Automatic Classifying of Mac OS X Samples

Spencer Hsieh, Pin Wu and Haoping Liu
Trend Micro Inc., Taiwan

TREND MICRO LEGAL DISCLAIMER

The information provided herein is for general information and educational purposes only. It is not intended and should not be construed to constitute legal advice. The information contained herein may not be applicable to all situations and may not reflect the most current situation. Nothing contained herein should be relied on or acted upon without the benefit of legal advice based on the particular facts and circumstances presented and nothing herein should be construed otherwise. Trend Micro reserves the right to modify the contents of this document at any time without prior notice.

Translations of any material into other languages are intended solely as a convenience. Translation accuracy is not guaranteed nor implied. If any questions arise related to the accuracy of a translation, please refer to the original language official version of the document. Any discrepancies or differences created in the translation are not binding and have no legal effect for compliance or enforcement purposes.

Although Trend Micro uses reasonable efforts to include accurate and up-to-date information herein, Trend Micro makes no warranties or representations of any kind as to its accuracy, currency, or completeness. You agree that access to and use of and reliance on this document and the content thereof is at your own risk. Trend Micro disclaims all warranties of any kind, express or implied. Neither Trend Micro nor any party involved in creating, producing, or delivering this document shall be liable for any consequence, loss, or damage, including direct, indirect, special, consequential, loss of business profits, or special damages, whatsoever arising out of access to, use of, or inability to use, or in connection with the use of this document, or any errors or omissions in the content thereof. Use of this information constitutes acceptance for use in an "as is" condition.

Spencer Hsieh
Trend Micro Inc., Taiwan
spencer_hsieh@trendmicro.com

Pin Wu
Trend Micro Inc., Taiwan
pin_wu@trendmicro.com

Haoping Liu
Trend Micro Inc., Taiwan
haoping_liu@trendmicro.com

Contents

Abstract

Thanks to the rapidly increasing volume of malware, the security industry has been struggling to improve automatic malware classification for many years. Many recent market research reports have suggested that the growth of Apple's Mac OS X has outpaced PC platforms for several years. This shifting trend is attracting more malware authors to develop malware for Mac OS X. In this paper, we present a study of classifying Mac OS X malware with a set of features extracted from Mach-O metadata and its derivatives in a sample collection from VirusTotal.

Like the PE format for Windows, the Mach-O format provides a variety of features for classification. We collected more than 300,000 Mach-O samples submitted to VirusTotal during 2015–16, and filtered out irrelevant samples, such as samples for iOS and PowerPC. We then generated metadata from the Mach-O files using tools like nm, otool and strings. Meta information from sample files, such as segment and section structures, imported functions of dynamic libraries, printable strings, etc., were used as features for classifying Mac OS X samples. Additionally, we included derivative numerical features created from meta information, which have been introduced into learning-based malware classification widely in recent research studies, e.g. function call distribution, structure complexity, etc.

This study summarizes the statistical change in view of Mac OS X malware families, and the structure trending between benign and malicious samples between 2015 and 2016. With our collection of more than 300,000 files and over 4,000 malicious samples, our feature evaluation is based on composition analysis of different malware families in both aspects of meta and derivative features. This work uses a variety of classification algorithms to generate predictive models with the 2015 dataset, and to analyse the test results with the 2016 samples and their difference from AV vendors' detections on VirusTotal. We also discuss the effectiveness of selected features, by ranking their importance levels in a predictive model among our classification tests with the 2015–16 dataset.

Introduction

In the early days, anti-virus tools were designed to detect malware by recognizing a fingerprint of a given file. However, because the number of malware patterns is prone to grow with the number of malware samples, signature-based detection had difficulty in coping with the exponential growth in malware samples. Some non-signature based approaches were developed to deal with this issue. Because of the prevalence of Windows operating systems, most research has focused on the PE executable format of Windows. However, the success of Apple in the personal computer market in recent years has attracted an increasing number of malware authors to create new malware for Mac OS X. In this paper, we describe our study of automated processing of the VirusTotal sample collection, including parsing Mach-O meta attributes, classification tests, and data processing for malware analysis tasks.

In the rest of Section 1, we will provide an overview of the Mach-O format and discuss other work of relevance to our studies. In Section 2, we present the statistics of our collection and the changes between the sample sets taken at different times. In Section 3, we show the results of testing predictive models for classification tasks. In Section 4, we study the malware families in the sample collection. Finally, we address discussions and our conclusions in Section 5.

Mach-O format

The format of executable files for Mac OS X is Mach-O [1], which is similar in many ways to the PE format for Windows [2] and the ELF format for Linux.

A Mach-O file can be divided into three parts: a header at the beginning of the file, load commands, and data in segments. The header contains only a few fields, e.g. the magic number for identification, file type, CPU type, and number of load commands.

In PE format, information about the memory layout and file structure are described in the optional header and section table. In Mach-O format, all of this type of information, as well as information about how to load an executable file into memory, such as the dynamic linked libraries needed for execution, and the entry point, are all described in load commands. In other words, load commands provide a variety of features for classification.

The actual data of a program reside in segments. Similar to the PE format, the ‘_TEXT’ segment usually contains the code for execution, and program data resides in the ‘_DATA’ segment.

Although Mach-O format is pretty similar to PE format, there are some major differences between them. Several Mach-Os can be combined into a single file, i.e. fat binary file, and these Mach-Os may have totally different behaviors. Therefore, we may need to consider them as different instances in classifying.

Another difference is that many PE files have a resource section, which may contain icons, cursors, pictures, and other kinds of resources needed for the executable file. For Mach-O most of these resources reside in separate files.

Related work

While there have been some studies on non-signature-based malware detection or classification, most of them focus on the Windows PE executable format.

In [3], the authors present a system that extracts 189 features from a PE file and describe how they used these features to detect malware. Although some of these features are extracted from the resource section of a PE file, most of the others are extracted directly from the headers of a PE file, and most of them have corresponding features in Mach-O format. They also used as features the DLLs referenced by a PE file. Mach-O files have similar information regarding the libraries referenced by an executable file.

There are other pieces of research that have tried to figure out the effectiveness of these features [4], but they are also based on similar features extracted from the headers of a PE file, or a combination of these features.

Besides the PE format, Shahzad and Farooq [5, 6] proposed a system for the ELF format of Linux. They used a similar approach and extracted 383 features from the ELF headers.

Mac OS X Samples Dataset

All of our samples were collected from VirusTotal between September 2014 and March 2016[1]. We downloaded samples tagged 'macho' and their reports every day and fed them into our pre-processing system. We stored the SHA256 of each file with its filename, and filtered out any corrupted samples. Because we were only interested in samples for modern Mac OS X, any samples that were not for I386 or X86_64 were discarded. After that, we used some scripts and built-in tools, such as *otool*, *nm* and *strings*, to extract structure information and features from the downloaded samples.

In our study, VirusTotal samples represent real malware data from security vendors. As shown in Figure 1, we collected Mac OS X samples from VirusTotal at an average rate of 2243.68 samples per day across 489 fetch days.

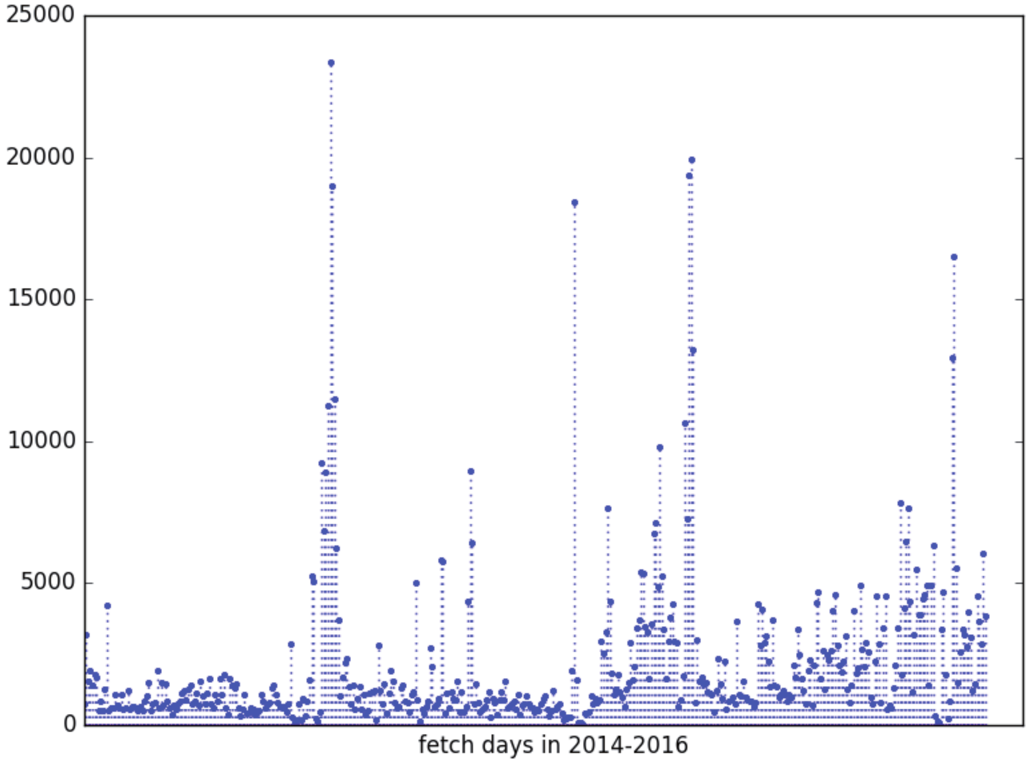

Figure 1. VirusTotal sample fetch logs

Stats

After we had sifted out irrelevant samples and corrupted files, we were left with 626,900 samples and over 4,000 malicious files for analysis. As shown in Table 1, half of the sample collection were *x86_64* type and the other half were *i386* type. In Table 2, we present the statistics of the file types in the collection, and in Table 3 we give the statistics for all malicious samples in the collection, including mean, standard variation and percentiles, and each sample is described in eight Mach-O attribute features: number of commands, number of segments, number of load dylibs, signed, number of uncommon segments, number of segment names, number of sections and number of section names. We use the malware detections of primary anti-virus vendors in VirusTotal to identify malicious samples.

CPU type	Samples (%)
i386	313,630 (50.02%)
x86_64	313,270 (49.97%)

Table 1. CPU type statistics.

File type	Samples	Percentage
MH_EXECUTE	238,028	37.97%
MH_DYLIB	224,537	35.82%
MH_BUNDLE	132,434	21.13%
MH_OBJECT	25,541	4.07%
MH_DYLIB_STUB	4,829	0.77%
MH_DSYM	1,480	0.24%
MH_DYLINKER	41	0.01%
MH_CORE	7	0.00%
MH_PRELOAD	3	0.00%

Table 2. File type statistics

#:4200	mean	std	min	25%	50%	75%	max
n_cmds	23.0	6.2	2	20	23	26	56
LC_SEGMENT	4.1	0.6	1	4	4	4	7
LC_LOAD_DYLIB	9.9	4.8	0	7	10	12	39
signed	0.5	0.5	0	0	1	1	1
n_uncommon_segms	1.3	0.5	1	1	1	2	4
n_segnames	4.1	0.6	1	4	4	4	7
n_sections	26.2	8.0	1	22	29	33	40
n_sectnames	25.5	7.7	1	22	28	32	38

Table 3. Statistics of malicious samples

As we go deeper into the samples, we extract and summarize the referenced library names in the Mach-O metadata across each sample set. A list of the most commonly referenced libraries is shown in Table 4 – we can see that the same libraries were commonly used across both of the batches.

In our data exploration stage, we observe a significant difference in scale between the malicious and non-malicious portions in our collection, that is approximately 4,000 versus 600,000. We also discover the fact that malicious and non-malicious samples have great overlaps in the feature space of basic Mach-O attributes.

Batch 1	Batch 2
libSystem	libSystem
libobjc	CoreFoundation
CoreFoundation	libobjc
libstdc++	Foundation
Cocoa	libgcc_s
libgcc_s	AppKit
Foundation	libstdc++
ExceptionHandling	CoreServices
CoreServices	Cocoa
AppKit	ApplicationServices
libnspr4	Security
AudioToolbox	Carbon
ApplicationServices	libz

Table 4. Most referenced libraries

Our experiment strategy turns to using known malware samples as clues for classifying new malware or unknown samples, and we shift our focus to the malicious samples in our collection. As shown in Figure 2, we apply a dimension reduction method [7] to investigating the distribution of Mach-O attribute features that are associated with malicious samples, and visualize the malicious samples in a two-dimensional reduced space of descriptive features.

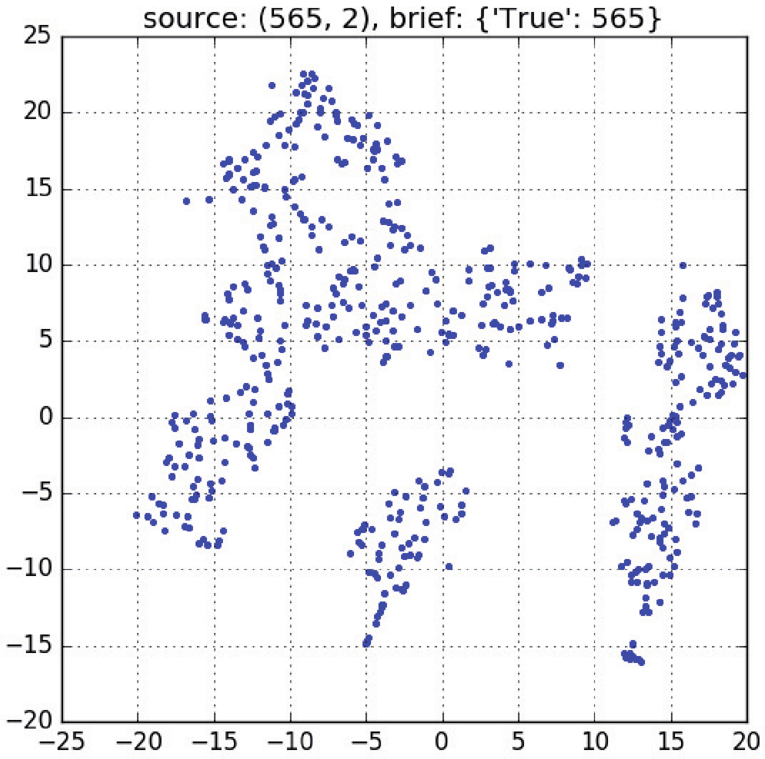

Figure 2. Illustration of malware family distribution in reduced space

Classification of Mach-O Files

Mach-O meta features

In our classification task, we start with a traditional predictive model scenario. The sample set Batch 1 is used as training data, and the newer sample set, Batch 2, represents testing data. We adopt the eight descriptive features used to calculate data stats.

Effectiveness analysis

We tested the effectiveness of applying statistical predictive models to the classification of Mac OS X samples using Mach-O attributes. As shown in Table 5, we used three classic predictive models: Naïve Bayes, nearest neighbour, and decision tree classifiers for the experiment. The results show that statistical methods merely reach recall rates of 30% to 60% with Mach-O meta information. However, in further statistical analysis, we observed that the current performance barrier is based on information limitation. Too many of these Mach-O meta features are seen in both malicious and non-malicious samples, making them unsuitable for predicting the nature of an unknown file.

Classifier	Recall rate
Naïve Bayes	60.0%
Nearest neighbors	35.2%
Decision tree	33.1%

Table 5. Prediction performance of three classification algorithms in testing

Malware Families

Different anti-virus companies use different ways of naming malware. However, in most cases the family name is the same. We unite the malware family labels of several primary vendors provided on VirusTotal, and perform a statistical analysis of selected malware groups with the united malware family labels in perspectives of time change and basic meta information of Mach-O for further analysis.

Statistics change

The Mac OS X samples uploaded to VirusTotal have changeable distributions of malware family. In our collection, over 20,000 raw detection names are grouped into 200 united malware family labels. As shown in Table 6, there is some similarity in the top 11 malware family labels appearing in the two sample sets. Over two different time frames, six malware groups have reappeared in all top-most places: VSearch, Genieo, InstallCore, SpiGot, MacKeeper and Yontoo, which indicates that each of these malware families constantly has a considerable number of new samples appearing in VirusTotal. In contrast, some of the malware families in the top 11 list appeared only in the 2016 sample set, e.g. TuneupMyMac, AMC (Advanced MacCleaner) and Tinyv, which are probably a new focus of Mac OS X samples.

Batch 1 (#:2760)			Batch 2 (#:1440)			Batch 1+2 (#:4200)		
Name	Count	Percentage	Name	Count	Percentage	Name	Count	Percentage
Bundlore	611	22%	Genieo	197	14%	Bundlore	646	15%
VSearch	421	15%	InstallCore	167	12%	VSearch	484	12%
Genieo	263	10%	MacKeeper	160	11%	Genieo	460	11%
InstallCore	222	8%	Tinyv	157	11%	InstallCore	389	9%
SpiGot	147	5%	TuneupMyMac	123	9%	MacKeeper	237	6%
MacKeeper	77	3%	AMC	94	7%	SpiGot	204	5%
Refog	70	3%	Yontoo	65	5%	TuneupMyMac	180	4%
KeyLogger	69	3%	VSearch	63	4%	Tinyv	157	4%
Morcut	63	2%	SpiGot	57	4%	Yontoo	125	3%
Downloader	61	2%	GetShell	52	4%	GetShell	102	2%
Yontoo	60	2%	Elite	42	3%	AMC	94	2%

Table 6. Top malware families in sample set

Composition

Among the united malware families, we select 13 notable malware groups: Bundlore, Flashback, Genieo, GetShell, InstallCore, MacKeeper, Morcut, Ocean-Lotus, Refog, SpiGot, TuneupMyMac, VSearch and Yontoo, and then make a statistical study of their Mach-O attribute features. Table 7 shows that some statistical consistency can be observed in the Mach-O features of the two batches of malware, suggesting that, by looking at many attributes, it will be possible to statistically distinguish malware from non-malware.

	Bundlore	#: 611	Genieo	#: 263	Refog	#: 70	SpiGot	#: 147	VSearch	#: 421	Yontoo	#: 60
	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
ncmds	23.47	3.87	23.20	4.21	33.24	11.48	24.16	2.75	20.19	3.05	22.27	2.80
LC_ SEGMENT	4.02	0.41	4.21	0.58	4.36	0.51	4.25	0.44	4.24	0.66	3.93	0.55
LC_LOAD_ DYLIB	10.62	2.91	10.87	3.75	17.56	9.92	10.03	2.81	6.98	1.97	7.83	2.53
signed	0.17	0.38	0.77	0.42	0.90	0.30	0.97	0.16	0.38	0.49	0.75	0.44
n_ uncommon_ segms	1.10	0.30	1.33	0.47	1.39	0.49	1.25	0.44	1.48	0.50	1.12	0.32
n_segnames	4.02	0.41	4.21	0.58	4.36	0.51	4.25	0.44	4.24	0.66	3.93	0.55
n_sections	30.28	6.67	25.31	7.71	27.57	6.02	23.99	4.90	26.63	4.00	27.75	7.05
n_sectnames	29.34	6.54	24.94	7.40	26.91	5.57	23.66	5.37	25.97	3.74	27.23	6.81

	Bundlore	#: 35	Genieo	#: 197	Refog	#: 19	SpiGot	#: 57	VSearch	#: 63	Yontoo	#: 65
	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
ncmds	23.63	1.19	24.16	3.65	38.84	17.19	26.04	4.63	21.19	4.29	21.97	2.60
LC_SEGMENT	4.09	0.28	4.00	0.63	4.47	0.51	4.23	0.42	4.27	0.65	3.72	0.57
LC_LOAD_DYLIB	8.77	1.46	12.04	3.34	23.63	15.17	11.95	4.84	7.65	3.33	7.06	1.96
signed	0.94	0.24	0.88	0.32	0.79	0.42	0.93	0.26	0.44	0.50	0.97	0.17
n_uncommon_segms	1.09	0.28	1.34	0.47	1.47	0.51	1.23	0.42	1.52	0.50	1.06	0.24
n_segnames	4.09	0.28	4.00	0.63	4.47	0.51	4.23	0.42	4.27	0.65	3.72	0.57
n_sections	33.09	0.28	25.72	6.93	27.84	7.56	28.70	2.88	26.89	4.00	27.02	5.76
n_sectnames	32.09	0.28	25.13	6.65	27.26	7.13	28.12	2.69	26.16	3.72	26.42	5.55

Table 7. Malware families in Batch 1 and Batch 2

In Table 8, signed and unsigned code samples both exist in several malicious or ‘unwanted-ware’ families. There may be the opportunity to use the signatures of code-signed malicious samples for identification.

	Flash back	#: 28	Get Shell	#: 50	Install Core	#: 222	Mac Keeper	#: 77	Ocean Lotus	#: 63	Tuneup MyMac	#: 57	Morcut	#: 63
	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
Ncmds	15.68	1.63	11.56	0.91	24.09	1.59	24.75	6.87	17.55	0.51	25.91	1.26	23.17	5.76
LC_SEGMENT	3.89	0.31	4.72	0.45	4.00	0.00	3.92	0.70	4.55	0.51	4.51	0.50	3.98	0.85
LC_LOAD_DYLIB	5.89	1.79	1.72	0.45	9.91	1.71	11.94	4.11	5.00	0.00	11.93	0.26	10.94	5.74
signed	0.00	0.00	0.00	0.00	1.00	0.00	1.00	0.00	0.00	0.00	1.00	0.00	0.03	0.18
n_uncommon_segms	1.00	0.00	1.72	0.45	1.00	0.00	1.51	0.50	1.55	0.51	1.51	0.50	1.63	0.58
n_segnames	3.89	0.31	4.72	0.45	4.00	0.00	3.92	0.70	4.55	0.51	4.51	0.50	3.98	0.85
n_sections	13.68	2.31	5.96	3.17	30.15	2.71	27.68	4.09	16.77	4.10	31.51	1.45	26.37	8.10
n_sectnames	13.54	2.01	5.96	3.17	29.15	2.71	27.12	3.67	16.77	4.10	30.51	1.45	25.86	7.91

	Flash back	#: 6	Get Shell	#: 52	Install Core	#: 167	Mac Keeper	#: 160	Ocean Lotus	#: 9	Tuneup MyMac	#: 123	Morcut	#: 2
	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
Ncmds	13.67	3.39	11.46	0.85	24.73	1.48	24.01	5.92	18.44	0.53	27.28	0.69	20.00	22.63
LC_ SEGMENT	3.67	0.52	4.77	0.43	4.00	0.00	3.90	0.70	4.44	0.53	4.52	0.50	3.00	2.83
LC_LOAD_ DYLIB	4.33	2.88	1.77	0.43	10.39	1.89	11.10	3.50	6.00	0.00	12.76	0.48	11.00	15.56
Signed	0.00	0.00	0.00	0.00	0.99	0.08	1.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00
n_ uncommon_ segms	1.33	0.52	1.77	0.43	1.00	0.00	1.51	0.50	1.44	0.53	1.52	0.50	1.50	0.71
n_segnames	3.67	0.52	4.77	0.43	4.00	0.00	3.90	0.70	4.44	0.53	4.52	0.50	3.00	2.83
n_sections	14.00	3.58	5.62	2.98	30.34	3.32	27.73	3.15	15.11	1.05	32.63	0.48	18.50	19.09
n_sectnames	13.67	3.27	5.62	2.98	29.34	3.32	27.12	2.83	15.11	1.05	31.63	0.48	18.00	18.38

Table 8. Malware families in Batch 1 and Batch 2 (code signed or unsigned)

As shown in Figure 3, we visualize the distribution of the samples of the selected 13 malware families in a 2-D space reduced from eight-dimensional original space. As indicated by the distribution of same-colour clusters in different regions, it shows that samples within a malware family can be aggregated closely to several central samples, and also within the malware family, the samples are variant in group and similar within subgroup.

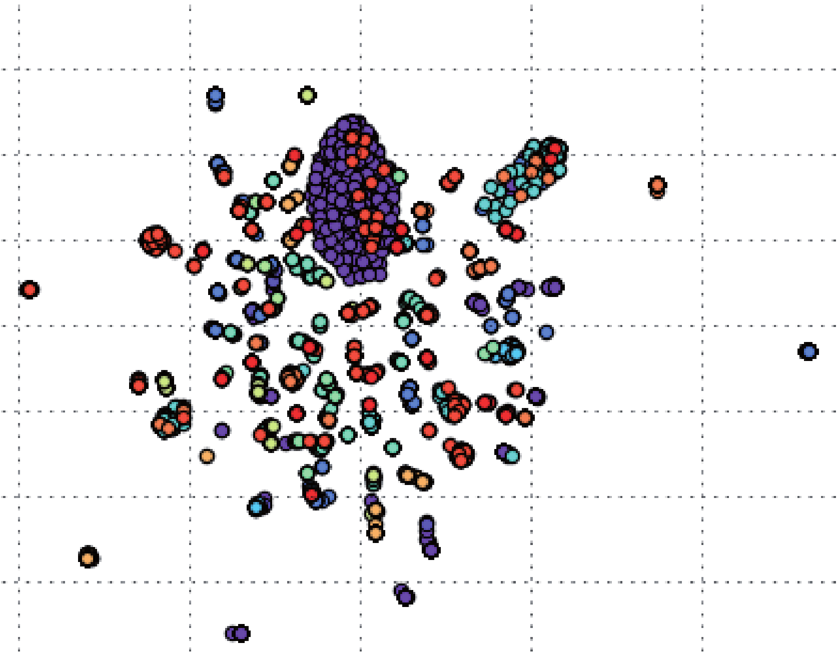

Figure 3. Illustration of sample distribution of selected 13 malware families in reduced space.

Conclusion

By observing the dataset, we can figure out that the number of pieces of Mach-O malware is still pretty small compared to the number of pieces of PE malware. Among the Mach-O malware, the most common malware families by number are adware (e.g. Bundlore, VSearch and Genieo). Because of the nature of adware, these pieces of malware are also more likely to have signatures of code compared to the backdoor malware used in targeted attacks, like Morcut or OceanLotus.

In comparison with PE, we speculate that the lack of a resource section in Mach-O files could limit the ability to describe the difference between malicious and normal samples at the meta information level. Our work will look for further effective representation of Mac OS X samples for classification tasks in future.

REFERENCES

1. OS X ABI Mach-O File Format Reference. https://web.archive.org/web/20090901205800/http:/developer.apple.com/mac/library/documentation/DeveloperTools/Conceptual/MachORuntime/Reference/reference.html.

2. Portable Executable and Object File Format Specification. https://web.archive.org/web/20160418132427/https:/download.microsoft.com/download/e/b/a/eba1050f-a31d-436b-9281-92cdfeae4b45/pecoff.doc.

3. Shafi q, M. Z.; Tabish, S. M.; Mirza, F.; Farooq, M. PE-miner: Mining structural information to detect malicious executables in real time. Recent advances in intrusion detection, pp.121–141. Springer Berlin Heidelberg, 2009.

4. Raman, K. Selecting features to classify malware. InfoSec Southwest 2012 (2012).

5. Shahzad, F.; Farooq, M. ELF-Miner: using structural knowledge and data mining methods to detect new (Linux) malicious executables. Knowledge and information systems 30, no. 3 (2012): 589–612.

6. Liao, Y. PE-Header-Based Malware Study and Detection. Retrieved from the University of Georgia: http://www.cs.uga.edu/~liao/PE_Final_Report.pdf.

7. Van der Maaten, L.; Hinton, G. Visualizing data using t-SNE. Journal of Machine Learning Research 9, no. 2579–2605 (2008): 85.

TREND MICRO™

Trend Micro Incorporated, a global cloud security leader, creates a world safe for exchanging digital information with its Internet content security and threat management solutions for businesses and consumers. A pioneer in server security with over 20 years experience, we deliver top-ranked client, server, and cloud-based security that fits our customers' and partners' needs; stops new threats faster; and protects data in physical, virtualized, and cloud environments. Powered by the Trend Micro™ Smart Protection Network™ infrastructure, our industry-leading cloud-computing security technology, products and services stop threats where they emerge, on the Internet, and are supported by 1,000+ threat intelligence experts around the globe. For additional information, visit **www.trendmicro.com**.

Securing Your Journey
to the Cloud