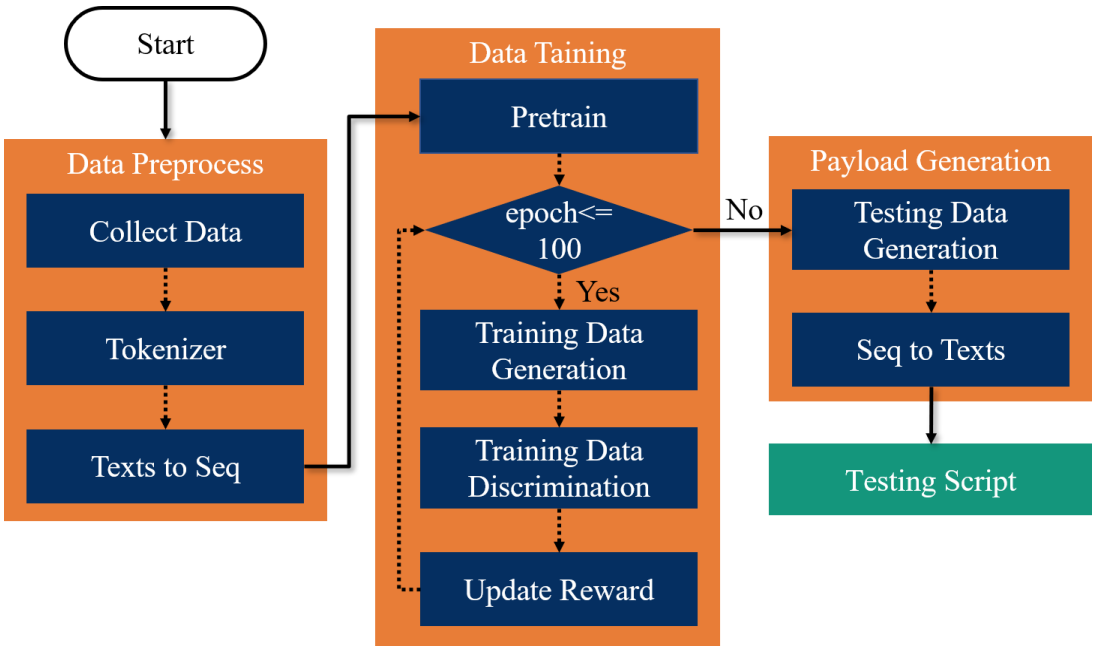

圖 4、系統架構與運作流程

表 1、跨網站腳本攻擊的語法分類[1]

Classification	**List**
Start Label	<script>, <frame>, <img, <body>, etc
End Label	</script>, </frame>, </body>, etc
Windows Event	onerror=, onload=, onblur=, oncut=, etc
Function Name	alert(, prompt(, String.fromCharCode(, etc
Script URL	javascript:, vbscript:, etc
Others	>,), #, etc

3.2 針對網頁的深度學習安全測試

接下來我們將針對我們所使用的生成對抗式網路做更詳細地介紹，一般生成對抗式網路都會以兩個卷積神經網路來做搭配，例如 DCGAN [9]，而使用這類生成對抗式網路訓練序列資料是困難的，因為這類生成對抗式網路並不擅於處理離散型資料，譬如顏色與顏色可以產生漸層效果，即時像素的 RGB 值做些微的變化也不影響結果，而文字資料則無法，從英文字母 A 到 Z 間，無法明顯得看出漸變過程，必須要將字母依照 ASCII 碼轉變為十六進制才能看出，而且當數值做細微調整時，結果將跟原先的完全不同，圖片像素細微的改變從視覺上來看是沒問題的，甚至肉眼也無法發現差異，但是假若要訓練文字，當我們對文字的十六進制值做些微的調整時，整個語意將改變，甚至毫無意義。

在模型的挑選上必須選擇更適合我們研究的，跨網站腳本測試資料的生成重點在於學習 Javascript 的語法，這種文字上的處理，交由負責文本相關應用的長短期記憶網路最適合了，同時也較遞歸神經網路更能學習字詞間的依賴訊息，激勵函數則使用當前主流的方法 tanh。而判別器由於模糊測試仰賴測試資料的變化性，因此需要完美的判斷能力，因此相較於文本的處理，則可以選擇善於處理圖像的卷積神經網路，激勵函數則使用 relu，讓深層的梯度能傳送到淺層，也避免梯度消失的問題，除此之外在學習率的設置上，也避免設置的過於高，導致訓練過程中參數的更新太大，另外，不管是生成器還是判別器，兩者都是完整的深度學習模型，所以也都有各自的參數，包括隱藏數、神經元數、學習率等等，在這種情況下一旦有一邊參數設定的不妥，將比單一個深度學習模型容易發生過度學習(Overfitting)的情況。

訓練次數過多，會過度學習，訓練測試過少，將形成毫無意義的模糊測試，因此本研究提出一個名詞為「相似度」，在未過度學習前，相似度是兩個生成資料詞集的相似程度，我們希望在訓練的過程中，找到一個平衡點，除了讓生成的測試資料能夠與測試資料有相符之處，還不歸類到測試資料任何一筆，也就是全新的一筆測試資料。

由於前述問題點，因此我們可以透過調整以下三個策略來完成我們的目標：(1) 測試向量 (2) 分詞策略 (3) 神經網路參數。至於語法的類別，於 2018 年 Y.Fang 等人以深度學習來偵測跨網站腳本漏洞 [12]，研究中針對跨網站腳本語法做了分類，並設計了一系列語法的正則表示式，好可以標記訓練資料。

但選擇要切割的片段越多樣，所要學習的複雜度也越高，訓練的次數也越多，因此我們將原先的分類做個調整，分成幾種策略來進行訓練和學習，並分析這幾種策略的訓練結果，如表 2。第一種策略是以大於和小於組成，由於跨網站腳本漏洞的測試資料由 HTML 語法構成，HTML 語法中最重要的元素為標籤，將代表網頁外觀特性的字詞由大於和小於括起來，因此測試資料中最普遍的也是這兩種符號，如表 3 所示。如果只以第一種策略，變化性並沒有很大，因為測試資料中還有大量的 Javascript 語法，語法有多種表現形式，因此第二種策略以函式的等號和外部腳本網址的冒號組成。除了以上兩種，測試目標為了避免遭受跨網站腳本的攻擊，經常會以消毒的方式，將字串中的特殊符號

經過轉譯處理，所以測試資料為了繞過轉譯處理，會以編碼後的形式出現，因此最後一種策略由多種編碼會出現的特殊符號,如 HTML 編碼的井字號以及 URL 編碼的百分號，如表 3 所示。

表 3、分詞策略

Policy	Symbol	Discription
1	<.*>	Start&End Label
2	*=, *:	Event, Func, URL
3	#, %...	Others

表 2、跨網站腳本測試資料的範例

Start&End Label	<script>alert(1)</script>
Func Label	"><img src=x onerror=alert(1)>
Script URL	"><iframe src='javascript:alert(1);'>
Others	%3cscript%3ealert(1)%3c/script%3e

4. 實驗結果

4.1 網頁安全性測試實驗環境

在實驗環境的部分，我們以 Wordpress 作為測試目標，整個測試目標的架構，如圖 5，網頁伺服器以 Apache 架設，PHP 程式作為網頁後端處理傳送而來的 HTTP 封包，Wordpress 作為內容管理系統(Content Management System，CMS)必須有資料庫來儲存帳號密碼、貼文、留言板等資料，在黑箱測試中，透過封包以及網頁原始碼中蒐集到的各項版本訊息如表 4，這些版本訊息關係到測試結果，因為漏洞的發生往往在於開發者撰寫不安全的程式碼，或者系統的不安全設定，在收集訓練資料的部分，我們從網路上蒐集多筆跨網站腳本漏洞的攻擊語法作為生成對抗式網路的訓練資料 [17] [18]，總共有一千筆資料，能夠發現這些資料具備表 3 所提到的四種表現形式。

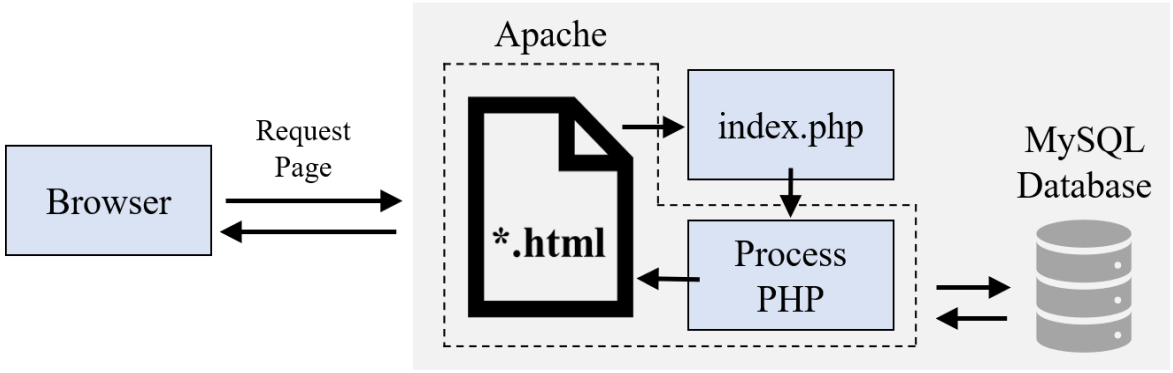

圖 5、測試目標架構

表 4、測試目標各組件版本訊息

Environment	Version
Web Server	Apache 2.4.7
Programming Language	PHP 5.5.9
Content Management System	Wordpress 3.2.1
Plugin	None
Database	MySQL

在評比安全測試結果的優劣，我們以能夠發現新的問題作為理想的結果，但是找到新的問題是一件極其困難的事情，因此除此之外我們還以生成的測試資料攻擊的成功與否及成功率高低作為評比，攻擊的成功率越高，說明網頁安全測試系統查找跨網站腳本漏洞越準確，且生成對抗式網路所生成的測試資料越接近於原始資料。

在參數設定上，由於參數的優劣實在很難透過攻擊的成功率展現，有時候成功率低，但卻能從生成的測試資料中發現新的攻擊，因此參考了 SeqGAN 中實驗的參數設定 [10] 大部分應用於文本生成的生成對抗式網路參數設定，參數如表 5、表 6 所示。

表 5、生成對抗式網路的生成器參數設定

Parameter	**Value**
Hidden Layer	256
Cell Size	256
Learning Rate	0.01
Activation Function	tanh
Optimizer	adam + clip gradient

表 6、生成對抗式網路的判別器參數設定

Parameter	**Value**
Hidden Layer	12
Cell Size	256
Learning Rate	0.0001
Activation Function	relu
Optimizer	adam

4.2 結果分析

在開始測試前，首先我們列舉測試目標可以被測試的參數，參數來源的位置極為多樣，網頁提供的功能不同，參數也不同，在各個功能下實作不同的 HTTP 方法，傳送參數的方式也不同，毫無插件的狀態下，發現 Wordpress 僅具備幾項功能，包括已版本參數來引用 Javascript 腳本，以及登入介面、忘記密碼介面、回覆貼文，其中僅版本參數使用 GET 方法來傳輸，我們將原始資料以及透過深度學習模型生成的測試資料，針對所有功能的參數皆做測試，並且做個比對，如表 7，發現兩者皆能於回覆貼文的功能中發現跨網站腳本漏洞，由此可見，我們運用生成對抗式網路生成的資料同樣具備查找漏洞的能力，除此之外我們將原始資料能夠查找漏洞的筆數做個紀錄，如表 8。

訓練資料總共 1000 筆，對於 Wordpress 網站的回覆貼文功能，能成功查找跨網站腳本漏洞的筆數僅有 193 筆，能夠觸發跨網站腳本漏洞的情況有不同種語法，這也造就每一筆資料能夠產生不同效果，為了發現生成對抗式網路是否能產生新的攻擊或新的問題，我們將這些測試成功的資料根據它產生的行為進行分類，最後我們將原始資料的行為上分成五種，如表 9 所示。在固定參數及訓練次數的前提下，我們將原始資料中能夠測試成功的 193 筆資料投入訓練，檢視不同成功率的資料集在不同的策略中，能夠測試成功的機率有多少，結果於表 10，可以發現成功率越高的資料集生成的資料，表現遠好於成功率低的資料集生成的資料，除此之外策略一的表現最佳，再來才是策略二，表現最差的為策略三。

表 7、輸入向量及漏洞查找結果

Feature	**HTTP Method**	**Parameter**	**Original Data**	**Generated Data**
Other	GET	s, ver,...	×	×
Login	POST	log, psw,...	×	×
Forgot Password	POST	user_login, redirect_to	×	×
Comment Reply	POST	comment,...	✓	✓

表 8、原始資料的測試結果

Total	**Success**	**Ratio**
1000	193	19.3%

表 9、原始資料的行為分類

行為分類	說明
Hyperlink	讓文字變成超連結，有可能造成網路釣魚。
Button	形成按鈕或其他輸入框，也有可能造成網路釣魚。
Load Image	載入不存在的圖片，導致駭客能夠將 Javascript 語法透過 onerror 事件觸發，除此之外，駭客也能夠將惡意程式隱藏在圖片內。
Popup Windows	彈出視窗。
Disable Function	將 HTML 原始碼被註解掉，導致網頁在外觀上失去它原有的功能，也可能被駭客以其他頁面做取代。

表 10、不同資料集於不同策略中的表現

Policy	Training Data	Generated Data	Success	Ratio
1	1000	300	20	6.7%
	193	300	151	50.3%
2	1000	300	8	2.7%
	193	300	140	46.7%
3	1000	300	5	1.7%
	193	300	85	28.3%

5. 結論

跨網站腳本漏洞一直是網頁開發者難以完全根除的問題，即時有網頁防火牆作為靠山，仍然有許多繞過技巧可以使用，有鑑於現今的安全評估技術，有白箱的方式，從原始碼分析中找到有問題的程式碼，或者黑箱的方式，以模糊測試發送非預期的資料，期望從發生崩潰的紀錄中找到問題點。所以我們希望從這些舊有的技術中，開拓一條新路，期望透過深度學習來學習漏洞的攻擊技術，因此我們藉由生成對抗式網路經過訓練之後，能憑空生成一筆原始資料沒有的資料，希望能生成新的跨網站腳本測試資料，同時保留測試資料變化的彈性。我們提出了一套結合生成對抗式網路(Generative Adversarial Network，GAN)的自動化網頁安全測試系統，來學習和產生跨網站腳本的攻擊語法，我們以不同的分詞策略及參數來預處理訓練資料，藉由這種方式來掌控攻擊語法的變化性，並且能夠加強安全評估的效果。

致謝

本研究由科技部計畫 MOST 108-2221-E-197 -012 -MY3 補助支持，特此致謝。

References

[1] S. Gorbunov and A. Rosenbloom, "AutoFuzz: Automated Network Protocol," International Journal of Computer Science and N etwork Security, pp. 239-245, August 2010.

[2] P. Godefroid, H. Peleg and R. Singh, "Fuzzing, Learn&Fuzz: Machine Learning for Input," in IEEE/ACM International Conference on Automated Software Engineering, Urbana-Champaign, Illinois, USA, 2017.

[3] B. P. Miller, L. Fredriksen and B. So, "An Empirical Study of the Reliability of UNIX Utilities," Communications of the ACM, vol. 33, no. 12, pp. 32-44, 1990.

[4] P. Hope and B. Walther, Web Security Testing Cookbook, O'Reilly Media, 2008.

[5] F. Yamaguchi, F. ’. Lindner and K. Rieck, "Vulnerability Extrapolation：Assisted Discovery of Vulnerabilities using Machine Learning," in WOOT'11 Proceedings of the 5th USENIX conference on Offensive technologies, San Francisco, CA, 2011.

[6] D. Wichers, "OWASP Top-10 2013," OWASP Foundation, 2013.

[7] D. Wichers, "Types of Cross-Site Scripting - OWASP," OWASP Foundation, [Online]. Available: https://www.owasp.org/index.php/Types_of_Cross-Site_Scripting.

[8] M. Meucci and A. Muller, "OWASP Testing Guide v4," OWASP Foundation, 2014.

[9] A. Radford, L. Metz and S. U. Chintala, "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks," in ICLR, 2016.

[10] L. Yu, W. Zhang, J. Wang and Y. Yu, "SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient," in AAAI, 2017.

[11] O. E. David and N. S. Netanyahu, "DeepSign: Deep learning for automatic malware signature generation and classification," in International Joint Conference on Neural Networks, Killarney, Ireland, 2015.

[12] Y. Fang, Y. Li, L. Liu and C. Huang, "DeepXSS: Cross Site Scripting Detection Based on Deep Learning," in International Conference on Computing and Artificial Intelligence(ICCAI), 2018.

[13] A. C. Bahnsen, E. C. Bohorquez, S. Villegas, J. Vargas and F. A. González, "Classifying Phishing URLs Using Recurrent Neural Networks," in APWG Symposium on Electronic Crime Research, Scottsdale, AZ, USA, 2017.

[14] I. Takaesu, "Deep Exploit: Fully automatic penetration test tool using Machine Learning," in BlackHat, 2018.

[15] A. C. Bahnsen, I. Torroledo, L. D. Camacho and S. Villegas, "DeepPhish : Simulating Malicious AI," in APWG Symposium on Electronic Crime Research, 2018.

[16] F. Duchene, S. Rawat, J.-L. Richier and R. Groz, "KameleonFuzz: Evolutionary Fuzzing for Black-Box XSS Detection," in Proceedings of the 4th ACM Conference on Data and Application Security and Privacy, San Antonio, Texas, USA, 2014.

[17] D. Wichers, "XSS Filter Evasion Cheat Sheet - OWASP," OWASP Foundation, [Online]. Available: https://www.owasp.org/index.php/XSS_Filter_Evasion_Cheat_Sheet.

[18] I. Tasdelen, "ismailtasdelen/xss-payload-list: Cross Site Scripting (XSS) Vulnerability Payload List," [Online]. Available: https://github.com/ismailtasdelen/xss-payload-list.

[19] A. Klein, "DOM Based Cross Site Scripting or XSS of the Third Kind," Web Application Security Consortium, 2005.

[20] I. Takaesu, "Method of detecting vulnerability in Web," in CODE BLUE, Tokyo, 2016.