無痕瀏覽模式調查惡意攻擊跡證

Evidence Investigations to Malicious Attacks in Incognito Browsing Mode of Internet Systems

莊禾暘
國立政治大學資訊科學系

左瑞麟
國立政治大學資訊科學系

柯宏叡*
國立中興大學資訊科學與工程學系

洪國寶
國立中興大學資訊科學與工程學系

王旭正
中央警察大學資訊管理學系
sjwang@mail.cpu.edu.tw
*whom correspondence

摘要

目前物聯網（Internet of Thing, IoT）發展已來到第四階段，也就是透過既有的 Web 標準來達成設備間互相通訊，稱之為 WoT（Web of Things）。對於新的趨勢，所會面臨到的安全議題不僅止於 IoT 連線設備，亦包含 Web 應用程式漏洞。而諸如無痕瀏覽模式等匿蹤技術的發展，使得鑑識人員於調查過程中遇到阻礙。因此，本文將著重於在無痕瀏覽模式下的 SQL Injection 與 Cross-Site Scripting（XSS）的攻擊手法進行分析，藉由記憶體鑑識技術找出攻擊端及受駭端之證據關聯性，建立完整鑑識程序。

Abstract

At present, the development of the Internet of Things (IoT) has reached the fourth stage. That is to say, being in the stage of communications between devices through existing Web standards, which is called WoT (Web of Things). For the upcoming new trends, the security issues not only face IoT connected devices, but also include web application vulnerabilities. The development of hidden technologies such as incognito browsing modes has made forensic investigator encounter obstacles in the evidence checking process. Therefore, this paper will focus on the analysis of SQL Injection and Cross-Site Scripting (XSS) attack methods in incognito browsing mode. It is by the exploration of memory forensics to find out

the correlation between the attacking and hacked terminal-ends. Then establish more complete forensic procedures in our study works.

關鍵詞: 記憶體鑑識、Web 應用程式漏洞、SQL Injection、Cross-Site Scripting

壹、前言

隨著資通訊技術的快速成長及進步，網際網路的各種創新應用不斷地被提出，改變了人們的生活及溝通方式。從雲端運算開始，這些創新技術更有爆炸性的發展，近年來熱門的大數據及物聯網（Internet of Thing, IoT）應用，更是風靡全世界。IoT 目前可運用的範圍相當廣泛，例如智慧家庭（Smart Home）或車聯網（Internet of Vehicle）皆是目前熱門的發展方向。尤其穿戴式裝置的普及，更促進了 IoT 的蓬勃發展。

IoT 的核心是實現設備之間能彼此互連，為達成此一願景，目前正邁向 IoT 發展的第四階段- WoT（Web of Things），也就是使用現有之 Web 標準來達成設備之間互相通訊。但是水能載舟，亦能覆舟，駭客亦可使用瀏覽器之無痕瀏覽模式，透過 SQL Injection、Cross-Site Scripting(XSS)等針對 Web 應用程式的漏洞攻擊，以減少所會留下之數位跡證。比起電腦，IoT 的相關應用涉及更多的隱私以及人身安全。若遭受到駭客進行攻擊，不僅會造成使用者損失，甚至會危害使用者的生命安全。

Web 應用程式的漏洞諸如 SQL Injection、XSS 等，多年來皆為常見之 Web 應用程式安全漏洞。顯示出雖然攻擊手法並非運用最新的技術，卻依然可以達到有效的攻擊。Kaur 等人即從 Web Hacking Incident Database(WHID)以及多個網路安全相關新聞網站等蒐集了網路攻擊事件，並且進行統計分析[11]，發現各個領域所遭受到的攻擊雖然不盡相同，但是 SQL Injection 以及 XSS 等都是各領域常見的攻擊。而根據 WhiteHat Security 發佈的第十二期《2017 應用安全統計報告》[25]，大多數的 Web 應用程式包含有至少三個安全漏洞，且其中近一半是極具威脅性的，也就是說，多數的 Web 應用程式都是置身於易受攻擊的環境。因此對於了解 Web 應用程式安全漏洞形成原因，以利後續預防之依據，亦不可忽視。

此外，由於執行 Web 應用程式時須透過瀏覽器，而瀏覽器業者在基於隱私的概念下，提供了無痕瀏覽模式。無痕瀏覽模式可以讓使用者在關閉瀏覽器後，不會紀錄使用者的瀏覽紀錄以及使用者於表單中所輸入的資訊等。在此模式操作下所會留下的資訊較少，對於駭客而言，是個可以藏匿所做之惡意行為的媒介。卻也使得鑑識人員在調查的過程更為艱辛，將會造成鑑識人員較難以完整的還原案件，然而任何應用程式執行時，作業系統都會將資料暫存於隨機存取記憶體(以下簡稱記憶體)中，使其得以順利執行。這顯示出記憶體鑑識的重要性。

本文嘗試使用記憶體鑑識技術作為鑑識蒐證方法，研究受到 Web 應用程式漏洞如

SQL Injection 及 XSS 攻擊後，於記憶體中所會留下的數位跡證，以協助鑑識人員還原整起資安事件。本文結構編排如下:在第二節將會介紹 Web 應用程式漏洞攻擊以及就相關文獻進行探討；在第三節提出記憶體鑑識分析方法；並且在第四節針對本文所提出的記憶體鑑識分析方式進行驗證，同時將所得到結果進行整理與說明，最後於第五節提出研究結論。

貳、相關研究

本文基於記憶體鑑識技術探討受到 Web 應用程式漏洞攻擊及感染惡意程式後所會留下的跡證，其中 Web 應用程式漏洞攻擊以影響重大的 SQL Injection 及 XSS 作為範例，在此先介紹 SQL Injection、XSS 及其相關研究文獻。

2.1 SQL Injection

由於網站中大多數的重要資料，如商品資訊、個人資料等皆儲存於系統的資料庫中，因此資料庫的保護特別重要，而駭客也最常針對資料庫進行攻擊。資料庫攻擊包括有漏洞入侵、SQL Injection、猜測使用者帳號密碼以及使用資料庫預設帳號密碼等攻擊手法，其中 SQL Injection 是最常見的資料庫攻擊。若要透過預防措施減少攻擊的發生機率，首要條件為須先了解此類攻擊的手法。因此，Sharma 等人將 SQL Injection 攻擊手法分為代碼注入(Orderwise Injection)、盲注(Blind Injection)以及攻擊資料庫(Against Database)三大類進行探討[21]。

SQL Injection 是常見的 Web 應用程式攻擊手法，由於 Web 應用程式在開發設計過程中設計不良，或系統工程師疏於檢查，導致攻擊者能夠惡意輸入夾帶 SQL 指令的字串，使得資料庫伺服器誤認為是正常的 SQL 指令而執行，造成資料庫遭到破壞或入侵。而依據其攻擊語法類型，主要可區分五大類型:

一、可聯合查詢注入(Union Based Injection)

在 SQL 語法中，「/*」、「--」以及「#」皆代表了註解之意，通常在 login 欄位之後為 password 欄位，假如攻擊者在 login 欄位輸入<SQL 語法>#即表示#後面的 password 欄位將被註解掉，因此攻擊者可以不需要輸入密碼即可登入網站。由於通常注射點和攻擊點是一樣的，因此也較容易被發現。

二、二階 SQL 注入(Second Order Injection)

不同於直接把惡意 SQL 指令的字串語法輸入到資料庫伺服器中，爾後立即得到回應。此種類型將 SQL Injection 分成兩個階段進行攻擊，即分為輸入惡意 SQL 指令並儲存於資料庫中以及透過相關請求使得系統執行此惡意 SQL 指令。如網站管理者對於「發佈文章」的功能只分配 INSERT 權限，對於「顯示文章」的功能只分配 SELECT 權限，而對於「修改用戶」的功能只分配了 UPDATE 權限。而攻擊者則可透過先發佈包含有

惡意 SQL 指令的文章後，執行顯示文章的功能，以便觸發惡意 SQL 指令。又如一個可能發生於日常生活中的例子，當攻擊者冒充他人身分打電話給某公司之客服人員，並告知其基本資料有誤，需做更改，如此一來，攻擊者便可更輕鬆的冒用他人身分進行非法行為。

三、基於錯誤的 SQL 注入(Error Based Injection)

此種攻擊方式為攻擊者刻意輸入不正確的語法，藉由資料庫回傳的錯誤訊息，如顯示資料庫表格某欄位輸入值錯誤等資訊，藉此可以得知此資料庫含有那些欄位，以獲取資料庫的相關資訊。

四、基於布林值的注入(Boolean Based Blind Injection)

此種方式稱為盲注，當網站伺服器沒有回傳任何錯誤訊息時，可藉由 TRUE & FALSE 的邏輯關係，推測資料庫的版本、欄位名稱等資訊，如可藉由語法查尋資料庫的某個字元值是否為某個數值等。

五、基於時間的延遲注入(Time Based Injection)

若網站伺服器沒有回傳任何錯誤訊息且無法透過 TRUE & FALSE 的邏輯關係，推測資料庫的資訊，則可藉由時間延遲之技術，得知資料庫的資訊，也就是說，可以藉由若 A=1 則等待 10 秒等方式，藉由回應時間進行推測。

根據 OWASP TOP 10 2017 報告[24]，可知 SQL Injection 為其調查報告排名第一的 Web 應用程式安全漏洞。有鑑於此，多位學者即針對 SQL Injection 進行相關研究。Johari 等人即研究了在延遲/容錯移動感測器(Delay/Fault Tolerant Mobile Sensor Network)和移動對等網路(Mobile Peer to Peer)中的不安全查詢[10]，並且實作於 Oracle 9i 企業版 9.2.0.1.0 版本，描述這些 SQL 查詢如何成為攻擊手法。另外 Moh 等人提出了一套多階段的分析架構[14]，藉由分析日誌文件，檢測最常見的網路攻擊 SQL Injection，透過使用機器學習和模式匹配的方法檢測是否發生 SQL Injection。而 Suteva 等人則針對三種常見類型之 Web 應用程式攻擊：SQL Injection、Stored XSS 以及 Remote File Injection 進行實驗[22]，分析攻擊端主機和受駭端主機之數位證據，在攻擊端主機上，可找到在瀏覽器上所留下來的歷史文件，而在受駭端主機，則可從日誌文件中找到相關跡證，這些都可以幫助鑑識人員重建網路攻擊行為，以利成為法庭上的有效證據。

2.2 Cross-Site Scripting (XSS)

由於網站中「記住密碼」功能是將會員密碼儲存於 Cookies 中，使得有心人士可以藉由 XSS 漏洞竊取會員的 Cookies 值，進而獲取該會員的權限。XSS 亦是常見的 Web 應用程式攻擊手法。攻擊者能夠透過注入使用者端腳本語言如 HTML、JavaScript、CSS 等的惡意程式碼，讓使用者瀏覽網頁時，會自動執行攻擊者所注入的惡意程式碼進而導向攻擊者所指定的另一個網站，而此網站可能會執行自動下載惡意程式、自動抓取使用者 Cookie 值等行為。而依據其攻擊類型，主要可區分兩大類型，分別是需要使用者點

擊特定連結的反射攻擊(Reflected Attack)以及因瀏覽已植入惡意語法的網頁的持續性攻擊(Persistent Attack)。

一、反射攻擊(Reflected Attack)

又稱為非持續性(Non-persistent)攻擊，主要是指惡意的網頁並非存在於目標網站中，而是被動的需要使用者點擊觸發。攻擊者先製作一個惡意網站，透過電子郵件傳送含有此惡意網站連結給使用者，藉由吸引人的標題以及內容，如某知名品牌線上特賣會等的資訊，引誘使用者點擊信件中的網址，而此網址所導向的網站，即為攻擊者所製作的惡意網站。當使用者欲購買商品時，所輸入的個人資料甚至信用卡資訊，便會落入攻擊者手中。

二、持續性攻擊(Persistent Attack)

又稱為儲存性 XSS(Stored XSS)攻擊，主要是指所注入的惡意程式碼，被儲存在目標網站的伺服器中，不需要使用者點擊特定連結及可發生攻擊。此種攻擊手法常見於網站內留言格式不拘的留言板中。攻擊者先將惡意語法藉由網頁中的 XSS 漏洞存入網站伺服器中，當使用者正常的瀏覽目標網站時，網站會從伺服器中讀取惡意攻擊者事先注入的惡意程式碼並且執行。若惡意程式碼的指令為取得使用者 Cookies，則當使用者登入後瀏覽目標網站時，攻擊者即可取得使用者之 Cookies，使得攻擊者不需要知道使用者的帳號密碼便可冒用使用者的身分。

2.3 私密瀏覽與無痕瀏覽

許多人目前都是透過瀏覽器來上網，並透過瀏覽器來尋找文章、購物或是交易等。瀏覽器會記錄並儲存使用者的瀏覽活動，包括所訪問過的網站連結、關鍵字搜尋、cookie 與 cache。這些訊息通常會留在使用者的電腦上一段時間，也未有加密，所以若有涉及犯罪行為，數位鑑識人員就能從這些紀錄中進行蒐證。

然而有些攻擊者，會試圖去竊取使用者的網路蹤跡，導致許多使用者愈來愈重視在網路環境中保護個人隱私，這也使得瀏覽器開發人員必需處理這些涉及隱私的功能。如今主流的好幾項瀏覽器均已有私密模式或無痕瀏覽的功能，如 Internet Explorer 的 "InPrivate 瀏覽"、Google Chrome 的無痕瀏覽（Incognito）、Safari 的私密瀏覽，另外像是 Opera 與 Firefox 均有類似的功能[23]。

原本私密瀏覽的功能在於讓使用者瀏覽網站不會在他們的電腦上留下瀏覽紀錄，也可以避免意外留下網站登入帳號，甚至是密碼的機密訊息[18]。然而一些攻擊者卻反而利用這種模式來進行不法行動，對網站進行攻擊時使用私密瀏覽或無痕模式，藉以隱藏上網紀錄，使鑑識人員無法取得有效的證據。以 Google Chrome 的無痕瀏覽模式為例如圖 1，其未紀錄的檔案如表 1 所示。

表 1: 無痕瀏覽模式未記錄之檔案

瀏覽器	Google Chrome
檔案資訊	1.瀏覽紀錄 2.下載清單 3.Web cache 4.Cookies 5.自動填入表單 6.帳號/密碼

圖 1 : Google Chrome 無痕瀏覽模式

以無痕模式方式瀏覽網站，其實還有可能留下證據，Chivers 等人就微軟的 Internet Explorer 10 進行分析，在使用 Internet Explorer 的 InPrivate 模式後若尚未關機，關閉瀏覽器後未再重開瀏覽器，還是有可能由硬碟中以工具方式取得部分的瀏覽紀錄[3]。在 Montasari 等人的研究中，發現若在無痕模式瀏覽時，Internet Explorer、Firefox、Safari 可被發現還是有活動紀錄會存在硬碟中，而 Chrome 則是未被發現有儲存活動紀錄[15]，因而推測出活動紀錄應可於記憶體中取得。

2.4 記憶體鑑識

無痕瀏覽可提供一般民眾使用後不會留下瀏覽紀錄，犯罪者亦可藉此進行網路攻擊，以隱匿犯罪行為之證據，犯罪者亦可在惡意程式中寫入自毀指令，也就是執行程式後自

我刪除。然而這些行為皆會暫存於記憶體中，因此，記憶體鑑識已是數位鑑識領域中不可或缺的一部分。

在記憶體鑑識的相關研究中，Ahmed et al.與 Heriyanto et al.均針對記憶體鑑識工具進行比較[1][8]。除了現有的記憶體鑑識工具之外，Petroni et al.自行開發記憶體萃取工具[17]；而 Schatz et al.則是結合基於硬體設備及軟體的記憶體萃取方式之優點自行開發記憶體萃取工具[19]。Ghafarian 針對無痕瀏覽器所會留下的數位證據進行研究[6]，而 Hua 則是在匿蹤技術的情況進行記憶體鑑識探討[9]。

無論是使用硬碟中的資料、複製文件、連線上網抑或是在螢幕上顯示相片等在電腦中的行為，都必須要透過記憶體才得以完成。也就是說，所有的訊息都會暫存於記憶體中。因此記憶體中所含有的數據對於鑑識人員而言是重要的線索，其中具有價值的數據如程序(Process)、一般文件(General Files)、網路流量的訊息(Information on Network Traffic)、網路數據(Internet Data)、密碼及加密密鑰(Passwords and Cryptographic Keys)甚至是解密的內容(Decrypted Content)等[7][13]。

不同於一般電腦，嵌入式系統(Embedded System)絕大多數時間皆為開機狀態，並不會關機，且系統相關資料皆儲存於記憶體中。由於嵌入式系統時常用於工業基礎設施，若被攻擊造成損壞，可能會影響全國。因此研究者 Seo 等人針對嵌入式系統之記憶體鑑識進行探討[20]。並透過實驗驗證透過記憶體的萃取及分析可以獲得當下記憶體映像檔之作業系統訊息等重要的訊息。

由此可知記憶體中含有大量可以協助釐清事件的關鍵數據。鑑識人員為了要能夠分析記憶體，首要條件即為取得記憶體。而萃取記憶體的方法可分為透過硬體設備或者軟體工具兩種方式。雖然透過硬體設備取得記憶體之方法可靠度較高，但因硬體設備價格高，使得便利性不足。

最常見的方式是透過軟體工具萃取記憶體。然而使用軟體工具有個缺點，啟用軟體時，工具本身也會像其他電腦中的程序暫存於記憶體中，可能會覆蓋掉一些重要的數據。因此多位研究者針對不同工具對記憶體的影響程度進行探討。也有多位研究者自行開發記憶體萃取工具。

Ahmed 等人針對 Windows 系統之記憶體萃取工具 MoonSols DumpIt、Access Data FTK Imager、WinPmem 1.6.2、Belkasoft RAM Capture、Mandiant’s Memoryze 及 Magnet RAM Capture 此六套工具，透過一般情況及安裝反偵錯器(Anti-debugging tool)的條件下進行工具比較[1]。實驗結果顯示，在一般情況下，所有工具都能萃取記憶體，然而在安裝反偵錯器的條件下，並非所有記憶體萃取工具都能準確地獲得記憶體的數據。鑑識人員可藉由此研究之工具比較結果，依照需求選擇適合之記憶體萃取工具。

惡意攻擊者可能會對操作系統的程序進行修改，以達到隱藏惡意程式的目的，若使用一般的記憶體鑑識方法，所得到的結果可能已遭竄改。為了解決這個問題，Hua 等

人基於 Virtual Machine Monitor (VMM)的方法提出了 VmRecoverySystem[9]。不同於傳統的安全檢測系統是在被監控的主機中運行，VmRecoverySystem 設置於被監控的主機外部，以提高準確性及抗干擾能力，確保數位證據的正確性和有效性。

而研究者 Schatz 為改善基於軟體的記憶體萃取工具會影響記憶體數據之缺點，並保留軟體工具的可用性，提出了 BodySnatcher[19]，其結合了硬體設備萃取記憶體的可靠性，透過獨立的作業系統，取得硬體設備的控制權，讓目標主機暫停，在不受時間影響的情況下取得完整的記憶體數據。然而此方法受限於只能用一個中央處理器的 Windows 2000 虛擬機驗證。

Dave 等人則認為現場鑑識是數位鑑識中最大的挑戰[4]，由於現場鑑識最重要的一環為記憶體萃取。然而記憶體具有易逝性且容易遭到破壞，因此鑑識人員須具備專業知識以達成預期成果。透過介紹適用於各種不同的作業系統之記憶體萃取工具，如 Windows 系統可以使用 FTK Imager、WinPmem。並展示萃取後的記憶體映像檔中存有電子郵件及社群軟體等相關資料，有助於鑑識人員於調查中獲取重要訊息。

Ghafarian 等人提出了一種新的記憶體鑑識架構[6]。此架構可分為三部分，首先為選擇記憶體萃取工具標準，其次為記憶體分析工具選擇標準，最後為取證步驟。透過所提出的架構，在 Firefox、IE、Chrome 和 Safari 瀏覽器的無痕瀏覽模式下使用後，探索所會留下的跡證。實驗結果表明，儘管關閉此四種瀏覽器，但仍然可以從記憶體中找到電子信箱的相關訊息。而 Ke 等人[12]則針對主流的三款瀏覽器：IE、Google Chrome 以及 Firefox 比較其無痕瀏覽模式及一般瀏覽刪除網頁紀錄的情形，並提出了在此兩種情況下萃取網頁紀錄的鑑識方法。換句話說，無痕瀏覽模式並非完全無痕。

此外，Periyadi 等人則特別透過三種網路攻擊行為說明記憶體鑑識有助於獲得事件真相[16]。藉由實際演練 Session Hijacking、FTP Attack 及 Illegal Access 此三種網路攻擊行為，爾後萃取記憶體進行分析。結果顯示，在三種攻擊行為的情境下，皆可從中找出可靠和有效的數位證據。Dija 等人[5]則提出了一種新的方法，透過分析 Windows 7 系統的記憶體中的可執行檔，以重建可執行檔。透過重建的可執行檔可以從中發現一些惡意程式的訊息，以協助鑑識人員進行調查。

參、記憶體鑑識分析方法研究

為達萬物互聯的目的，WoT 時代已悄悄來臨，間接使得 Web 應用程式的漏洞不可忽視。在不遠的將來，駭客可能會透過 IoT 設備上的 Web 應用程式漏洞取得或更改使用者的帳號密碼，攻擊流程如圖 2 所示。並將使用者的帳號密碼存入惡意程式的字典檔中，以增加惡意程式暴力破解後感染 IoT 連線設備的成功機率。

針對 WoT 的攻擊，本文認為可以將記憶體鑑識分析的方向分為兩大部分，第一部分是針對駭客之攻擊端主機進行鑑識分析，主要目的是要找出駭客確實有利用 Web 應

用程式漏洞進行攻擊行為之跡證，提出一套完整的記憶體鑑識分析方式，藉由本文所建議的方法，完成記憶體鑑識分析的工作。

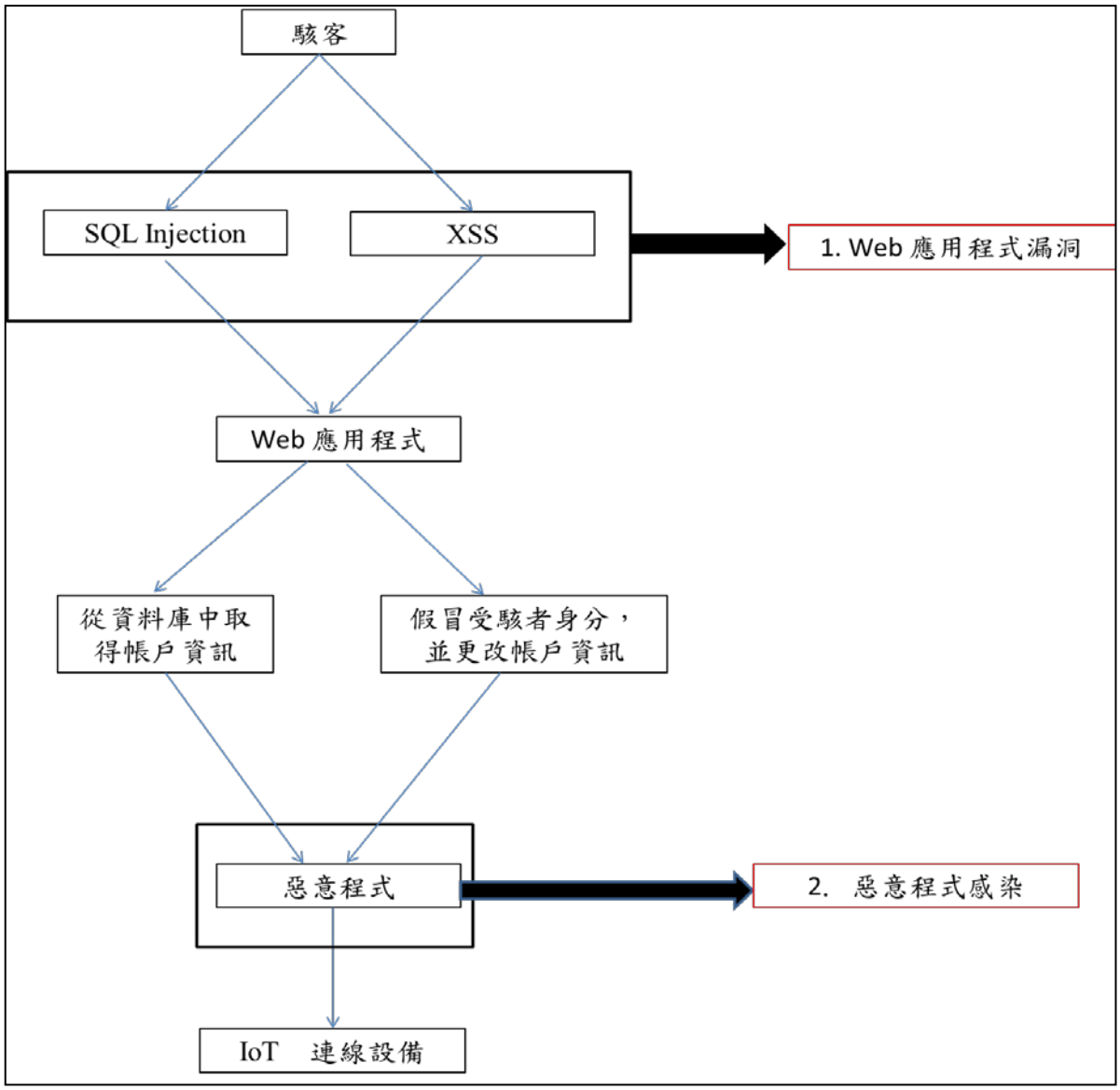

圖 2 : IoT 攻擊流程圖

3.1 Web 應用程式鑑識分析研究

對於 Web 應用程式鑑識分析，首要探討的議題為 Web 應用程式漏洞攻擊，以下說明兩種不同的 Web 應用程式漏洞攻擊。並說明如何透過記憶體鑑識技術找出攻擊端之不法行為。

3.1.1 Web 應用程式漏洞之攻擊

根據 OWASP TOP 10 2017 報告[24]，目前存在幾種主要的 Web 應用程式安全漏洞，表示雖然某些攻擊手法雖然存在已久，也並非運用最新的技術，但依然是最有效的攻擊

手段，如 SQL Injection、XSS 等在 Web 應用程式安全漏洞統計報告中時常為榜上名單。

本文透過情境一及情境二分別說明駭客如何使用 SQL Injection 及 XSS 攻擊手法取得使用者帳號密碼以達到擴散惡意程式之目的，並探討在駭客以無痕瀏覽模式作為媒介的情況下，該如何使用記憶體鑑識方式找出數位跡證。

情境一:

駭客透過 SQL Injection 攻擊方式取得網站資料庫中存放帳號密碼之表單，並添加於惡意程式的字典檔中，以增加感染 IoT 連線設備的成功率，流程圖如圖 3 所示，其詳細攻擊流程說明如下:

1. 駭客在無痕瀏覽模式下利用 SQL 語法對資料庫進行查詢。
2. 藉由資料庫所回傳之訊息，推測資料庫結構。以此方式找出記錄使用者帳號密碼的表單。
3. 駭客取得表單後，即可將此表單上之帳號密碼寫入惡意程式的字典檔。
4. 便可破解符合此字典檔帳號密碼的 IoT 連線設備。

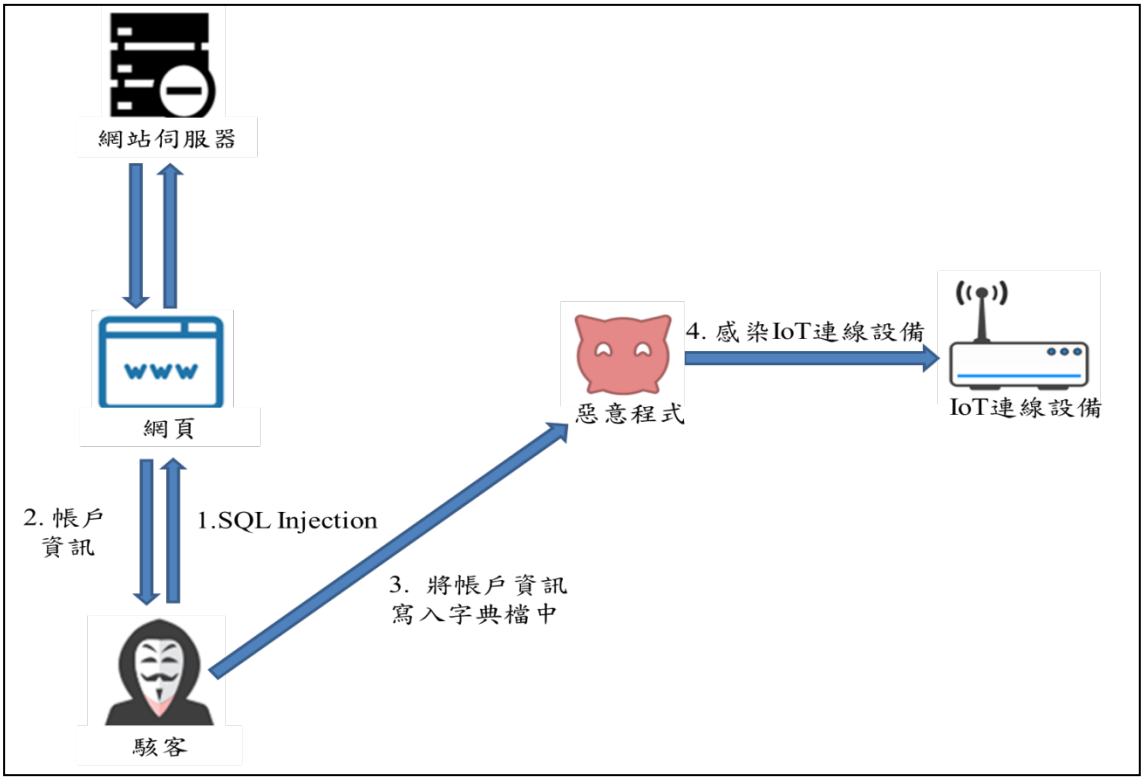

圖 3：情境一攻擊流程圖

情境二 ：

駭客可藉由 XSS 攻擊方式取得含有帳號密碼資訊的 Cookie 值，以便假冒使用者登入網站，並且更改使用者密碼，將此訊息寫入惡意程式的字典檔中，便可感染此 IoT 連

線設備，詳細攻擊情境流程如下圖 4 所示:

1. 駭客在無痕瀏覽模式下利用網頁漏洞將欲取得 Cookie 值之惡意 Script 儲存於網站伺服器中。
2. 當使用者登入此網頁後，網頁會從網站伺服器中回傳資料。
3. 由於網站伺服器已儲存惡意 Script，因此所回傳給使用者之資料即包含惡意 Script。
4. 駭客即可獲得使用者之 Cookie 值。
5. 駭客利用所得到的使用者 Cookie 值登入網站，即可冒用使用者身分進行密碼更改。
6. 駭客將帳號資訊及更改後的密碼寫入惡意程式的字典檔中。
7. 即可成功感染該 IoT 連線設備。

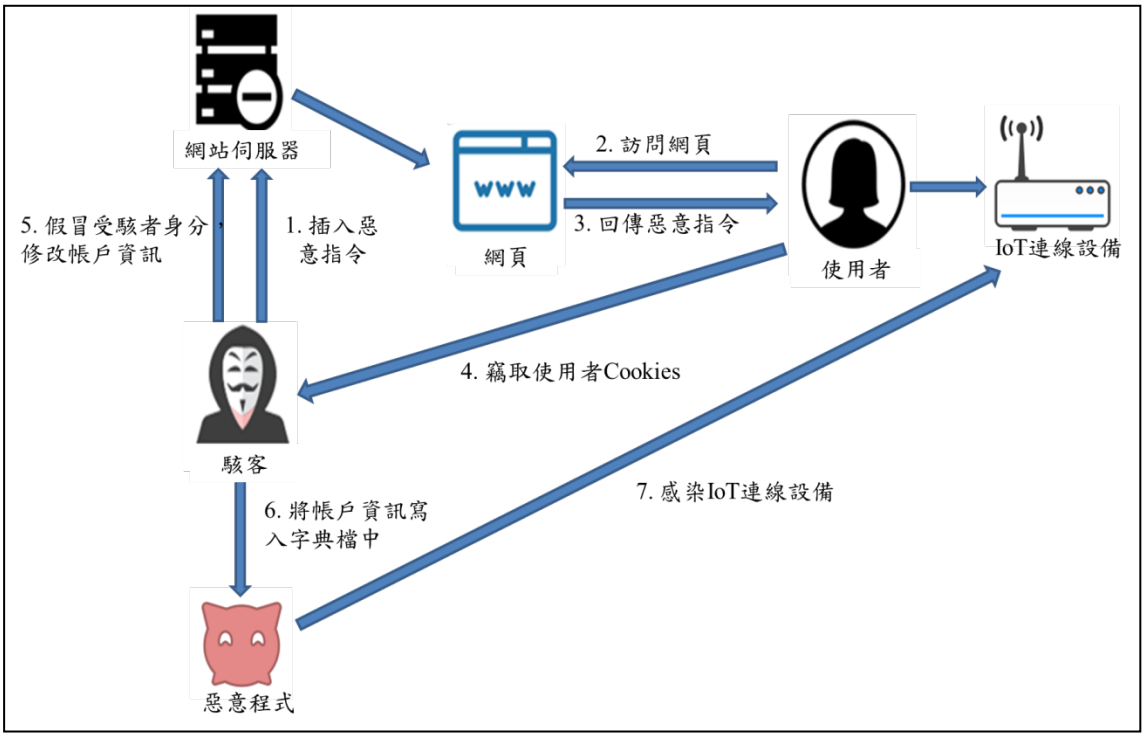

圖 4: 情境二攻擊流程圖

3.1.2 Web 應用程式漏洞之記憶體鑑識

倘若駭客使用無痕瀏覽模式進行 Web 應用程式漏洞攻擊，其所會留下的數位證據較少。使得其惡意行為不易被人發現，也讓鑑識人員在調查的過程更為艱辛，造成鑑識人員較難以完整的還原案件的全貌。

雖然瀏覽器在無痕瀏覽模式時，並不會於電腦中的快取(Cache)、Cookies 或歷史紀錄(History Record)留下任何紀錄，但執行大多數程式時，作業系統都會將資料儲存在記憶體中，使其得以順利執行。由此可推論，當開啟瀏覽器執行無痕模式瀏覽網頁時，雖然不會在電腦中儲存任何檔案，但是可能會將瀏覽網頁的紀錄、輸入之字串等資訊暫存

於揮發性記憶體中，因此，本文嘗試使用揮發性記憶體鑑識技術作為鑑識蒐證方法。而在數位鑑識的過程中，對於鑑識人員而言，除了從受駭主機中找出被攻擊的證據以外，如果能夠從攻擊主機中找到與攻擊有關的數位證據，將可使整體調查結果更加完整，使證據更有說服力。如此一來，不僅能藉由數位證據還原犯罪過程，更可以證明攻擊端確實有不法行為。

因此，本文將針對攻擊端主機進行記憶體鑑識分析，以證實其確實有進行攻擊行為，詳細流程如下：

步驟 1: 查看日誌檔

每種連網裝置都會產生系統服務的紀錄檔案，也就是俗稱的日誌檔(Log)。在不同的作業系統中，鑑識人員可以透過日誌檔，查看不同應用程式的紀錄。藉由異常行為的日誌檔紀錄，可以幫助鑑識人員縮小調查範圍，如發現有異常行為之網際網路協定(IP)位置，便可協助鑑識人員鎖定可疑的嫌疑人，進而能將調查資源著重於特定惡意紀錄進行分析，而非如大海撈針，茫然於系統中找尋惡意行為紀錄。

步驟 2: 揮發性記憶體萃取

選擇萃取揮發性記憶體工具取決於不同因素，如處理時間、使用者介面及所支援的作業系統等。本文使用 FTK Imager 且在電腦未關機的前提下進行揮發性記憶體萃取。FTK (Forensic Toolkit)是由 Access Data 公司所開發的一套數位鑑識工具。而旗下的軟體 FTK Imager 則專門用於獲取映像檔，其他功能如匯入多種類型之映像檔及進行數位證據的預覽等，且可透過圖形介面提供數位鑑識人員更便利的操作，從左上角之 File 欄位中選擇 “Capture Memory”則可進行記憶體之萃取，如圖 5 所示。

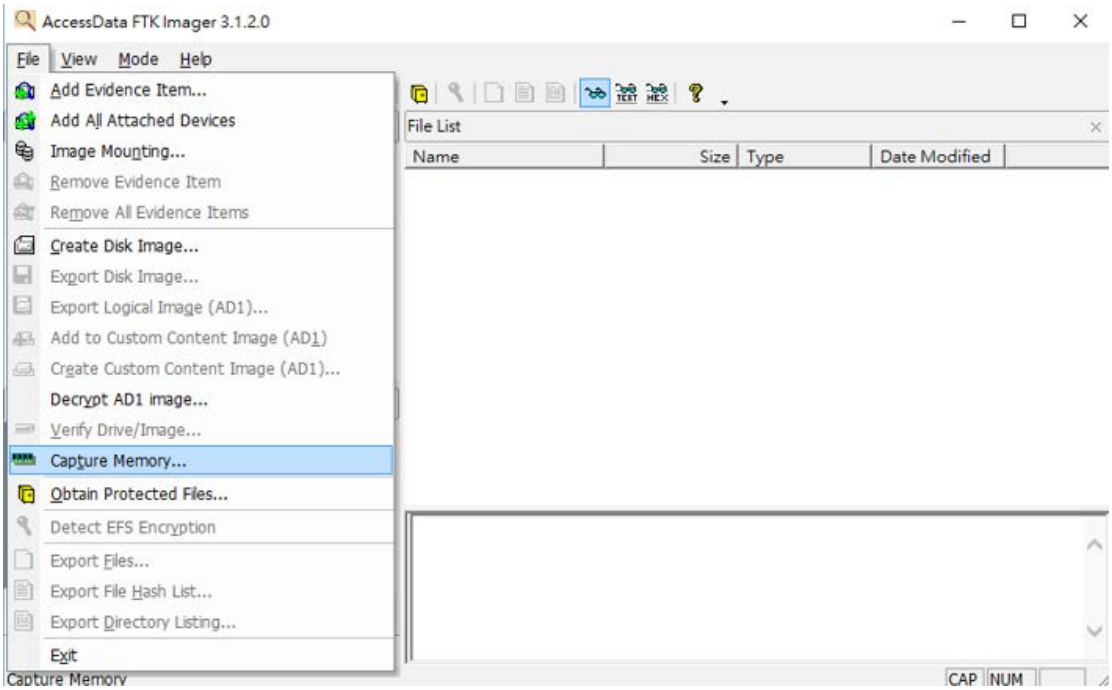

圖 5： 使用 FTK Imager 萃取記憶體

步驟 3: 揮發性記憶體分析

鑑識人員可透過查看日誌檔的方式從中發現異常行為，藉此協助鑑識人員鎖定目標，建立偵查方向，例如在記憶體分析過程中，可將相關資訊以關鍵字搜尋方式進行偵查，鑑識人員不僅可從分析結果得知嫌疑人是否有進行惡意行為，亦可藉由關鍵字之上下文的記憶體資訊，推測嫌疑人之惡意行為。本文即從日誌檔中找出攻擊行為相關資訊，在取得揮發性記憶體後，使用 WinHex 工具分析暫存於記憶體中的資訊，並從中搜尋既定的關鍵字串或關鍵檔案，證明嫌疑人確實有進行不法行為。WinHex 是由 X-Ways Software Technology AG 公司所開發的一套十六進制編輯器，可以用來復原數據並且提供數位鑑識人員進行資料的剖析。如圖 6 所示。

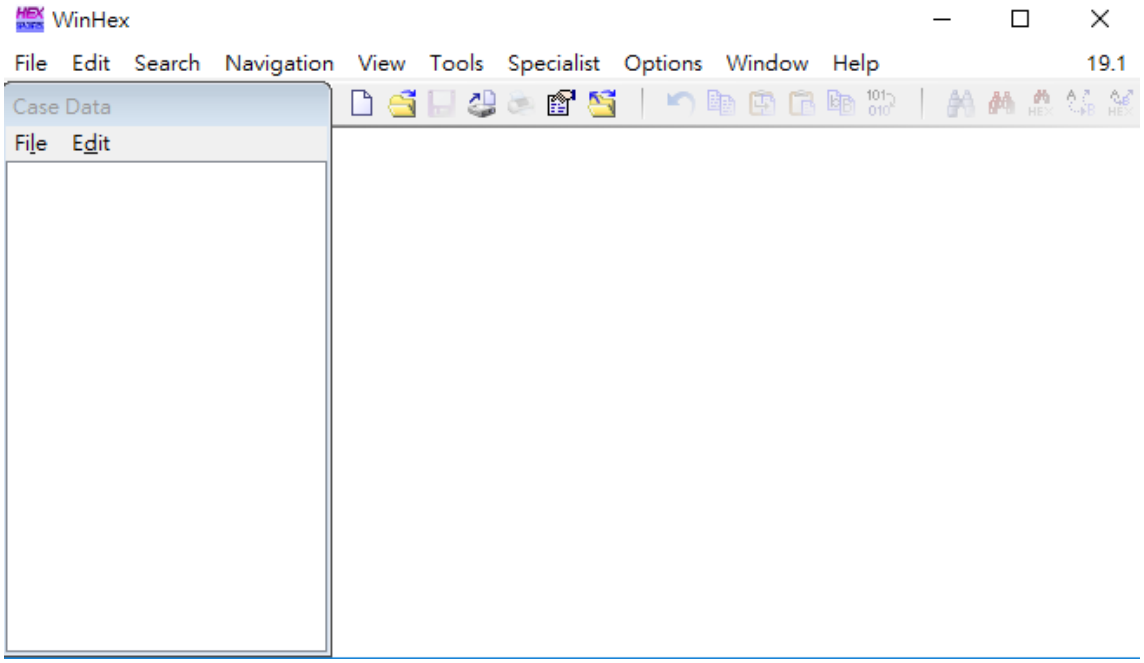

圖 6 : WinHex 畫面

肆、記憶體鑑識分析實驗

本文將記憶體鑑識分析分為兩大方向探討，一為針對攻擊端主機之記憶體鑑識分析，證明駭客確實有透過 Web 應用程式漏洞進行非法行為，從中找出其特徵值。相關實驗環境資訊如表 2、3、4 所示。

表 2 : Web 應用程式漏洞攻擊端主機環境

	SQL Injection	XSS
硬體規格	intel i7-4790/8 GB RAM	intel i7-4790/8 GB RAM
作業系統	Windows 10 (64 bits)	Windows 7 (64 bits)
瀏覽器	Chrome 67.0.3396	Chrome 67.0.3396

表 3 : Web 應用程式漏洞受駭端主機環境

	SQL Injection	XSS
虛擬機	VM Workstation 11	VM Workstation 11
受駭網站	OWASP Mutillidae II, Version 1.2	OWASP WackoPicko, Version 1.2
瀏覽器	Chrome 67.0.3396	Chrome 67.0.3396

表 4 : Web 應用程式漏洞鑑識端主機環境

	SQL Injection	XSS
硬體規格	intel i7-4790/8 GB RAM	intel i7-4790/8 GB RAM
作業系統	Windows10 (64 bits)	Windows 7 (64 bits)
FTK Imager	3.1.2.0	3.1.2.0
WinHex	19.1.0.0	19.1.0.0

4.1 Web 機制實驗方法

依據 NetMarketShare（https://netmarketshare.com）近幾年所做的調查， Google Chrome 為目前全球最多人使用的瀏覽器之一，因此本文使用 Google Chrome 的無痕瀏覽模式進行攻擊行為之實驗。

情境一攻擊方式 ：

駭客以Google Chrome無痕瀏覽模式訪問具有漏洞的網站OWASP Mutillidae II(圖 7)，並進行 SQL Injection 攻擊。

圖 7 : OWASP Mutillidae II

假設使用者的帳號為「user」且密碼為「pass」，則在正常情況下輸入帳號密碼後，SQL 敘述會執行:

“ SELECT * FROM accounts WHERE username='user' AND password='pass' ”

但當駭客輸入一段有語意的 SQL 敘述，例如:「'OR 1=1 #」，

“ SELECT * FROM accounts WHERE username='' OR 1=1 # 'AND password='' ”

如圖 8 所示，則會使得 SQL 在實際執行時會執行下列敘述(表)並成功顯示名為 accounts 表單之內容，駭客便可成功取得多個使用者的帳號密碼，如圖 9 所示。

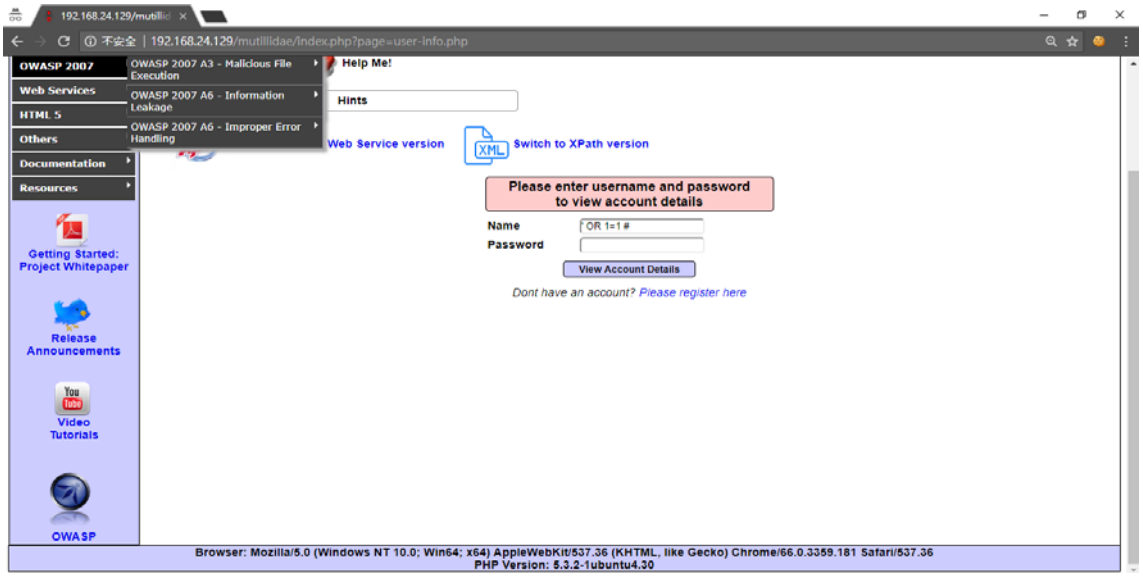

圖 8: 輸入 SQL Injection 攻擊指令

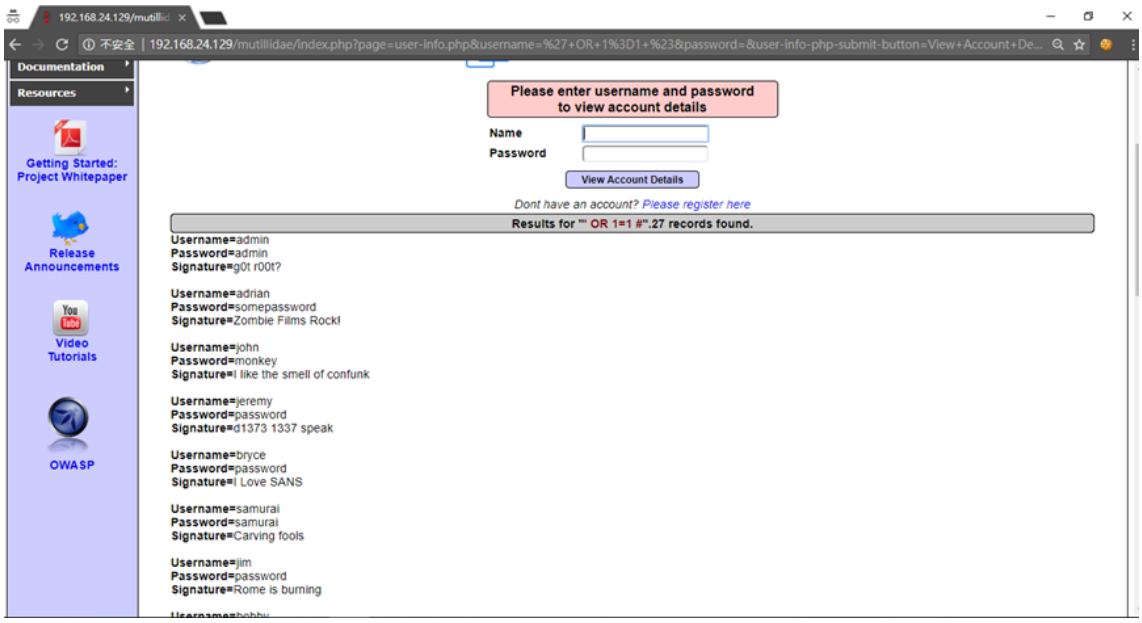

圖 9: 顯示名為 accounts 表單之內容

情境二攻擊方式：

駭客以 Google Chrome 無痕瀏覽模式訪問具有漏洞的網站 WackoPicko(圖 10)，並進行 XSS 攻擊，以下將依序逐步說明:

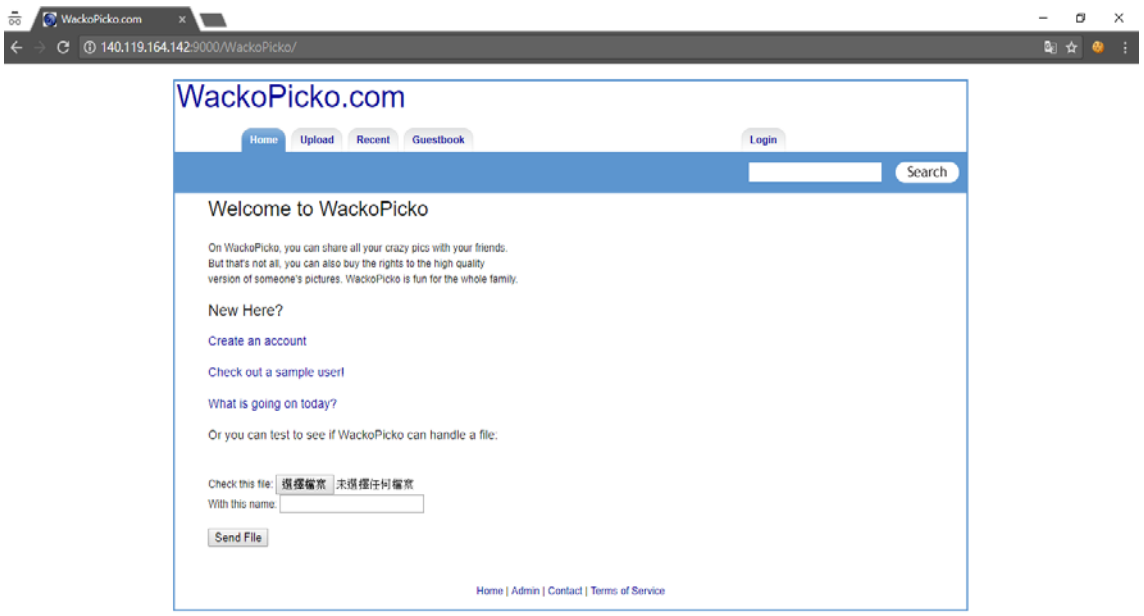

圖 10: WackoPicko

步驟 1. 駭客在 Guestbook(此網站的留言板)輸入惡意程式碼(3)，如圖 11 所示。此惡意程式碼用意為將受駭者導向 http://xxx.xxx.xxx.142/test.php，且其 output 內容為受駭者之 Cookie 值。

```
<script>
    location="http://xxx.xxx.xxx.142/test.php?output="+document.cookie;
</script>
```

圖 11： 輸入 XSS 攻擊指令

步驟 2. 當駭客成功將惡意語法注入網站後，即可等待受駭者瀏覽 Guestbook，以觸發駭客自製的網頁 http://xxx.xxx.xxx.142/test.php (圖 12)並取得受駭者 Cookie 值。

圖 12： 駭客監聽受駭網站

步驟 3. 當受駭者登入網站後瀏覽 Guestbook 頁面時，便會自動執行駭客設計的網頁，使駭客順利取得 Cookies (圖 13)。

圖 13: 取得受駭者的 Cookies

步驟 4. 當駭客取得受駭者 Cookies 後，可藉由 Google 套件 EditThisCookie，置換 Cookies(圖 14)，即可假冒受駭者身分登入網站(圖 15)，亦可於該網站中以受駭者的身分上傳惡意程式或更改其密碼等行為。

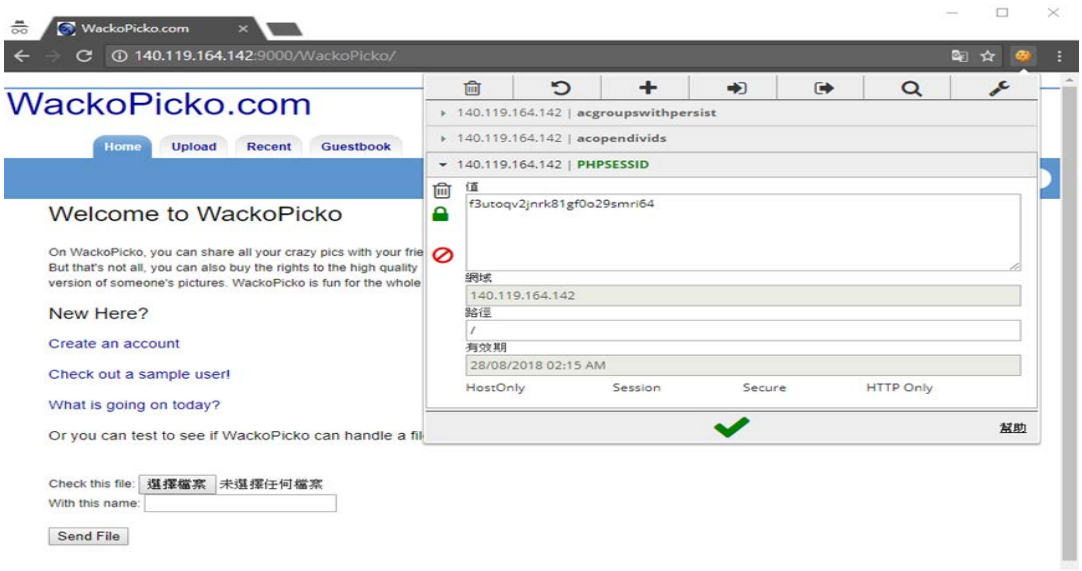

圖 14: 置換 Cookies

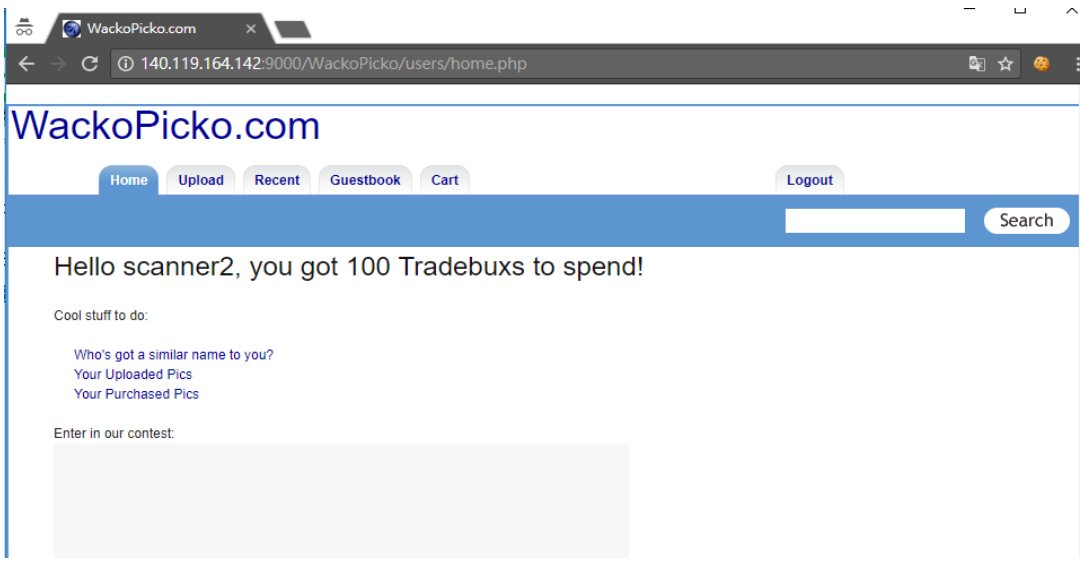

圖 15： 假冒受駭者身分登入網站

4.2 Web 機制討論分析

本文以常見的 Web 應用程式漏洞 SQL Injection 及 XSS 為例，說明當駭客以無痕瀏覽模式進行不法行為時，鑑識人員該如何從記憶體中找出跡證。

A. 情境一分析流程

步驟 1. 鑑識人員先從網站伺服器中查詢連線紀錄，如圖 16 所示，發現 IP 192.168.24.1 在 2018 年 5 月 31 日 2 時 53 分時，有大量連線紀錄，且其時間間隔相近，所以懷疑網站於此時遭受駭客攻擊。

```
192.168.24.1 - - [31/May/2018:02:53:39 -0400] "GET /mutillidae/index.php?page=us
er-info.php&username=%27+OR+1%3D1+%23&password=&user-info-php-submit-button=View
+Account+Details HTTP/1.1" 200 9371 "http://192.168.24.129/mutillidae/index.php?
page=user-info.php&username=%27+OR+1%3D1+%23&password=&user-info-php-submit-butt
on=View+Account+Details" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/
537.36 (KHTML, like Gecko) Chrome/66.0.3359.181 Safari/537.36"
```

圖 16: 可疑 IP 位置連線紀錄

步驟 2. 在進一步調查之下，發現網站伺服器可能遭到 SQL Injection 攻擊，於 mysql.log 日誌檔內搜尋其連線資訊，發現有遭受 SQL Injection 攻擊之指令字串，如圖 17 所示。

```
                635 Query     INSERT INTO hitlog(hostname, ip, browser, refere
r, date) VALUES ('192.168.24.1', '192.168.24.1', 'Mozilla/5.0 (Windows NT 10.0;
Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/66.0.3359.181 Safari/5
37.36', 'Recieved request to display user information for: \' OR 1=1 #',  now()
)
                637 Query     SELECT * FROM accounts
                      WHERE username='' OR 1=1 #' AND password=''
```

圖 17: SQL Injection 攻擊之指令字串

步驟 3. 由上述鑑識調查結果，調查人員已可知駭客的 IP 位址為 192.168.24.1，經過一連串的 IP 追查後，發現該 IP 的使用者為 Bob，於是調查人員會同刑警鎖定 Bob 後，經由合法程序進入 Bob 家，發現正在運作的電腦一台，同時發現 B 君有使用無痕模式瀏覽網頁的習慣，鑑識人員為避免揮發性證據逸失，立即透過 FTK Imager 進行記憶體的萃取，如圖 18 所示。

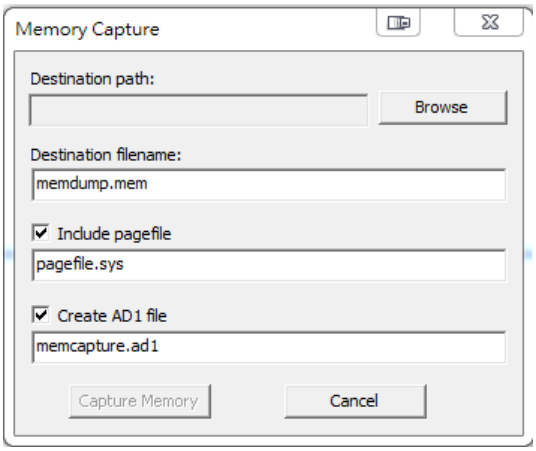

圖 18: 使用 FTK Imager 進行記憶體萃取

步驟 4. 鑑識人員利用 FTK Imager 取得記憶體的映像檔後，以 WinHex 工具搜尋記憶體的內容，查看是否有 Bob 電腦連線到該網站伺服器的紀錄，故以該網站網址為關鍵字進行搜尋，如圖 19 所示，而其搜尋結果，確實發現 Bob 曾經連線瀏覽該網頁，如圖 20 所示。

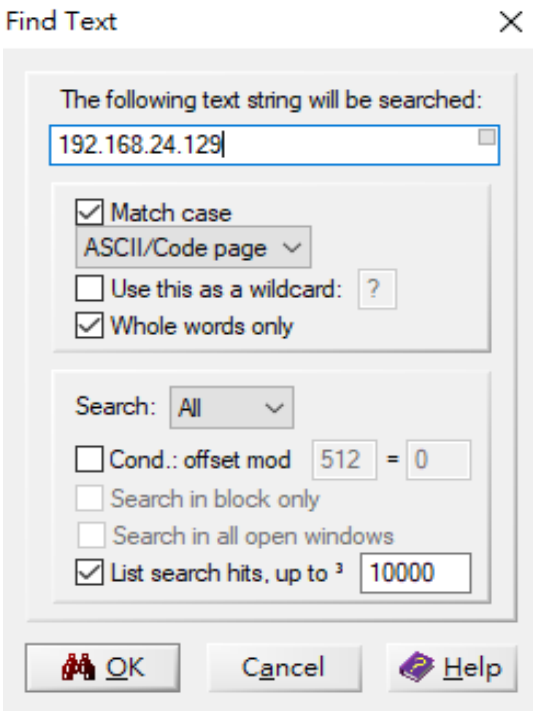

圖 19: 以網址「192.168.27.129」為關鍵字搜尋

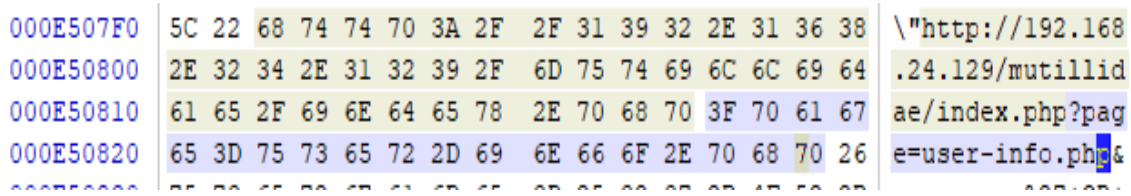

圖 20: 找出連線到該網站的紀錄

步驟 5. 因為已經搜尋到連線資訊，於是鑑識人員進一步搜尋是否有輸入 SQL Injection 語法之紀錄，並利用已知的攻擊語法，嘗試進行關鍵字搜尋，如圖 21 所示。而且所得鑑識結果，發現 Bob 不僅曾經連線至該網站，並涉嫌以 SQL Injection 手法發動攻擊（如圖 22）。

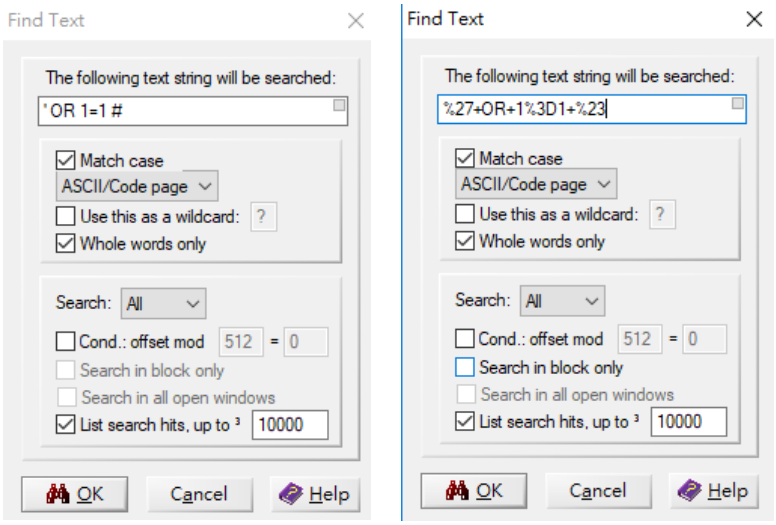

圖 21: 以「' OR 1=1 #」及轉換編碼方式進行關鍵字搜尋

```
00D272760  68 74 74 70 3A 2F 2F 31  39 32 2E 31 36 38 2E 32  http://192.168.2
00D272770  34 2E 31 32 39 2F 6D 75  74 69 6C 6C 69 64 61 65  4.129/mutillidae
00D272780  2F 69 6E 64 65 78 2E 70  68 70 3F 70 61 67 65 3D  /index.php?page=
00D272790  75 73 65 72 2D 69 6E 66  6F 2E 70 68 70 26 75 73  user-info.php&us
00D2727A0  65 72 6E 61 6D 65 3D 25  32 37 2B 4F 52 2B 31 25  ername=%27+OR+1%
00D2727B0  33 44 31 2B 25 32 33 26  70 61 73 73 77 6F 72 64  3D1+%23&password
00D2727C0  3D 26 75 73 65 72 2D 69  6E 66 6F 2D 70 68 70 2D  =&user-info-php-
00D2727D0  73 75 62 6D 69 74 2D 62  75 74 74 6F 6E 3D 56 69  submit-button=Vi
00D2727E0  65 77 2B 41 63 63 6F 75  6E 74 2B 44 65 74 61 69  ew+Account+Detai
00D2727F0  6C 73 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ls
```

圖 22: 疑似 SQL Injection 語法之字串

B. 情境二分析流程

步驟 1. 鑑識人員於網站伺服器中查詢連線紀錄，發現 IP 140.xxx.xxx.34 在 2017 年 8 月 28 日 2 時 13 分時，有連線紀錄，如圖 23 所示，且使用者以 scanner2 身分登入網站，其 id=5，如圖 24 所示。除此之外，亦發現在短時間內，有另一個 IP 140.xxx.xxx.142 以不同的作業系統卻以同樣身分登入網站，如圖 25 所示，這引起鑑識人員的關注。

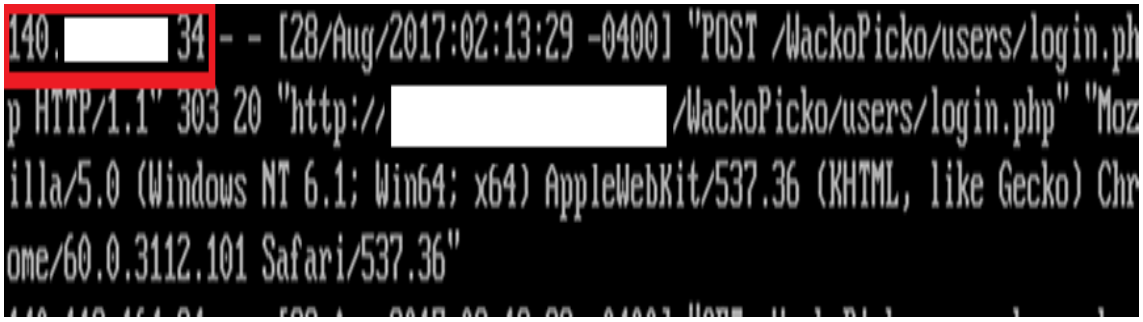

圖 23: IP 140.xxx.xxx.34 連線紀錄

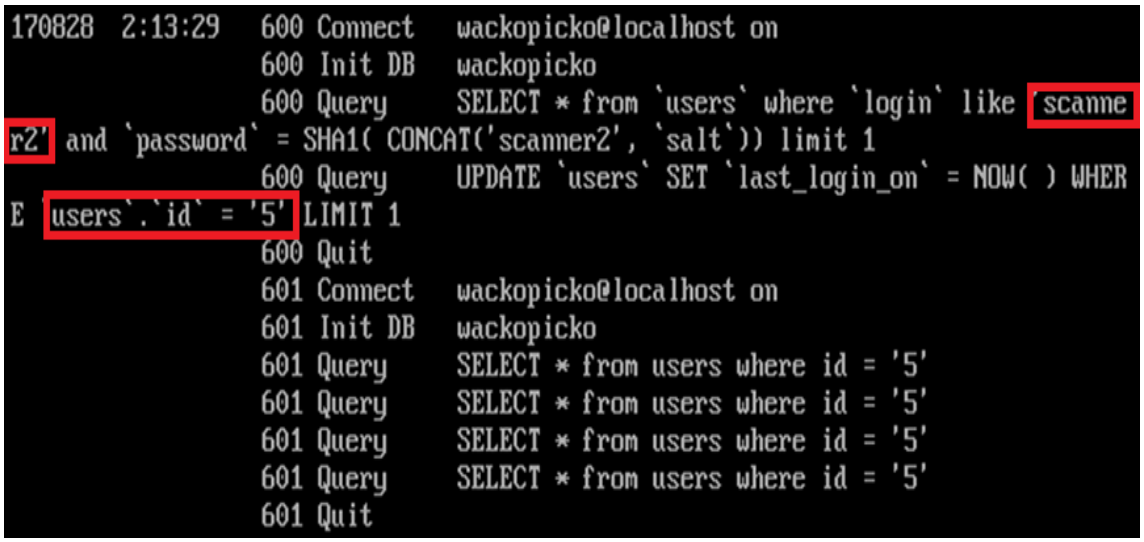

圖 24: 以 scanner2 身分登入網站，其 id=5

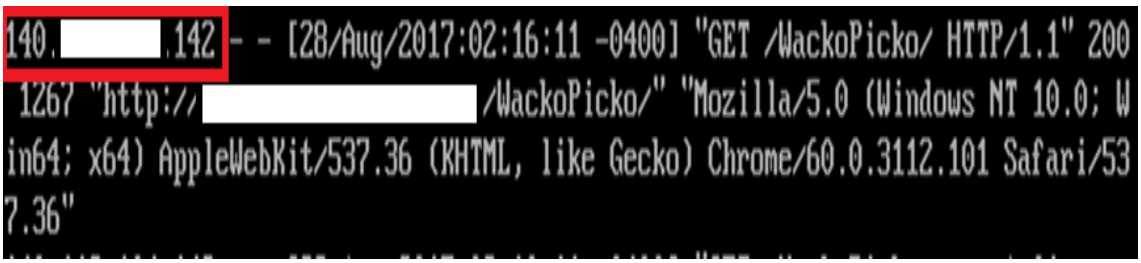

圖 25: IP 140.xxx.xxx.142 連線紀錄

步驟 2. 在鑑識人員調查下，發現可能是網站使用者的身分遭到冒用，於是進一步在 mysql.log 日誌檔內搜尋紀錄，發現在使用者登入前，網站的留言板(guestbook)被嵌入了 XSS 攻擊語法，如圖 26 所示。且使用者也確實有查看 guestbook 網頁，如圖 27 所示，因此鑑識人員推測，使用者可能觸發了 XSS 攻擊。而根據圖 28 所示，可知攻擊手法除了將使用者導向另一個網頁，同時也取得了使用者的 cookie 值，由此可推斷使用者的身分可能遭到冒用。

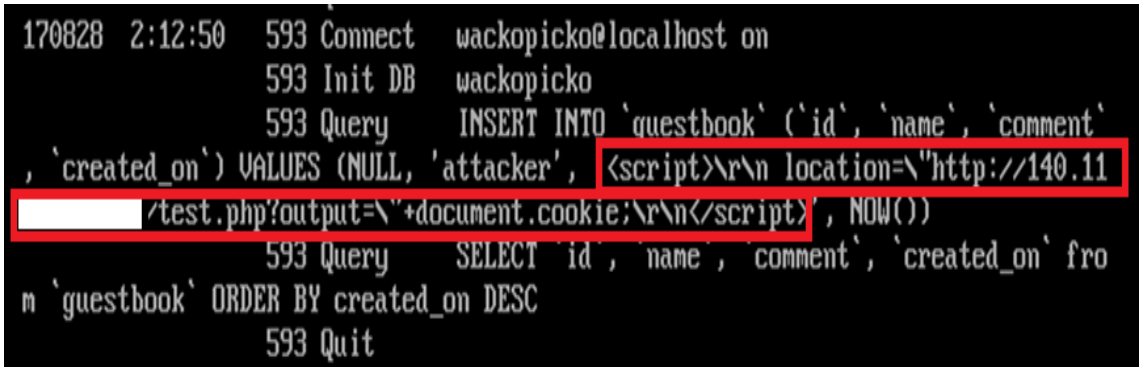

圖 26: XSS 攻擊之指令

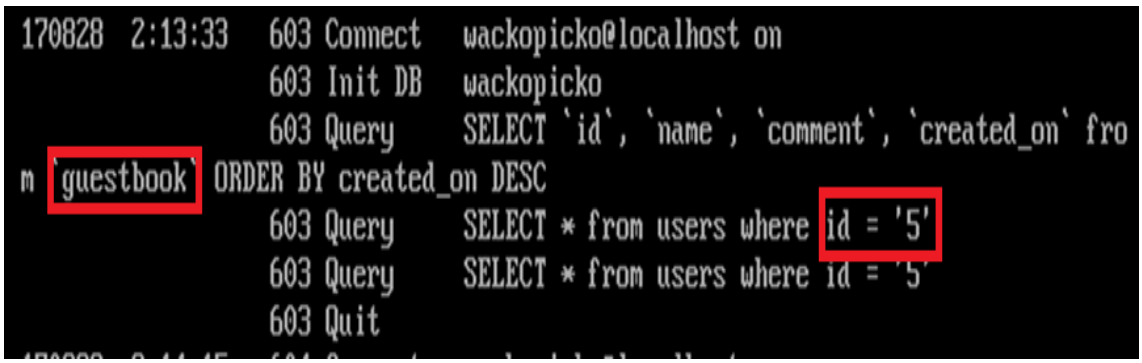

圖 27: 使用者點擊 guestbook 頁面

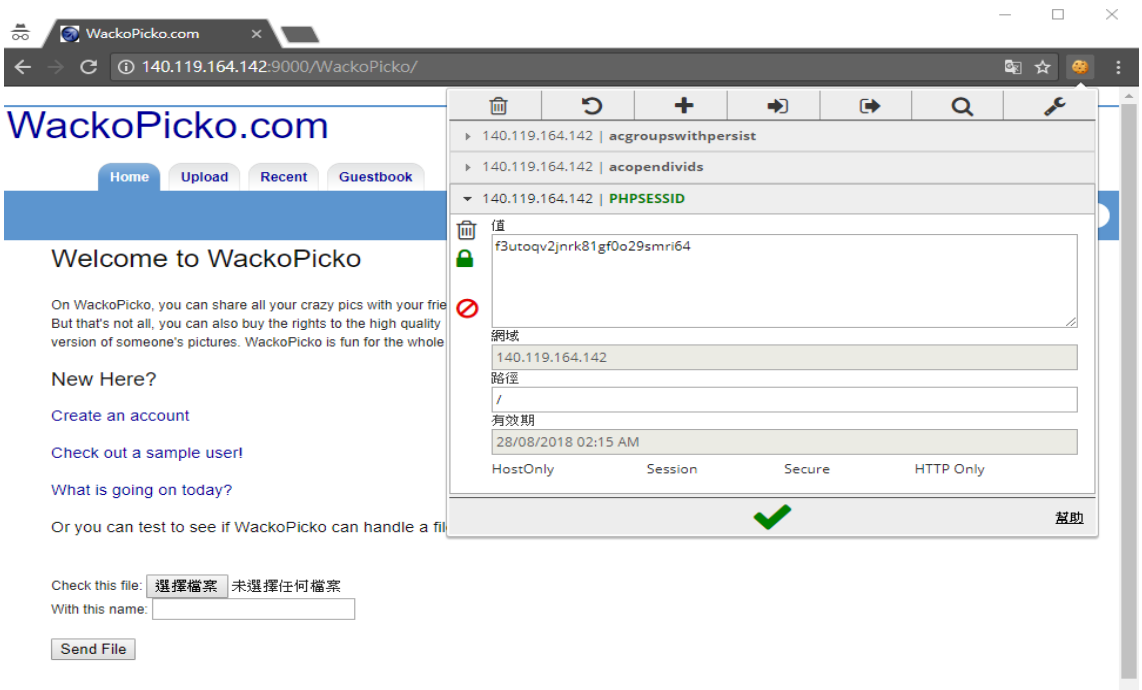

圖 28: 編輯 Cookies 以冒用使用者身分

步驟 3. 根據上述鑑識調查結果，合理的推斷 IP 140.xxx.xxx.142 為可疑目標，經過一連

串的 IP 追查後，發現該 IP 的使用者為 Alice，於是調查人員鎖定 Alice 後，經由合法程序進入 Alice 住所並發現一台正在運行的電腦，因此立即透過 FTK Imager 進行採證，避免失去揮發性證據，如圖 29 所示。

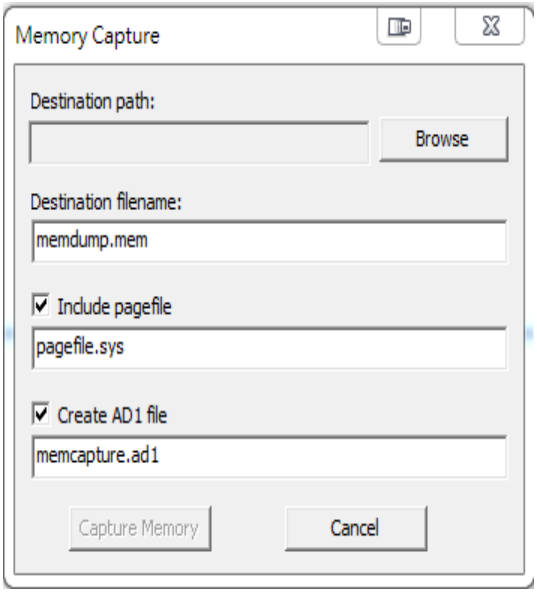

圖 29: 使用 FTK Imager 進行記憶體萃取

步驟 4. 鑑識人員利用 FTK Imager 取得記憶體的映像檔後，以 WinHex 工具搜尋是否有該網站的網址，如圖 30 所示，而其搜尋結果，確實發現 Alice 曾經連線瀏覽該網頁，如圖 31 所示。

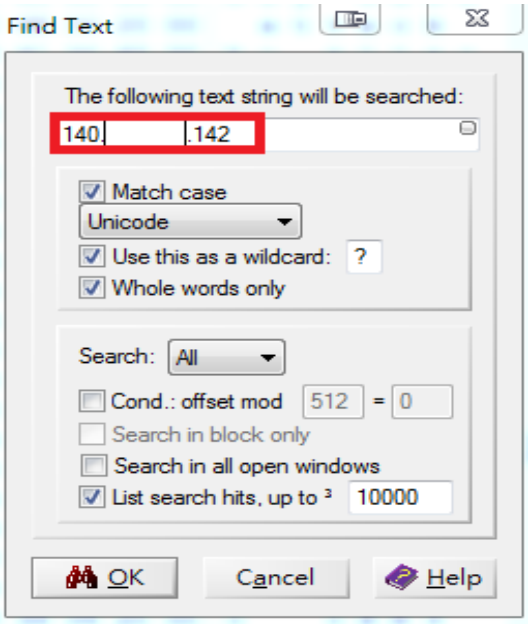

圖 30: 以「140.xxx.xxx.142」為關鍵字搜尋

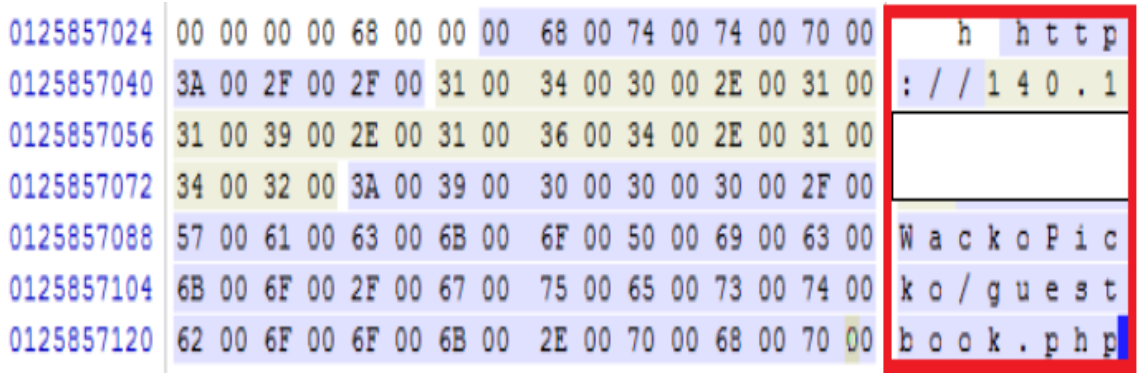

圖 31: 找出連線到該網站的紀錄

步驟 5. 因為已經搜尋到連線資訊，於是鑑識人員進一步搜尋是否有輸入 XSS 語法之紀錄，並利用已知的攻擊語法，進行關鍵字搜尋，如圖 32 所示。透過所得鑑識結果，確實發現 Alice 不僅曾經連線至該網站，並涉嫌發動 XSS 攻擊（如圖 33）。

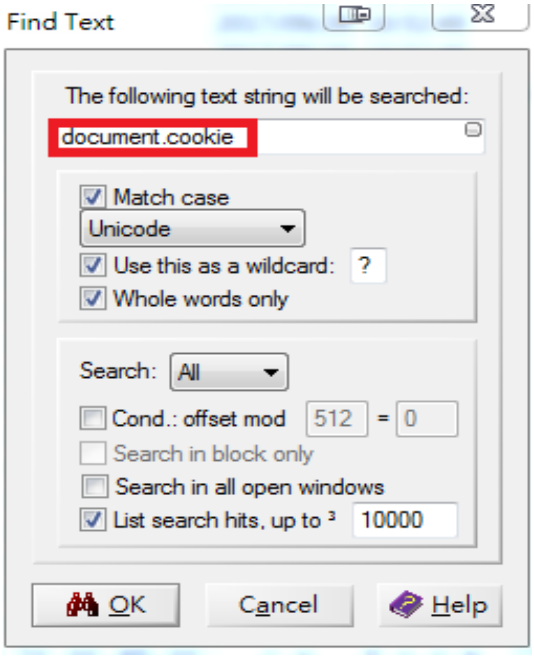

圖 32: 以「document.cookie」為關鍵字搜尋

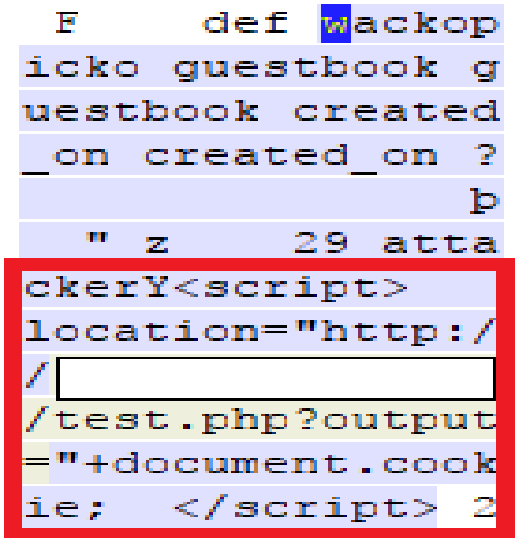

圖 33: 疑似 XSS 語法之字串

步驟 6. 由於已知的 XSS 攻擊語法包含有抓取 cookie 值，因此鑑識人員進一步搜尋，發現 Alice 有從 guestbook 這個被攻擊的頁面抓取使用者的 cookie，如圖 34 所示，因此可確認 Alice 確實涉嫌發動 XSS 攻擊並偽裝其他使用者上線。

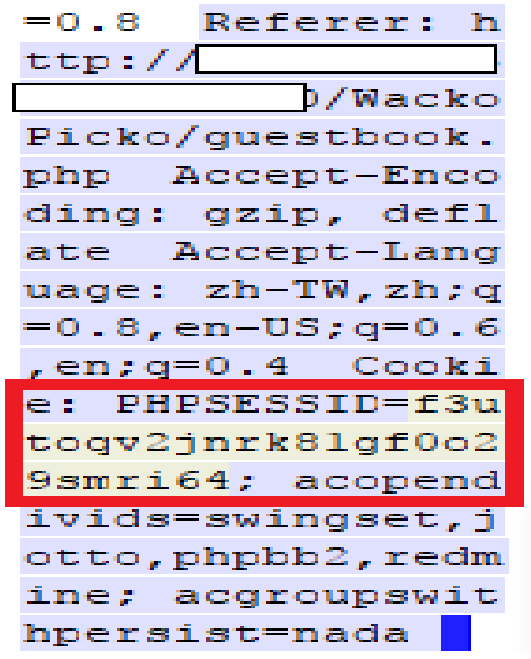

圖 34: 攻擊者在 guestbook 頁面抓取使用者的 cookie

4.3 研究比較與研究貢獻

本文提出在 WoT 時代下可能會發生的攻擊模式，即駭客可能透過 Web 應用程式漏洞從網站資料庫中取得受駭者之帳號密碼，並寫入惡意程式的字典檔當中，以成功感染

IoT 連線設備。因此本文透過記憶體鑑識之技術針對 Web 應用程式漏洞攻擊進行探討，相關研究文獻整比較如表 5 所示。

表 5: 研究文獻比較表

比較項目	Ghafarian 等,2015[6]	Periyadi 等,2017[16]	我們的方法
主要機制	主要針對隱私瀏覽模式提出記憶體取證框架，協助鑑識人員有效的萃取與分析隱私瀏覽相關的訊息。記憶體取證框架由記憶體萃取工具標準、記憶體分析工具選擇標準及取證步驟此三部分組成。最後透過 4 款常用之瀏覽器之隱私瀏覽模式進行實驗。	主要針對受駭端主機進行記憶體鑑識調查。透過三種網路攻擊手法模擬攻擊情境，並針對受駭端主機記憶體中所殘留的跡證進行分析。同時詳細說明攻擊流程以及分析流程。結果闡明，透過記憶體鑑識調查可以有效地找出攻擊跡證。	對 WoT 時代可能會面臨到的安全問題之一:Web 應用程式漏洞進行探討。針對常見之 Web 應用程式漏洞攻擊，並在駭客以無痕瀏覽模式下進行攻擊行為的條件下，藉由記憶體鑑識技術透過受駭端主機進行記憶體鑑識調查，並從跡證中找出與攻擊端之關聯性。
我們的方法與其他機制之綜合比較	1. 除了針對受駭端主機之記憶體中所殘留的跡證進行探討外，更藉此找出與攻擊端之間的連結，以協助鑑識人員鎖定可疑嫌疑人。 2. 由於 WoT 時代是未來的趨勢，因此針對 WoT 時代所會面臨的安全議題進行探討。並提出 WoT 時代，駭客可能會透過 Web 應用程式漏洞進行攻擊，透過實驗說明記憶體鑑識技術可協助鑑識人員釐清案情。		

伍、結論

IoT 能讓許多設備都透過各種不同的網路建立連結，進而構成物聯網，而 WoT 則是讓 IoT 設備透過現有的 Web 標準互相溝通，這也導致出現常見 Web 應用程式漏洞。然而多數人皆會忽略 IoT 連線設備之安全性，如未更改預設帳號密碼等，也因此需要大量殭屍電腦的攻擊者自然而然地將具有龐大數量的 IoT 連線設備做為攻擊目標。本文針對 WoT 時代所會面臨的安全議題，即 Web 應用程式漏洞所會帶來的威脅，提出一套完整的記憶體鑑識機制。首先針對 Web 應用程式漏洞的部分，透過 2 種常見之 Web 應用程式漏洞攻擊手法：SQL Injection 及 XSS 進行實驗，探討於記憶體中所會留下的跡證。結果表明，透過本文所提出的記憶體鑑識機制，在 Web 應用程式漏洞攻擊的部分，可於記憶體中找出跡證，並可從中推測出可疑的嫌疑人，以協助鑑識人員鎖定目標。

Acknowledgment

This research was partially supported by the Ministry of Science and Technology of the Republic of China under the Grant MOST 107-2221-E-015-001-MY2-.

參考文獻

[1] Ahmed W. and Aslam B., "A comparison of windows physical memory acquisition tools", *IEEE Military Communications Conference (MILCOM)*, 2015: pp. 1292-1297.

[2] Balasundaram I. and Ramaraj E., "An Authentication Scheme for Preventing SQL Injection Attack Using Hybrid Encryption", *European Journal of Scientific Research*, vol. 53, no. 3, 2011: pp. 359-368.

[3] Chivers H., "Private browsing: A window of forensic opportunity", *Digital Investigation*, Volume 11, Issue 1, March 2014: pp. 20-29.

[4] Dave R., Mistry N. and Dahiya M.S., "Volatile Memory Based Forensic Artifacts & Analysis", *International Journal for Research in Applied Science and Engineering Technology (IJRASET)*, vol. 2, no. 1, 2014: pp. 120-124.

[5] Dija S., Suma G.S., Gonsalvez D.D. and Pillai A.T., "Forensic reconstruction of executables from Windows 7 physical memory", *IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)*, 2016: pp. 1-5.

[6] Ghafarian A. and Seno S.A.H., "Analysis of Privacy of Private Browsing Mode through Memory Forensics", *International Journal of Computer Applications*, vol. 132, no. 1, 2015: pp. 27-34.

[7] Hausknecht K., Foit D. and Burić J., "RAM data significance in digital forensics," *International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, 2015: pp. 1372-1375.

[8] Heriyanto A., Valli C. and Hannay P., "Comparison of Live Response, Linux Memory Extractor (LiME) and Mem Tool for Acquiring Android's Volatile Memory in the Malware Incident", *Australian Digital Forensics Conference*, 2015: pp. 5-14.

[9] Hua Q. and Zhang Y., "Detecting Malware and Rootkit via Memory Forensics", *International Conference on Computer Science and Mechanical Automation (CSMA)*, 2015: pp. 92-96.

[10] Johari R. and Gupta N., "Insecure Query Processing in the Delay/Fault Tolerant Mobile Sensor Network (DFT-MSN) and Mobile Peer to Peer Network", *International Conference on Network Security and Applications*, 2011: pp. 453-462.

[11] Kaur D. and Kaur P., "Empirical Analysis of Web Attack", *Procedia Computer Science*, vol. 78, no. 1, 2016: pp. 298-306.

[12] Ke B.S., Lin J.S., Wang S.J., and Tso H.K., "Private Browsing Evidence of Google History

Investigations in Computer Forensics", *Journal of e-Business*, vol. 16, no. 1, 2014: pp. 85-106.

[13] Liming C., Jing S. and Wei Q., "Study on Forensic Analysis of Physical Memory", *International Symposium on Computer,Communication, Control and Automation (3CA)*, 2013: pp. 221-224.

[14] Moh M., Pininti S., Doddapaneni S., and Moh T.S., "Detecting Web Attacks Using Multi-Stage Log Analysis", *IEEE International Conference on Advanced Computing (IACC)*, 2016: pp. 733-738.

[15] Montasari R. and Peltola P., "Computer Forensic Analysis of Private Browsing Modes", In: Global security, safety and sustainability: tomorrow's challenges of cyber security. Springer, 2015: pp. 96–109.

[16] Periyadi G. A., Mutiara and Wijaya R., "Digital forensics random access memory using live technique based on network attacked", *International Conference on Information and Communication Technology (ICoIC7)*, 2017: pp. 1-6.

[17] Petroni N.L., Walters A., Fraser T. and Arbaugh W.A., "FATKit: A framework for the extraction and analysis of digital forensic data from volatile system memory", *Digital Investigation*, vol. 3, no. 4, 2006: pp. 197-210.

[18] Reed A., Scanlon M., "Forensic Analysis of Epic Privacy Browser on Windows Operating Systems", July 2017, [online] Available: http://cyberforensicator.com/wp-content/uploads/2017/07IEpicPrivacyBrowser.l.compressed.pdf.

[19] Schatz B., "BodySnatcher: towards reliable volatile memory acquisition by software", *Digital Investigation*, vol.4, no.1, 2007, pp. S126 -S134.

[20] Seo J., Lee S. and Shon T., "A study on memory dump analysis based on digital forensic tools", *Peer-to-Peer Networking and Applications*, vol. 8, no. 4, 2015: pp. 694-703.

[21] Sharma C. and Jain S. C., "Analysis and Classification of SQL Injection Vulnerabilities and Attacks on Web Applications", *International Conference on Advances in Engineering & Technology Research (ICAETR)*, 2014: pp. 1-6.

[22] Suteva N., and Mileva A., "Computer Forensic Analysis of Some Web Attack", *World Congress on Internet Security (WorldCIS)*, 2014: pp. 42-47.

[23] Tsalis N., Mylonas A., Gritzalis D. and Katos V., "Exploring the protection of private browsing in desktop browsers", *Computers & Security*, Volume 67, June 2017: pp. 181-197.

[24] Open Web Application Security Project, "OWASP Top Ten Project", Retrieved March 1, 2017 from http://www.owasp.org/index.php/Category: OWASP Top Ten Project.

[25] WhiteHat Security, "12th Annual Application Security Statistics Report", Retrieved July 11, 2019 from https://info.whitehatsec.com/.