The Decline of Exploit Kits as an Exploitation Strategy

Ma, Zicong
zicong.ma16@imperial.ac.uk

Abstract
Exploit kit is a software kit, which normally runs on web servers. It identifies the vulnerabilities in the client machines by communicating with it through the browser, then using malicious code to exploit the vulnerability to deliver and execute malicious software on the victim's machine. The first form of exploit kits started appearing in 2006, then it slowly became more and more popular. With the introduction of the Blackhole exploit kit, it became one of the most used cyber-criminal strategy in 2012. However, following some arrests to the dominant exploit kits' author, the growth trend of exploit kits was successfully stopped. We analysed data on blogs and twitters from the year of 2016 and 2017 to get a better picture on the most recent trend of the exploit kits. Through several graphs and tables that we produced, we concluded that the exploit kit landscape is in a state of decline and predicts that this trend will carry on unless there is some explosion in the effectiveness of exploit kits.

1. Exploit Kits
Background on Drive-by Malware
There were days where you would have to click on something on the webpage to download a software onto your device. Nowadays, opening any website can put you at risk of getting malicious programs installed onto your computer because of "drive by" malwares, as suggest by its name, when you visit a website or simply "drive by" it, the malware and malicious scripts can download and execute in the background without you noticing it. This process is automated by some cyber-criminals and made into a software toolkit which is called Exploit kit.

Exploit kits, in brief, automates the pipeline of gathering information from victim's machine, choosing suitable vulnerabilities based on the information and delivering the malware which will be executed on the victim's computer. Exploits kits are usually found on either malicious website that the operator hosts, or legitimate website that have been compromised. Compromising popular websites has been very effective in targeting as many victims as possible. For example, recently a popular bank in India, the Cosmos Bank, was compromised with the infamous RIG Exploit Kit for 3 days, putting tens of thousands of the customer's machines at risk.

Since 2006, exploit kits has been becoming increasingly popular over the last decade. The first forms of it were WebAttacker and Mpack which were sold on Russian underground forums. After that, many different exploit kits started appearing one by one in the wild, but none of which seemed to dominate the market, until the appearance of the Blackhole exploit kit in 2010. The Blackhole Exploit kit became more and more popular over those years and peaked at 2012 where it made up of 29% of the web threats detected by Sophos and 91% of the exploited kit detection by AVG. Many actions have taken place trying to stop the rapid development of exploit kits, whether through developing more sophisticated detection algorithm, or arresting the exploit kit authors, which is what happened to the Blackhole exploit kit author and the later dominant Angler exploit kit's author.

However, in the recent years, there had been seen a declining trend of exploit kits. There are many potential reasons for that, such as the disappearance of some of the dominant exploits due to arrests, the constant evolution of anti-virus products, reduced number of vulnerabilities and many more.

How a typical EK works

One of the reason for the exploit kits success is the low-level knowledge requirement for the customer to successfully deliver malwares. The exploit kits are usually provided as a software with very user-friendly statistics to the people buying it, which can be seen in Figure 2. Normally, the exploit kit shows the statistics of success

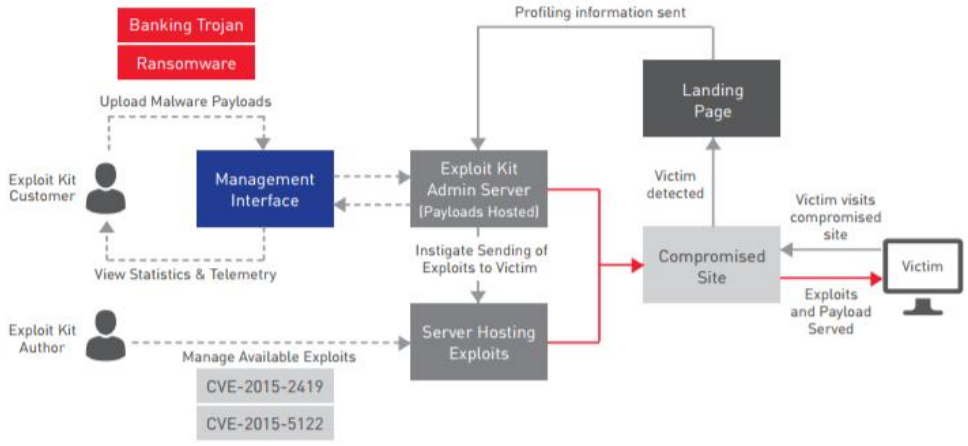


Figure 1 - Infrastructure of an Exploit Kit by CERT-UK

rate on different countries, different computer systems and different software targeted. The customer only has some simple jobs, one is to upload some payloads (normally in the form of banking Trojan or Ransomware) to drop onto the victim's machine, and another is to manage the initial infection vector through a website. After that, they are good to go because the rest will be managed by the admins of the exploit kit.

The admins of the exploit kit have sole control over the rest of the procedure. They decide what exploit server and the proxy servers are going to be used. Another very important job for them is that they need to provide vulnerabilities delivered to the victim. However, the main vulnerabilities that can be exploited is decreasing and a lot of them have been patched by the software producers, therefore this part of the job can be difficult sometimes.

If the management of the exploit kit has been done properly, it is not difficult for a victim to get infected if they don't have good awareness or up-to-date software. Once the victim visits a compromised website they would be redirected a few times to the exploit kit through proxy servers. The victim's browser is then profiled, JavaScript

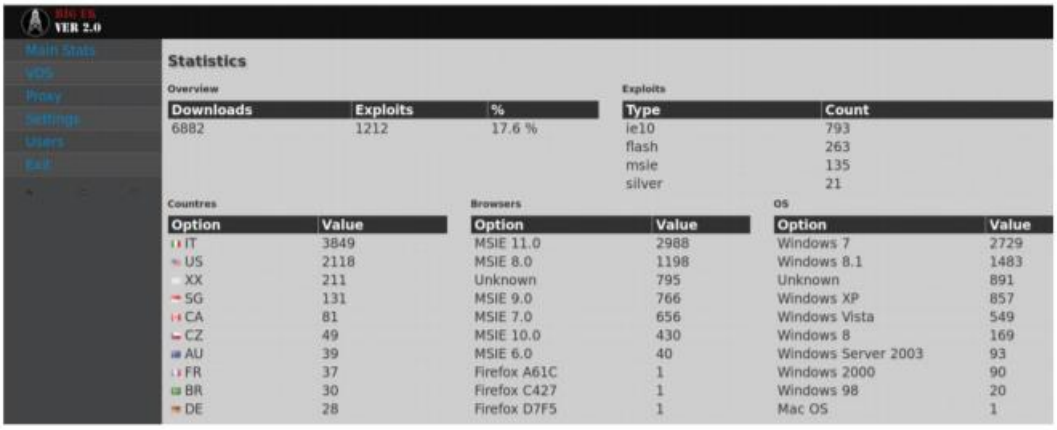


Figure 2 - RIG's user interface from CERT-UK

is used to check whether virtual machine or security product is used. This is normally done by JavaScript function 'Ufe3S' to check which of the system components are present. Operating system is also checked thereafter, since the exploit kit is mainly focused on the Windows operating system, they would try to avoid pointlessly exploiting Mac or Linux Operating Systems. After that, they should check if the browser of the victim is vulnerable to the exploit. Since a lot of the browser have implemented the Sandbox technology into them, it is now very hard to execute the payload on the client's computer directly through the browser. However, the browser's plugins

sometimes allow the victim's machine to be able to be exploited. Most of the vulnerabilities of the plugins come from Adobe Flash, Adobe Reader, Java Oracle, and Microsoft Silverlight.

Related Work

Drive-by attacks

Over the past decade, there has been a large number of researches done in the area of preventing drive-by attacks. NOZZLE [1] is an run-time detector that target specifically at heap spraying attacks. Heap spraying attack, which mostly happens in web browsers, increases the success rate of exploits by injecting as many malicious codes in the heap. JavaScript allows the attacker to easily inject the code into a webpage. With NOZZLE, each object on the heap is examined and a static analysis of the code is carried out to detect weather it is malicious or not. The detector was proven to be very effective as it successfully detected more than 2,000 heap-spraying exploits and it produces no false positives when ran over 150 popular websites.

Other researchers have taken a different approach by introducing ADSandbox [2]. ADSandbox is an analysis system which helps against websites that primarily attacks through JavaScript. Since JavaScript does not have built-in sandbox functions, the embedded JavaScript of the website is executed within an isolated environment and its actions are logged. Heuristics is used on the logs to detect whether the website is malicious or not. This approach targets at the drive-by attacks in a more general way compared to the NOZZLE that is mentioned above because it would prevent the user against more types of different attacks. It is also very flexible in a way that although the code might have been obfuscated and encoded several times, the behaviour can still be caught by the logs which helps many researchers in manually analysing malicious JavaScript. However, this type of method can result in high processing power required for some of the websites which use more JavaScript codes. A similar research have been done to produce CUJO [3], as an extension to web proxy, for automatically detecting and preventing drive-by attacks. For this specific system, machine learning is used to efficiently analyse for malicious code, thus it produces a median run-time of 500ms per web page which is considerately better than the previously discussed ADSandbox.

Exploit Kits

As Exploit kit has become increasingly popular starting from 2010, there has been more researches targeted at specifically this toolkit instead of the more general drive-by attacks. WebWinnow [4] tackles the problem of detecting whether a given URL is hosted by an exploit kit. This paper analyzed the worlflow of 40 different exploit kits, and they were observed to apply a machine learning strategy to extract some of the distinguishing features of attack and defence centric exploit kits. Research by I. Nikolaev, M. Grill, and V. Valero [5] also focuses on exploit kit website detection but they are only detecting it solely based on the information extracted from HTTP proxy logs. They rely on the fact that the exploit kit characteristics are common across different ones and extract several indicators from the crucial part of the characteristics.

In this paper [6], the researchers collected 70+ popular exploit kits and they were able to successfully deploy 30+ kits, which have been further analysed in terms of offensive component, defensive component, management component, code protection and code re-use. For the offensive component, they could discover that all kits implement user agent protection, while most implements UA validation, exploit selection and exploit obfuscation. In terms of defensive mechanisms, kits use IP blocking, payload obfuscation, crawler evasion and checking itself against virus databases. In terms of management component, most of the kits provides the user with market statistics and only basic settings. They also discovered that not many kits use code protection and there are barely any common codes used by the malware authors.

A more recent research [7] leverages the inherent structural patterns in HTTP traffic to classify exploit kit instances. It captures the behaviour such as the browser makes multiple requests from malicious servers to download the payload. Those type of interactions are captured and modelled in a "tree-life" form and the detection process is modelled as a sub-tree similarity search problem. This also allows the researcher to determine where the root sites, or advertising network was launched from.

Researches discussed above on exploit kits had focused mainly on examining the server-side components of the exploit kits, but KIZZLE [8], a signature compiler for detecting exploit kits, is the first prevention technique specifically designed for finding exploit kits. The analysis on the unpacked code of the exploit kits found that they don't differ by much between codes because the reuse between different versions of the kits. Therefore, KIZZLE can generate anti-virus signatures for detecting EKs, which can create new signatures within hours. A. K. Sood

and S. Zeadally[9] also analysed the built-in features of the exploit kits, they conducted a feature-oriented comparative analysis of the primarily used attack techniques in the last few years. The main features that they compared were JavaScript obfuscation, JavaScript-based redirection, JavaScript content injection on the fly, JavaScript-based domain-generation algorithm, Malicious URL distribution through phishing and they discovered that most of the features are used for every exploit kit analysed and the only feature that is only implemented by limited number of exploit kit is JavaScript-based domain-generation algorithm.

2. Information Sources Analysed

There have been many sections from different industries reports featuring the exploit kits in the previous years. Those are also what we analysed at first to get a better understanding on how exploit kits have been developing over the past years. Figure 3 was taken from Demystifying the exploit kit from CERT-UK. This table shows the different vulnerabilities discovered throughout 2015 and which of them was used in different exploit kits. It is

Target Application	Vulnerability CVE	Angler	Fiesta	Magnitude	Neutrino	Nuclear	RIG
Adobe Flash	CVE-2015-0310	•					
Adobe Flash	CVE-2015-0311	•	•	•	•	•	•
Adobe Flash	CVE-2015-0313	•			•		
Adobe Flash	CVE-2015-0336	•		•	•		
Adobe Flash	CVE-2015-0359	•	•	•	•		•
Adobe Flash	CVE-2015-3014	•		•		•	
Adobe Flash	CVE-2015-3090	•		•	•	•	•
Adobe Flash	CVE-2015-3113	•	•	•	•	•	•
Adobe Flash	CVE-2015-5119	•		•	•	•	•
Adobe Flash	CVE-2015-5122	•		•	•	•	•
Adobe Flash	CVE-2015-5560	•				•	
Adobe Flash	CVE-2015-7645	•				•	
Microsoft Internet Explorer	CVE-2015-2419	•		•	•	•	•
Microsoft Silverlight	CVE-2015-1671	•		•			

Figure 3 - Vulnerabilities provided by CERT-UK

Figure 4 - Vulnerabilities by TrendMicro in 2016

obvious to see that most of the CVE comes from Adobe Flash while there are also several on MSIE and Microsoft Silverlight. Even though they all have been patched by the vendors, there could still be many customers that are late on patching the software on their own machines. With so many exploits available and kit like Angler which used all those CVEs, there still would have been many victims vulnerable to the attacks back in 2015.

In a presentation used by company Qualys in the RSA Conference 2016, they also provided a table and it is extremely like the data provided by Figure 3, with dominated number of exploit kits focusing on Adobe Flash. They also provided another graph, which shows that despite the thousands of exploits available, there are very few that can be implemented and exploited by the exploit kits. We were able to acquire some data on different vulnerabilities used in the year of 2016. Figure 4 was published by TrendMicro in their 2016 yearly roundup, it is shown that there are even less vulnerabilities used by the exploit kits, namely CVE-2016-4117, CVE-2016-1001, and CVE-2016-1019 for Adobe Flash; CVE-2016-0189 for MSIE and CVE-2016-0034 by Microsoft Silverlight. Figures 5 shows the number of access to exploit-kit hosting URL, also from TrendMicro, indicating the general declining trend of exploit kits.

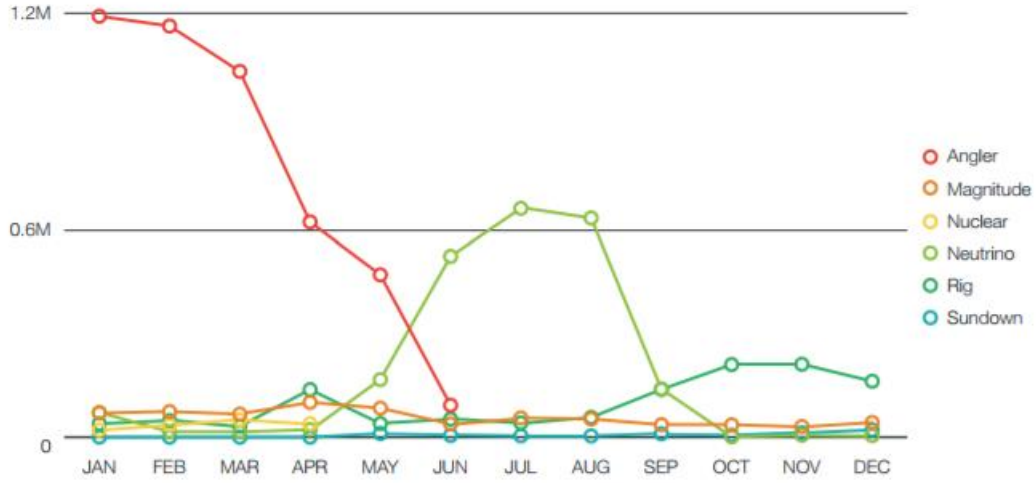


Figure 5- number of hits to different exploit-kit hosting URLs by TrendMicro

Summary of Main Sources

After reading the industry reports, we had a better idea on what sort of data we need to collect and how we would collect them. We need to collect data on how active each different exploit is at different stage of the year, preferably like those shown in Figure 3. We thought about collecting information from existing data over the Internet. However, after going through many different websites, we found out that there is no source that gives data on the frequency of appearance either daily or monthly.

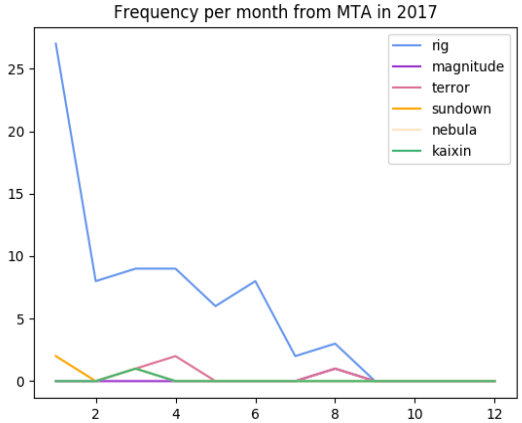


Figure 6 - Frequency of mention per month from MTA in 2017

Main Blogs

In the process of doing looking through different websites, we discovered a few interesting blogs that reports about malwares, such as Malware Traffic Analysis – which frequently reports network traffic related to malware infections; Malware Breakdown – a website of similar function but it also provides a category function to help separate out the posts related to exploit kits, as well as ZeroPhage, a blog primarily focusing on exploit kits. We thought that we could produce some form of data such as frequency of different exploit kits by the mentions from the blogs that we analysed. We produced simple scraping programs for all the three blogs to produce frequency graphs. Firstly, we used python's built-in library to retrieve the webpage's source code, then we would walk through the page source string and find, instances of "exploit kit" or "ek".

Whenever such instance is detected, we would look for the exploit's name which would be normally right before "exploit kit", and we could also retrieve the data of the post similarly. After that, we would group the data into months and plot the frequency using python library matplotlib. Note all the graph were produced by the date of 2017/09/15, therefore there are no data on the last two months of the year 2017.

Malware Traffic Analysis

This is the first blog that we looked at because it has been posting for about 5 years and it posts very frequently throughout the years, therefore we feel we could get a more reliable form of data. An example of the data that we are able to retrieve is shown in Figure 6, which is a frequency graph of the exploit kits in 2017.

Malware Breakdown

This blog separated their posts into several categories and one of them is exploit kit. Therefore, it was straightforward for me to scrap the data and produce a graph in Figure 7 below.

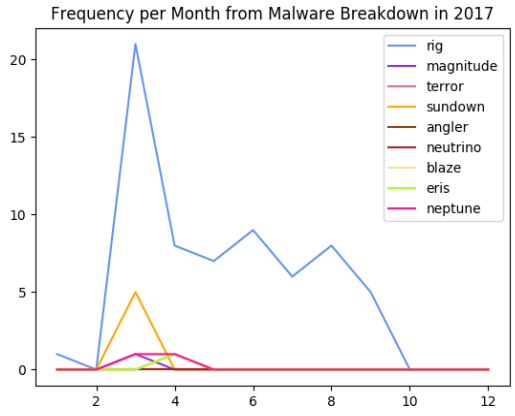


Figure 8 - Frequency of mention per month from Malware Breakdown 2017

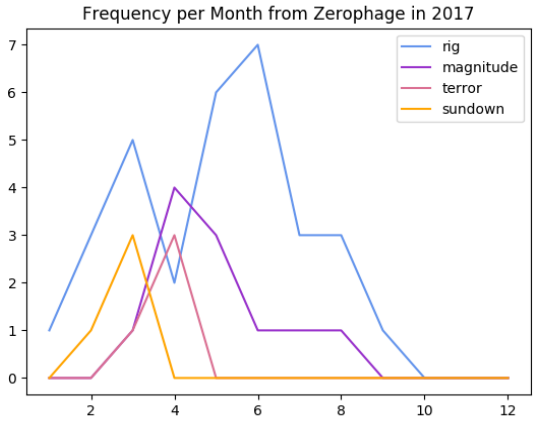


Figure 7 - Frequency of mention per month from Zerophage in 2017

- **Zerophage**

  This blog only started posting January 2017, but it was sufficient since we are only focusing on the year of 2017. The frequency of the data is lower compared to the other two blogs as well, the graph produced is shown in figure 8.

Comparison of the data collected

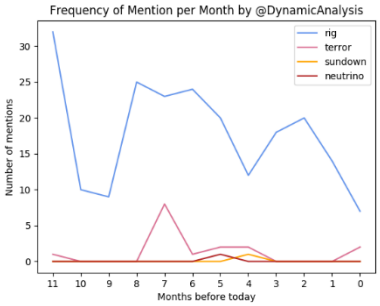
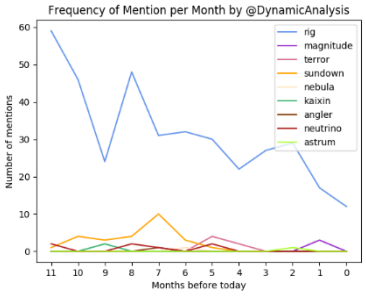
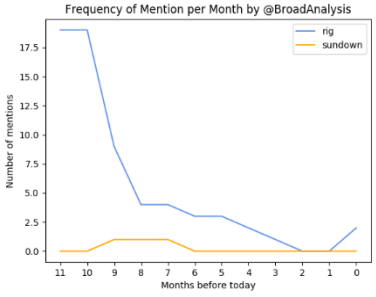
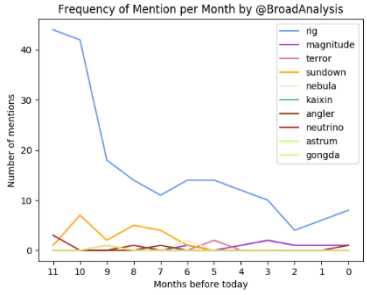
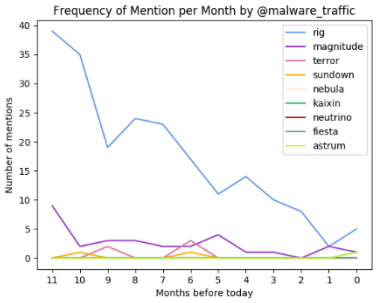
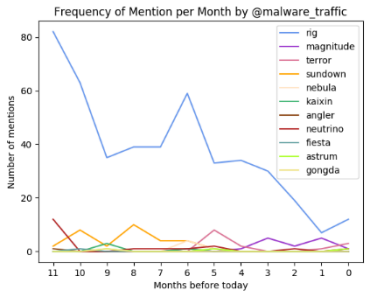
Looking at the three graphs produced from the three blogs, there are not much significant similarities except for the dominance of RIG exploit kit throughout the year of 2017. Other exploit kits have barely made an appearance in the blogs. All three graphs display a declining trend in the dominant RIG exploit kit towards September 2017 when the graphs were produced.

Main Twitter Accounts

After having analysed a few of the blogs available there, we realised same can be done for Twitter. A lot of the people working on network security have access to Twitter accounts and constantly update them with information regarding malware incidents. First of all, we need to find a few twitter accounts that regularly updates information related to Exploit Kits before we start data collection. We did this by manually looking through different Twitter accounts and looking at the "who to follow" section of each of them. Here is the list of Twitter users that we felt

would be relevant: @BroadAnalysis, @DynamicAnalysis, @TrendMicro, @TrendLabs, @virusbtn, @executemalware, @nao_sec, @Cyberasfuc, @truely_secure, @malwrhunterteam.
We faced several challenges as we tried to collect data from twitter automatically. Firstly, Twitter's API has a limit to the number of tweets can be accessed from a single user of which is 3,240. For the majority of the accounts that we are analysing, there are significantly more tweets than the ones that are accessible and it is nearly impossible for us to access some of the data earlier. Therefore, to adapt with the amount of data that is available to us, we thought we could try to access data only in the last 12 months and analyse it by the number of months before the current time to get a better understanding on how the exploits have been progressing in the past year.
When we finished the program and produced some data, we realised that some of the accounts either don't have enough data points to see any sort of trend, or there still isn't enough data over the last 12 months because there have been over 3,240 tweets in the past year. We could produce reasonably looking graphs for only three of the accounts.
When we were looking at the tweets collected, we realised that most of them came in the form of retweets. We thought it would be also interesting to see how the graphs would look without those retweets, therefore we produced another column containing no retweets.

Table of graph produced by the date of 12/09/2017

Twitter Name	Only Original Tweets	Including Retweets
@DynamicAnalysis		
@BroadAnalysis		
@Malware_traffic		

Comparing all the data shown in the table above, the first thing that we noticed is that there is *not* a huge difference between the shape of the graph with or without retweets. The trend that can be observed in all the graph above is a declining trend of the most dominant exploit kit RIG over the past year, with a minimal amount of activity for the other exploit kits.

Difficulties of Getting Accurate Exploitation Data

There is an overall shortage of reliable information on data in the field of exploitation. We reached out to Symantec, McAffee, Trendsmicro, and Kaspersky. We were not able to get data beyond what is contained in a single 2016 report from Trendsmicro in Figure 5. The vendors either didn't have the information we requested or were reluctant to share it with us. We anticipate other researchers facing similar issues in terms of getting reliable data. At the same time, we do not quite see a reliable way to collect this information without broadly crawling the web or collaborating with a honeypot-style project.

In Figure 5 shown previously, we can easily see that there is a decline of Angler throughout the first half of the 2016, it was also shown that at the end of 2016 there has been more appearance made by the RIG exploit kit. In our analysis of the blogs and twitter accounts, it has shown that RIG has been dominating in the past year, while also being on a declining trend. Other exploit kits have made minor appearances here therefore not much trend can be seen from them.

3. Understanding Changes to EKs Over Time

Overall Setup

In this section, we wanted to outline how the content included in different exploit kits changed over time in the recent years. To achieve that, a large amount of data of exploit kits files over the past year is required and luckily the website Malware Traffic Analysis would provide exactly what we needed. In every one of the posts related to exploit kit on the website, the page contains a zip file which includes the malware artefacts. Sometimes it includes the files dropped over several runs, someone only one run.

Normally, the structure of the file included is: a .txt landing page file, a .swf flash exploit (normally present in most of the files sometimes there can be .dll or none), an .exe malware payload and a .tmp file. We wrote a program to automatically go through all the links throughout the last two years: 2016 and 2017 and download every zip file which include the malware artefacts. We also realised that it would be very hard to analyse just the zip file so we wrote another program that would unzip every single file that we downloaded for analysis later.

General Information on the exploit data collected

	Number of achieves	**Number of files per archive**	**Average size for archive/kb**
2016	233	9.18	1,470
2017	53	8.25	750

After we downloaded all the files and unzipped them, we decided to use the Virustotal to check for the vulnerabilities inside them. Virustotal provides a python api which we can make use of. We could upload the file using the Virustotal API and check the response to see two things: 1: the detection rate, 2: if "cve" is included in the detection message. CVE is an abbreviation for Common Vulnerabilities and Exposures which exploit kits normally must make use of to successfully exploit a machine.

Analysis of Captures in 2016

EK	Occurrences	VirusTotal detects CVE	Local detection for CVE	Automatic+Local
Rig	100	45	30	60
Angler	59	14	2	16
Neutrino	53	0	0	0
Sundown	7	3	0	3
Magnitude	6	2	2	4
Kaixin	3	3	0	3

Analysis of Captures in 2017

EK	Occurrences	VirusTotal detects CVE	Local detection for CVE	Automatic+Local
Rig	45	25	41	42
Sundown	3	2	2	2
Magnitude	2	0	1	1
Nebula	1	1	0	1
Terror	3	3	1	3

Most of the CVE discovered was from Flash files. One of the reason for that could be that on Virustotal only a limited amount of the detection message is shown so the CVE part of it is not shown. Another reason could be that most of the exploitation is only done through the flash files anyways so it would make sense for most of the CVE to be detected inside of them.

However, we also found that even for the Flash files, there are a fair number of cases that no CVE is shown to be detected, so we decided to have a look at the flash files which were detected not to have a CVE and compare them to the one which are detected to have a CVE. For this part, we used the flash decompiler JPEXS Free Flash decompiler.

Discussion of SWF files

Looking through the resources online about the exploitation code of different flash CVEs, we found 3 of the CVE which we could try to write a program to detect them locally rather than uploading them to the Virustotal. Those are CVE-2015-8651, CVE-2015-7645 and CVE-2015-5122. We will discuss the detection of them below.

```
private function li32_overflow(param1:int):int
{
    var _local_2:int;
    var _local_4:int;
    var _local_3:int;
    _local_3 = (2147483644 + this.add(param1));
    _local_4 = li32(_local_3) /*FlasCC (Alchemy)*/ ;
    _local_3 = (_local_3 - 2097148);
    _local_2 = li32(_local_3) /*FlasCC (Alchemy)*/ ;
    _local_3 = (_local_3 - 4);
    _local_2 = li32(_local_3) /*FlasCC (Alchemy)*/ ;
    return (_local_4);
}

private function si32_overflow(param1:int, param2:int):void
{
    var _local_4:int;
    var _local_3:int;
    _local_3 = (2147483644 + this.add(param1));
    si32(param2, _local_3); //FlasCC (Alchemy)
    _local_3 = (_local_3 - 2097148);
    _local_4 = li32(_local_3) /*FlasCC (Alchemy)*/ ;
    _local_3 = (_local_3 - 4);
    _local_4 = li32(_local_3) /*FlasCC (Alchemy)*/ ;
}
```


Figure 10 - Exploit code of CVE-2015-8651 from 360

```
//
// CVE-2015-8651
//
private function method_238(param1:int) : int // Read memory
{
   var _loc2_:* = 0;
   var _loc3_:* = 0;
   var _loc4_:* = 0;
   _loc4_ = 2147483644 + this.method_80(param1);
   _loc3_ = li32(_loc4_);
   _loc4_ = _loc4_ - 2097148;
   _loc2_ = li32(_loc4_);
   _loc4_ = _loc4_ - 4;
   _loc2_ = li32(_loc4_);
   return _loc3_;
}

private function method_134(param1:int, param2:int) : void // Write memory
{
   var _loc3_:* = 0;
   var _loc4_:* = 0;
   _loc4_ = 2147483644 + this.method_80(param1);
   si32(param2,_loc4_);
   _loc4_ = _loc4_ - 2097148;
   _loc3_ = li32(_loc4_);
   _loc4_ = _loc4_ - 4;
   _loc3_ = li32(_loc4_);
}
```

Figure 9 - Exploit code of CVE-2015-8651 from forcepoint

CVE-2015-8651

The exploit code that we found both from 360 and Forcepoint shown above are extremely similar. Therefore, we decided we could try to detect some of the patterns in the code such as the number 2147483644. We found later that this number does not appear in any of the other flash files that does not base on CVE-2015-8651, therefore it seems that this number is specially reserved for this specific exploit. Therefore, my exploit code detection was mainly based on detecting this specific number.

CVE-2015-7645

CVE-2015-7645 is a type-confusion vulnerability. According to the SonicWall Security Center, the vulnerability exists when the writeExternal function is overwritten by another with the same name.
Therefore, in my code for detecting this vulnerability, we specifically look for code there the writeExternal

```
public var writeExternal = 7777;

public function subexternalizable()
{
   super();
}
```

Figure 11 - CVE-2015-7645 code from SonicWall

function is called or overwritten.

CVE-2015-5122

CVE-2015-5122 makes use of the opaqueBackground property in Adobe flash. Therefore, we would look for whether the opaqueBackground keyword is used in the code.

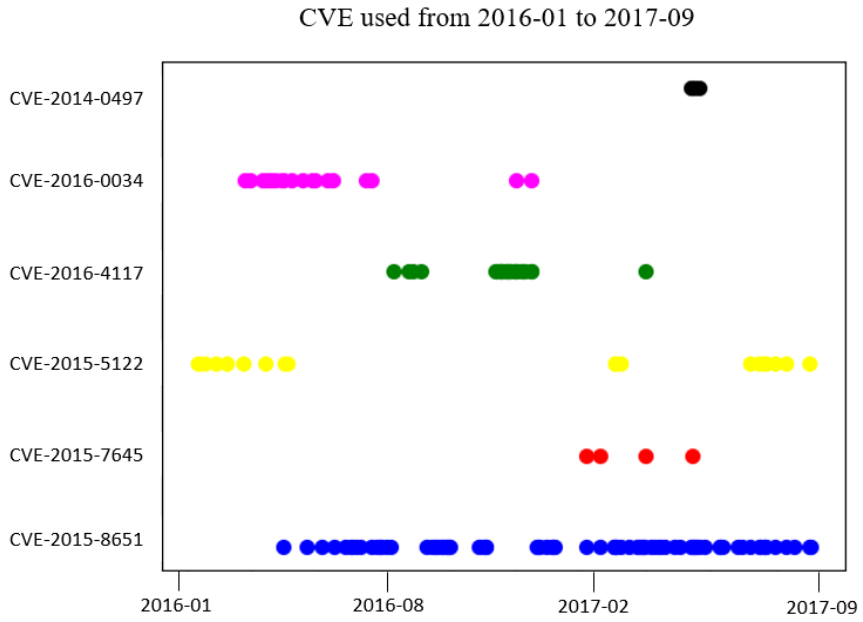


Figure 12 - different CVEs used over the last two years

Note that all the detection method described above are not complete by any means. However, there is not a method to check for the false positive rate as there does not exist a service which can check all the vulnerabilities perfectly. As far as the result is concerned, it does increase the number of flash files to be detected to contain the CVEs, which can be seen in the local detection column in the table above.

4. Conclusions

Both the blog data and the Twitter data have shown a *decline* as the general exploit kit trend. The blog data shows that as we are coming close to the end of the 2017, the amount of mentions to the exploit kits, mainly RIG, have been seen at a much lower frequency. The blog owners have been focusing on malspam, ransomware and other areas of exploitation.

The amount of mentions through different twitter accounts have also shown a clear decline over the last 12 months which could also indicate the declining trend of exploit kits. There has also been a consolidation in the EK space

thanks to the diminish of some of the competitive exploit kits. Some of the exploit kits that dominated 2016 did not make an appearance in the year of 2017: Angler dominated the first half of 2016, completely disappeared on the market, Neutrino, which made up most of the market in the middle of 2016 also barely made any appearance in the year of 2017 since they moved towards private usage with established customers. Other not so popular exploit kits such as Magnitude and Terror still made an appearance in the year of 2017, but up to 80 to 90% of the mentions through Twitter accounts and blogs can be contributed towards the RIG exploit kit.

There are some other factors for the decline, one of the main reasons could be the limited number of CVEs which is also shown in our study in the previous section. There has been only a very limited number of CVEs implemented in the exploit kits, compared to Angler which used about 6-7 CVEs, exploit kits only use about 1-2 CVEs recently. One of the main reasons is the lack of new CVEs being discovered and the software authors are actively patching their software. Therefore, only the people who are late towards patching their software could be vulnerable to the attacks, which is a small amount of people. Another reason can be contributed towards the rapid development of protection from different browsers. Internet Explorer, which was known to be a very vulnerable browser in the past years, have been releasing many newer versions which are much more resistant to the drive-by attacks. The development of better detection techniques from anti-virus products also means that the success rate of the exploit kits would be much lower than what it was in the earlier years.

TrendMicro also stated in their blog that due to the decline of exploit kits, recently spam has been the preferred way to deliver ransomware to victims. Unless there is an unexpected explosion in the popularity and effectiveness of new exploit kits, we expect that cyber criminals will likely turn their focus to the social engineering methods – phishing or spam email.

Bibliography

[1] P. Ratanaworabhan, B. Livshits, and B. Zorn, "NOZZLE : A Defense Against Heap-spraying Code Injection Attacks," *Proc. 18th USENIX Secur. Symp.*, pp. 169–186, 2009.

[2] A. Dewald, T. Holz, and F. C. Freiling, "ADSandbox," *Proc. 2010 ACM Symp. Appl. Comput. - SAC '10*, p. 1859, 2010.

[3] K. Rieck, T. Krueger, and A. Dewald, "Cujo: Efficient Detection and Prevention of Drive-by-Download Attacks," *ACSAC '10 (26th Annu. Comput. Secur. Appl. Conf.*, pp. 31–39, 2010.

[4] B. Eshete and V. N. Venkatakrishnan, "WebWinnow: Leveraging Exploit KitWorkflows to Detect Malicious URLs," *Proc. 4th ACM Conf. Data Appl. Secur. Priv. - CODASPY '14*, pp. 305–312, 2014.

[5] I. Nikolaev, M. Grill, and V. Valeros, "Exploit Kit Website Detection Using HTTP Proxy Logs," *Proc. Fifth Int. Conf. Network, Commun. Comput. - ICNCC '16*, pp. 120–125, 2016.

[6] V. Kotov and F. Massacci, "Anatomy of exploit kits: Preliminary analysis of exploit kits as software artefacts," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 7781 LNCS, pp. 181–196, 2013.

[7] T. Taylor *et al.*, "Detecting Malicious Exploit Kits using Tree-based Similarity Searches," *Proc. ACM Conf. Data Appl. Secur. Priv.*, pp. 255–266, 2016.

[8] B. Stock, B. Livshits, and B. Zorn, "Kizzle: A signature compiler for detecting exploit kits," *Proc. - 46th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Networks, DSN 2016*, pp. 455–466, 2016.

[9] A. K. Sood and S. Zeadally, "Drive-By Download Attacks: A Comparative Study," *IT Prof.*, vol. 18, no. 5, pp. 18–25, 2016.