APT 逆向與正向工程技術

廖峰澤 林盈達
國立交通大學資訊工程系
300 新竹市大學路 1001 號
E-MAIL：frank0124.cs02g@nctu.edu.tw, ydlin@cs.nctu.edu.tw
9-30-2013

摘要

近年來由 APT 攻擊的興起，出現了利用文件作為惡意攻擊的手法，早期覺得文件不可能是病毒的想法已被推翻。本文利用逆向工程及正向工程並專對 RTF 檔進行研究，一方面，發現 RTF 檔中 pFragments、Themedata、Datastore、Objdata、Outside structure 皆可能藏有 shellcode，並發現利用 CVE-2010-3333 進行攻擊的惡意文件最常嵌入 shellcode 到 pFragments，而利用 CVE-2012-0158 進行攻擊的則最常嵌入 shellcode 到 Objdata 及 Outside structure，但是由於並非所有樣本皆嵌入 shellcode 到同樣區塊，這說明了惡意文件的多變性。另一方面，藉由切割並嵌入惡意文件的 shellcode 到正常文件的技術，讓我們對於分析惡意樣本能夠進行更深入的研究及應用。

關鍵字：APT，RTF，Shellcode，惡意文件，架構分析

1. 簡介

在資訊越來越普遍的現在，出現了很多的便利服務，如：電子商務、網路銀行等，而也同時產生了如木馬病毒這種不懷好意的惡意程式，為的就是去竊取個人帳密甚至是網路銀行帳密，故如何安全地在電腦上使用便利服務而不被惡意攻擊就成了我們想達成的目標，不過近幾年，新型的攻擊崛起，我們稱為 APT[1] (Advanced Persistent Threat)，APT 攻擊手法複雜且利用人們之間的信任及人性的弱點，以及更難以偵測的病毒，導致 APT 成為近幾年來資安的一大威脅。

APT 是近年來興起的一種攻擊方式，不像以往不懷好意的人只是散佈惡意病毒於網路，在 APT 的攻擊中有著特定的目標，可能是名人、團隊或企業，在攻擊前會利用社交工程對目標及周遭人進行調查，並利用信件夾帶惡意文件的方式攻擊受害者，受害者不一定就是目標本身，可能是目標的公司同事或生活中的家人朋友，等攻擊成功後再進一步利用受害電腦進行滲透直到任務達成，任務如竊取客戶資料或企業內的機密文件[2]，圖 1 為 APT 攻擊的流程圖，圖中說明了 APT 的攻擊步驟。

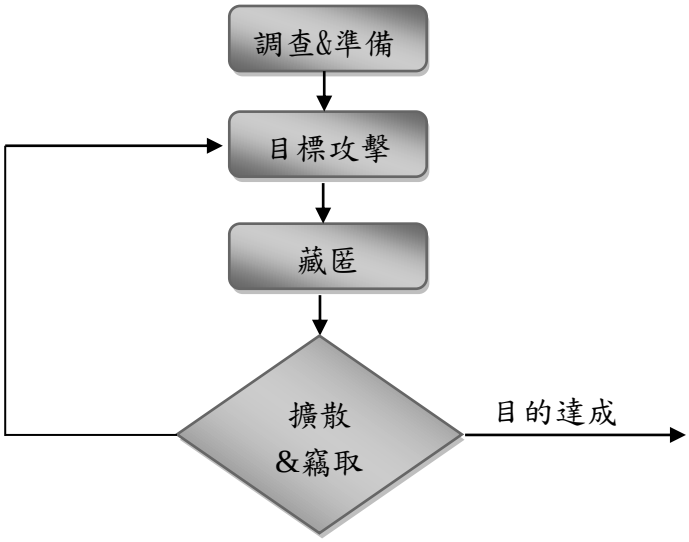

圖 1　APT 攻擊流程圖

由表 1 我們可知，傳統攻擊往往注重亂槍打鳥的方式以竊取帳密或進一步竊取金錢，攻擊對象注重於個人；而 APT 將目標轉向企業，以有計畫、複雜的手法攻入企業，為的是更龐大的金錢、個資以及更寶貴的技術。在病毒的使用上，因為只對單一目標進行攻擊，製作一個大部分防毒軟體皆偵測不到的病毒可大大減少被發現的機率，而病毒並進一步嵌入至文件中成為惡意文件，由於大部分人皆認為文件不太可能成為病毒加上電子郵件使用頻繁，故使用惡意文件作為攻擊的手法。

表 1　APT 與傳統攻擊之比較

		APT	Traditional attack
Fixed target		Yes	No
Purpose	Personal account password	Yes	Yes
	Secret data	Yes	No
	Personal information	Yes	No
Virus	Malware document	Yes	No
	Zero-day malware	Yes	No

本文將研究如何將惡意程式嵌入於文件中作為主要方向，並利用兩種角度進行研究。一方面分析並切割現有的惡意文件架構以了解可能嵌入惡意程式的位置與方法，我們稱此方面為逆向工程，另一方面我們利用逆向工程所得到樣本中的惡意區塊，並進一步嵌入到正常文件中，若正常文件擁有惡意文件的區塊，則就等同於我們將正常文件做成惡意文件，此方面稱為正向方程。

藉由逆向及正向工程，我們希望可以因此更加了解惡意文件的組成。

2. 逆向工程：惡意文件樣本分析

2.1 樣本分析

樣本取得方式來自 Xecure Lab，共得樣本 392 個（CVE-2010-3333[3]樣本 92 個、CVE-2012-0158[4] 樣本 300 個），皆為使用漏洞攻擊的樣本，由於 CVE(Common Vulnerabilities and Exposures)列舉已知的安全漏洞，故由此可知樣本進行攻擊所利用的軟體弱點，其中，兩種樣本皆屬於 RTF 檔（RTF 檔是微軟所開發的一種文字格式，主要目的是作為跨平台文件，可在多種文書軟體中檢視[5] ，在 Microsoft Office 中以 Word 作為預設檢視工具）且這兩種樣本皆可對 Office2003 及 Office2007 作攻擊，表 2 指出兩種樣本所針對的軟體漏洞，其中 CVE-2010-3333 藉由刻意安排的 RTF 檔產生 buffer 溢位進而允許遠端執行任意程式，而 CVE-2012-0158 則藉由 MSCOMCTL.OCX 中的特定 ActiveX 控制以允許遠端執行任意程式。

表 2 樣本所用漏洞比較

	CVE-2010-3333	**CVE-2012-0158**
Exploit vulnerability	Stack-based buffer overflow	Some Active X controls in MSCOMCTL.OCX
Attack by	● RTF	● Web site ● Office document ● RTF

2.2 分析 Office 的工具比較

因為 RTF 屬 Office 的文件之一，故我們用可以分析 Office 的工具來作 RTF 的分析，如：officeMalScanner[6]、offvis[7]、pyOLEscanner[8]，在分析上我們以執行環境、工具是否能分析 RTF 為主要的分析重點，其次，因為 Office 惡意文件有時會伴隨著 OLE 檔進行攻擊，所以是否能分析 OLE 檔也是我們分析工具的重點之一。

表 3 分析 Office 的工具之比較

	officeMalScanner	**offvis**	**pyOLEscanner**
Type	.exe	.exe	.py
Open source	No	No	Yes
Analysis on RTF	Yes	Yes	No
Inspect shellcode	Easy	Difficult	No
Detect OLE file	Yes	No	No
Inspect OLE file	No	Yes	Yes

由表 3 可得知以下訊息：

a. 以 OS 的角度去分析，officeMalScanner 跟 offvis 只可以在 Windows 環境下執行，而 pyOLEscanner 則可在 Linux 或 Windows 環境下執行。
b. 是否能分析 RTF 檔的部分，officeMalScanner 跟 offvis 都可做分析，但是在分析的結果來看，officeMalScanner 會直接判斷是否有毒而 offvis 只對特定漏洞有反應，而對於其他種漏洞只會顯現其架構，必須進一步的分析，故較耗費時間；而 pyOLEscanner 則專對 Office 檔中內嵌的 OLE 檔做完整分析，其餘部分較不支援。
c. 在分析內嵌 OLE 檔的部分，officeMalScanner 只顯示該 OLE 檔是否有害的資訊，對於進一步的分析較不支援，而 offvis 及 pyOLEscanner 則可對 OLE 檔做進一步的分析。

因為 officeMalScanner 可在 Windows 環境下執行且可分析 RTF 並回報 shellcode 及 shellcode 的所在位置，故在實驗部份我們以 officeMalScanner 作為主要的分析工具。

3. 正向工程：惡意文件嵌入技術

由於我們好奇惡意文件是如何嵌入惡意程式的，所以除了分析樣本外，我們嘗試做出惡意文件以更深入了解惡意程式是如何嵌入。

3.1 利用文件本身製作惡意文件

由於文件本身具有可程式化之功能，如 Office 內建的 macros 或 PDF 的 action，所以我們試著利用此特性去製作惡意程式。在本小節我們以 Office 為範例（圖 2），利用 macros 來寫一個簡單的惡意行為之程式。

```
Sub Run_Program(program, argments, visiblity, wait_on_execute)
Dim WshShell As Variant
Set WshShell = CreateObject("WScript.Shell")
WshShell.Run program & " " & argments & " ", Visibility, wait_on_execute
End Sub

Sub Workbook_Open()
Const VISIBLE = 1, INVISIBLE = 0
Const WAIT = True, NOWAIT = False
Run_Program "ieplore.exe http://www.yahoo.com/", "", VISIBLE, NOWAIT
End Sub
```

圖 2 打開文件自動存取惡意網頁之 macros

圖 2 所顯示的 macros 在打開文件後便會暗中瀏覽特定網頁，使用此方法的惡意程式需搭配惡意網頁及社交工程來執行，因為 Office 已對這種手法做出防範，內嵌 macros 的檔案本身檔名會有所不同，故受害者會發現檔案內含程式，故多少會有警覺，而且 Office 會將 macros 預設為不執行，所以此文件要可以正常執行必須利用社交工程誘使受害者將 macros 功能打開才可順利執行。

3.2 漏洞（exploitation）攻擊概要

首先將大量的資料放入文件中的陣列，企圖造成溢位，企圖迫使 CPU 執行同在記憶體中的 shellcode 已執行欲執行的程式[9]，漏洞攻擊流程圖如下圖 3 所示：

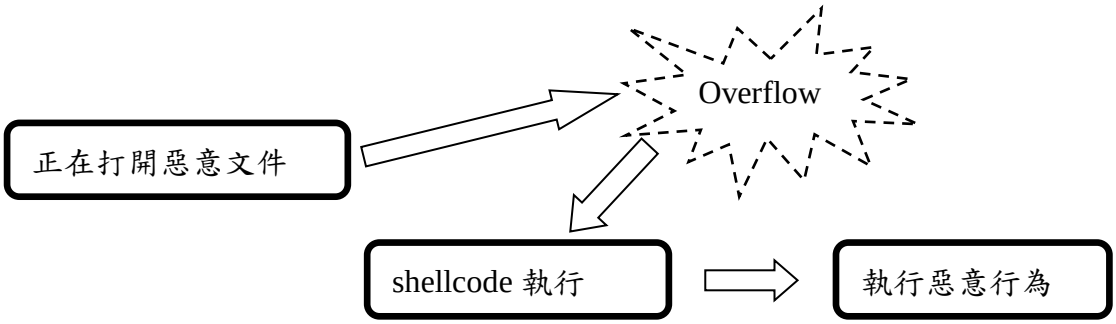

圖 3 漏洞攻擊流程圖

3.3 比較正向工程兩種惡意文件類型

從 4.1 及 4.2 我們提到兩種做法的惡意文件，在這邊我們對這兩種文件進行比較，由表 4 可知，利用文件內建程式進行攻擊被察覺的風險較大，故會降低攻擊的成功率；但是利用漏洞攻擊的文件在外表上及內容跟一般文件雷同，受害者較不容易察覺，所以成功機率較大。

表 4 分別使用文件內建程式跟漏洞攻擊之 Malware 比較

		By macros	**By exploitation**
Find the difference		Easy	Difficult
When opening	Display message isplay memacros”	Yes	No
	Need to enable macros	Yes	No

4. 實驗

4.1 實驗流程

此節先利用逆向工程分析樣本文件，再利用正向工程進行製作。實驗流程圖如下圖 4 所示，在此實驗中，我們利用 RTF 分析工具分析所有樣本，並依結果分成 clean 及 maliciou 兩類（分析工具偵測率高達 95%），在此時我們將重點著重於可疑文件的部分，利用分析工具之結果去對可疑文件進行調查及分析。

首先，利用分析工具得知 shellcode 內容及其所在位置，從而了解樣本所有可能藏有 shellcode 的部分，在此我們稱為惡意區塊。然後，一方面調查 RTF 的架構並深入調查樣本惡意區塊的部分，另一方面我們對所有惡意區塊進行分析，一一統計樣本中所有使用惡意區塊，從而得知不同種漏洞最常使用到的惡意區塊，因為樣本中並非一定只使用一個惡意區塊進行攻擊，故依照樣本所使用到的惡意區塊將樣本分類，從而得知相同漏洞中所有惡意區塊搭配種類。

最後，我們進一步將樣本的惡意區塊嵌入正常文件，並利用線上掃毒工具的分析結果，如果被嵌入的正常檔案有跟原本樣本一樣的分析結果，就證明確實為惡意區塊。

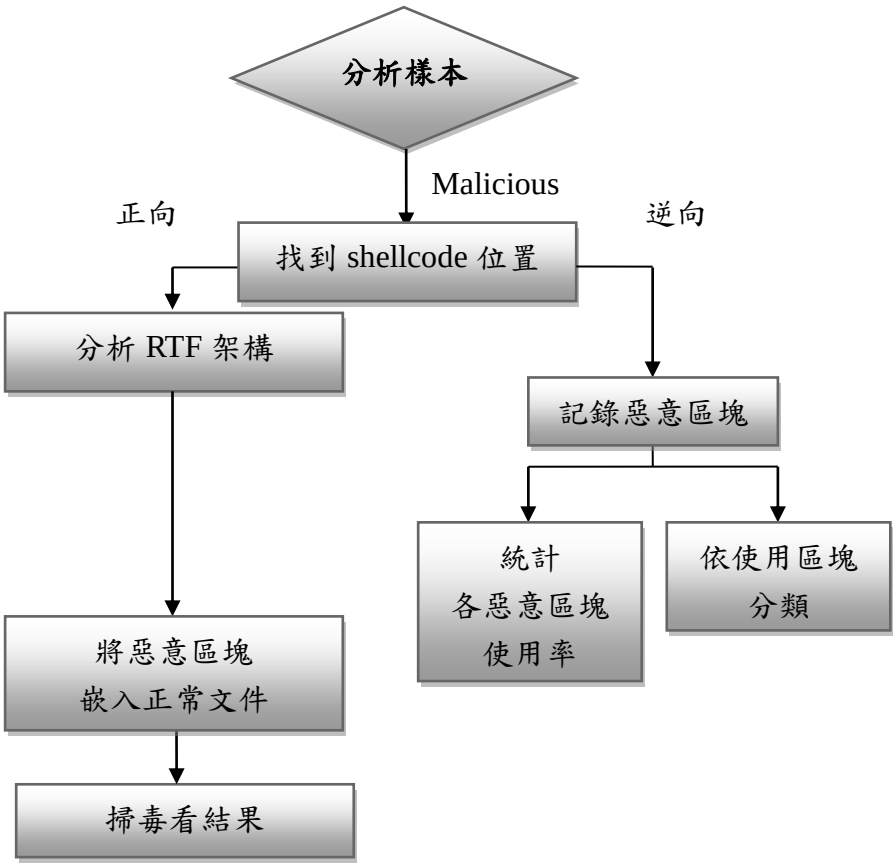

圖 4　實驗流程圖

4.2 分析惡意 RTF 的方法

4.2.1 利用 officeMalScanner 找出 RTF 的 shellcode

在分析的部分，由於 officeMalScanner 本身只能分析 doc、xls、ppt，所以我們用的是裡面附屬的 RTFScan，從圖 5 的分析結果我們可以看到這個樣本裡面包含一個 shellcode 儲存在 SV 中，左方的紅線框住的數字代表嵌入的 shellcode，由於 SV 屬 pFragments 的一部分，故在之後我們皆以 pFragments 而非 SV 作為紀錄。

```
Scanning for shellcode in SV...
FS:[30] <Method 4> signature found at offset: 0xef

648B7130                                mov esi, fs:[ecx+30h]
8B760C                                  mov esi, [esi+0Ch]
8B761C                                  mov esi, [esi+1Ch]
8B5E08                                  mov ebx, [esi+08h]
8B7E20                                  mov edi, [esi+20h]
8B36                                    mov esi, [esi]
66394F18                                cmp [edi+18h], cx
75F2                                    jnz $-0Ch
C3                                      ret
60                                      pushad
8B6C2424                                mov ebp, [esp+24h]
8B453C                                  mov eax, [ebp+3Ch]
8B540578                                mov edx, [ebp+eax+78h]
01EA                                    add edx, ebp
8B4A18                                  mov ecx, [edx+18h]
8B5A20                                  mov ebx, [edx+20h]
```

圖 5 officeMalScanner 分析結果

4.2.2 分析 Shellcode 在 RTF 的位置

利用 officeMalScanner 的結果找到 shellcode 在 RTF 檔中的位置，而此位置為 pFragments 中的 SV 中。

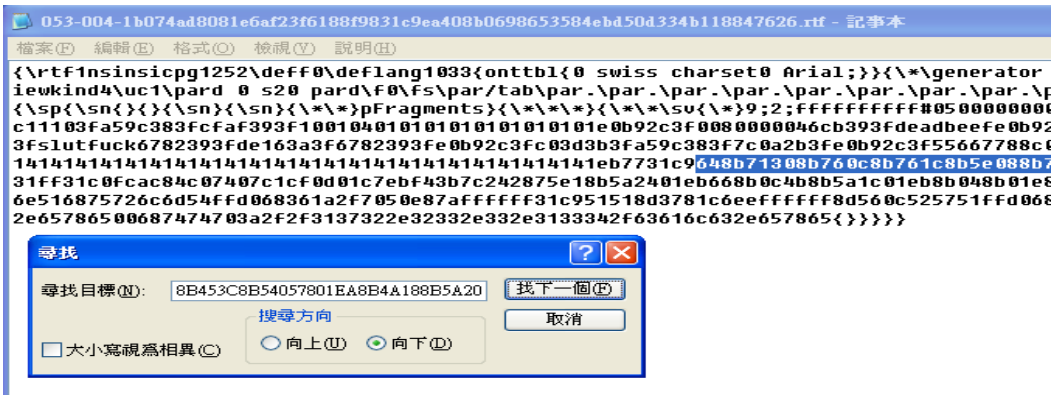

圖 6 shellcode 在檔案的位置

4.2.3 分離 shellcode 並嵌入正常文件

為證明判斷位置確實為惡意區塊，將樣本的惡意區塊嵌入正常文件，並用線上掃毒工具作掃毒，發現其結果和樣本的結果雷同，故證實為惡意區塊。

5. 實驗結果

5.1 文件與程式切割

在 RTF 檔中，依照單純文字區塊即可存資料區塊，我們可以把整個結構分成兩個部分，切割方式如圖 7 所示：

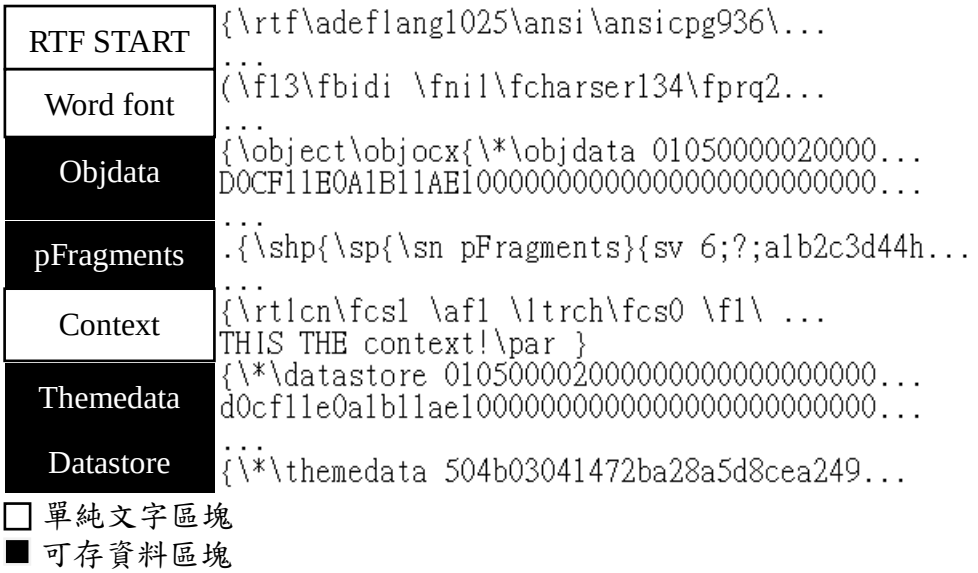

圖 7 RTF 區塊切割

切割的依據是因為在統計完惡意區塊後發現會出現 shellcode 的區塊皆可存入大量資料於陣列中，且並沒規範資料字串長度，似乎符合可用來作弱點攻擊的

特性，而 RTF 主要格式皆用於控制文字，本身並不支援程式執行，故利用可存入資料的格式造成溢位，進一步執行事先存入的 shellcode，是 RTF 進行漏洞攻擊的方法。

5.2 惡意區塊判別

由表 5 我們可以發現：

a. 對不同漏洞進行攻擊的文件，其常用惡意區塊不同

如：CVE-2010-3333 經常使用到 pFragments。

CVE-2012-0158 經常使用到 Objdata。

b. 幾乎一半的樣本皆會利用 Outside Structure 做為惡意區塊，Outside structure 不屬任何架構，RTF 本身並不會產生此部分，由一大群亂碼文字組成，所有樣本的 Outside structure 皆置於文件底端。

表 5 RTF 惡意區塊比較

		CVE-2010-3333	**CVE-2012-0158**	**All RTF samples**
Detection rate		84%(81/92)	98%(294/300)	95%
Malware region	pFRagments	88%	1%	19%
	Themedata	0%	28%	23%
	Datastore	17%	1%	4%
	Objdata	1%	54%	43%
	Outside structure	46%	61%	58%

5.3 樣本依惡意區塊分類

將兩種漏洞攻擊的樣本依使用到的分類，如下圖 8、9 所示，會發現：

a. 兩種樣本皆有多種惡意區塊搭配方式

在 CVE-2010-3333 可依使用到的惡意區塊分成 7 種類型，而 CVE-2012-0158 則可依使用到的惡意區塊分成 10 種類型。

b. 兩種漏洞中最常使用的惡意區塊搭配

如 CVE-2010-3333 大部分只由 pFragment 作為惡意區塊（42%），而 CVE-2012-0158 大部分以 Objdata 及 outside structure 作為惡意區塊（39.6%）。

由此可知，同一種漏洞攻擊，可以由不同區塊搭配進行攻擊，也就是說，不一定皆用同樣的惡意區塊進行攻擊，故製作惡意文件的手法多樣化。雖然如此，但是從分析結果來看，還是可以發現每種漏洞有最常使用到的惡意區塊搭配，就如同此次實驗我們可以得知兩種不同漏洞攻擊樣本最常進行的惡意區塊搭配已進行攻擊。

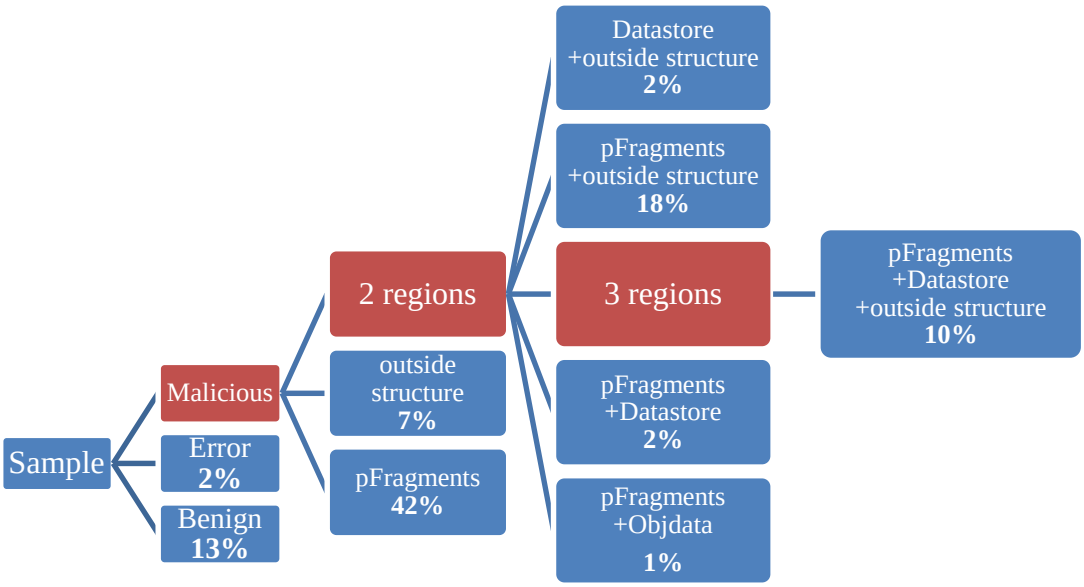

圖 8 CVE-2010-3333 樣本之分類

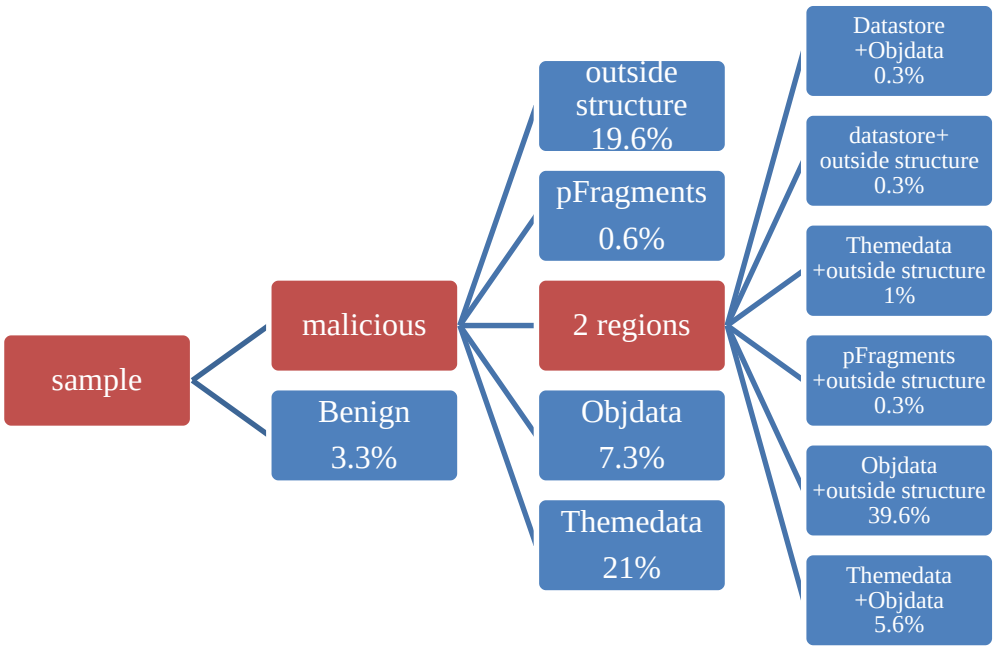

圖 9 CVE-2012-0158 樣本之分類

6. 結論

本文利用逆向工程及正向工程進行研究，藉由逆向工程，我們得知 shellcode 及其嵌入位置，發現以下四點：

a. 樣本利用 RTF 可存資料區塊造成溢位執行 shellcode
b. 不同漏洞攻擊經常使用的惡意區塊不一定相同
c. 相同漏洞攻擊可用多種惡意區塊組合進行攻擊
d. 兩種漏洞攻擊樣本最常使用到的惡意區塊組合

從 a 我們可以瞭解 RTF 為何可以成為惡意文件的原因，從 b、c 兩點我們可

以發現惡意文件製作方法多變，以及藉由 d 我們知道兩種惡意文件主要的趨勢。

而藉由正向工程，我們可以證明逆向工程所指出的惡意區塊確實為惡意行為本身，並藉由正向工程了解到如何做出惡意文件。

本文主要對象鎖定利用漏洞攻擊的惡意文件，經過這次實驗我們對於漏洞攻擊的樣本有更進一步的了解，但是本文只列出少部分區塊而並未列出所有可存資料的區塊，而單單只是兩種樣本所利用到的區塊，且目前也無法得知那些區塊可以做為惡意區塊而那些不行，故未來惡意文件還是有變化的空間，這告訴我們製作惡意文件手法的多元。

參考文獻

[1] Aditya K Sood and Richard J. Enbody, “Targeted cyberattacks: a superset of advanced persistent threats”, IEEE Security & Privacy, 2013 January/February.

[2] Command Five Pty Ltd, “Advanced Persistent Threats: A Decade in Review”, June 2011.

[3] Common Vulnerabilities and Exposures, “CVE-2010-3333”, http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2010-3333.

[4] Common Vulnerabilities and Exposures, “CVE-2012-0158”, http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2012-0158.

[5] Microsoft Office, “Rich text format(RTF) Specification version 1.9.1”.

[6] Frank Boldewin, “Analyzing MSOffice malware with OfficeMalScanner”, 30th July 2009.

[7] Russ McRee, “Offvis 1.0 Beta: office visualization tool”, ISSA Journal.

[8] pyOLEscanner, https://github.com/Evilcry/PythonScripts/blob/master/pyOLEScanner.py.

[9] Malicious Office Files Analysis – pyOLEScanner and Cryptoanalytical Approach https://www.evernote.com/shard/s9/note/42f28727-8671-4ec0-83b8-4635574d0c38/wishi/crazylazy#st=p&n=42f28727-8671-4ec0-83b8-4635574d0c38