IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

A Multi-Classifier Network-based Crypto Ransomware Detection System: A Case study of Locky Ransomware

AHMAD O. ALMASHHADANI[1,2], MUSTAFA KAIIALI[1], (Senior Member, IEEE),SAKIR SEZER[1] AND PHILIP OKANE[1],

[1]Centre for Secure Information Technologies (CSIT), Queen's University Belfast, Belfast, UK
[2]Computer Center, University of Baghdad, Baghdad, Iraq

Corresponding author: Ahmad O. Almashhadani (e-mail: aalmashhadani01@qub.ac.uk).

ABSTRACT Ransomware is a type of advanced malware that has spread rapidly in recent years, causing significant financial losses for a wide range of victims including organizations, healthcare facilities, and individuals. Modern host-based detection methods require the host to be infected first in order to identify anomalies and detect the malware. By the time of infection, it can be too late as some of the system's assets would have been already exfiltrated or encrypted by the malware. Conversely, network-based methods can be effective in detecting ransomware attacks, as most ransomware families try to connect to command and control servers before their harmful payloads are executed. Therefore, a careful analysis of ransomware network traffic can be one of the key means for early detection. This paper demonstrates a comprehensive behavioral analysis of crypto ransomware network activities, taking Locky, one of the most serious families, as a case study. A dedicated testbed was built, and a set of valuable and informative network features were extracted and classified into multiple types. A network-based intrusion detection system was implemented, employing two independent classifiers working in parallel on different levels: packet and flow levels. Experimental evaluation of the proposed detection system demonstrates that it offers high detection accuracy, low false positive rate, valid extracted features, and is highly effective in tracking ransomware network activities.

INDEX TERMS Domain Generation Algorithm (DGA), Dynamic Malware Analysis, Locky, Machine learning, Malware Analysis, Ransomware

I. INTRODUCTION

With the increased use of computer systems and the growth of the Internet, cybercrime has also increased [1]. In the developed world, computing and internet use is ingrained in society, the economy and infrastructure. This complex mix has created a lucrative environment of potential victims, which can be preyed upon by cybercriminals. Cyber attacks are increasing daily, often fueled by evolving technology [2]. Ransomware has emerged as a major threat to individuals and businesses alike [3]. By encrypting data and files, it can disable computer systems to extort payment (ransom) to restore functionality [4]. Given the ubiquitous nature of computing, ransomware is often highly automated, thus enabling criminals to target a wide range of victims like industries, governmental agencies, and individuals [5].

Between 2015 and 2016, there was an increase in the number of ransomware families that went undetected as malicious. Researchers at Symantec observed an average of over 4,000 ransomware attacks per day in the first quarter of 2016, an increase of 300 percent in the attacks observed in 2015 [6]. SonicWall's annual threat report in 2017 shows a tenfold increase in the number of ransomware attack attempts during 2016, up to 266.5 million attempts by the end of Q4. Ransomware continues to be one of the most dangerous cyber-attacks in 2017 and 2018 [7] [8]. The highly connected world of the Internet (and cloud computing) facilitates the propagation of ransomware, including several communication protocols. These enable adversaries to launch numerous attack campaigns, such as the availability of botnets for hire, e.g., Necurs [5], the emergence of Ransomware-as-a-Service (RaaS) [7], and the TOR network [9]. Ransomware mainly targets Windows platforms, however they have begun to

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2907485, IEEE Access

target other platforms, including Apple, Android and Linux servers.

Ransomware can be categorized into several types, the most virulent and aggressive of which is crypto ransomware, which encrypts the victim's data, and locker ransomware, which locks the victim's computer, preventing users from accessing the system [10].

Crypto ransomware is not only capable of encrypting users' files, it attempts to encrypt any file located on both mapped and unmapped network drives, bringing a department or the entire organization to a halt if one system is infected [7]. Crypto ransomware is divided into three types, based on the employed cryptosystem [11] [12]:

- Symmetrical Cryptosystem Ransomware: Employs a symmetrical encryption algorithm, e.g., DES or AES to encrypt the victim's files, using the same key for encryption and decryption. This makes it plausible for the victim to recover the secret key by applying reverse engineering or memory scanning techniques. Fig. 1 (a) illustrates the network activities involved in this type of ransomware.
- Asymmetrical Cryptosystem Ransomware: In this type, a public key, embedded within the ransomware file or downloaded during the communication with the command and control (C&C) server, is used to encrypt the victim's file. As the private key is kept only with the attacker, it is impossible for the victim to obtain it without paying the ransom. However, this technique consumes more resources while encrypting the files. The corresponding network communications are depicted in Fig. 1 (b).
- Hybrid Cryptosystem Ransomware: This uses a dynamically generated symmetric key to encrypt the victim's files and a pre-loaded public key to encrypt the symmetric key itself, after clearing it out of memory. Most modern families of crypto ransomware use this technique to take the advantage of both types of encryption.

Most families of crypto ransomware try to connect to a C&C server after the target is infected and before the payload is executed. As such, network traffic analysis can yield significant results in detecting ransomware. There are few research papers in the literature about ransomware network-based analysis, however these studies are limited and inadequate to develop effective detection methods as outlined in the following section. This paper presents a network traffic analysis of crypto ransomware, extracts a set of various potential network features, and builds an effective multi-classifier intrusion detection system (IDS).

Locky (Ransom.Locky) is a hybrid cryptosystem ransomware family, which was released in February 2016 and was considered to be one of the major ransomware threats in 2017 by Symantec [5].It is mainly distributed by spam campaigns however, it can be distributed using exploit kits, e.g., Neutrino and Rig. After infection, it scans the victim's drives, including the network drives, for specific file types, e.g., .pdf, .doc, .jpg, etc. to encrypt them using AES and RSA algorithms [14]. One of Locky's key features is the use of Domain Generation Algorithm (DGA) [15] to find its C&C server. While being one of the most recent and prevalent ransomware families during 2017 [7] [8], Locky also relies heavily on external communications, making it ideal for a use-case study.

The paper is organized as follows: Section 2 discusses the related work, Section 3 describes the ransomware analysis environment, network traffic analysis is provided in Section 4, the multi-classifier IDS is presented in Section 5, and finally, Section 6 concludes the paper.

II. RELATED WORK

The current body of literature contains some work focusing on the analysis and detection of ransomware attacks based on network traffic. Cabaj et al. [16] analyzed CryptoWall ransomware dynamically in a dedicated environment using a honeypot and automatic run-time malware analytical system, Maltester. They identified some of the network activities of CryptoWall and presented practical results, i.e., identifying some proxy servers, the protocols used, and the hardcoded addresses of some servers. The authors used four virtual machines running on one physical server to perform the analysis. Moreover, the provided analysis is confined to the extraction of numerical values that are distinct to CryptoWall. It does not extract any generic behavioral feature that can be applied to the detection of other kinds of crypto ransomware.

Ahmadian et al. [12] proposed the Connection Monitor & Connection Breaker (CM&CB) framework to detect 'high survivable ransomwares' which employ Domain Generation Algorithm (DGA). The proposed method analyzes the requested domain names against a previously trained English language Markov-chain model. This work is built solely upon a single network feature. Further, the effectiveness of using a Markov-chain model against short texts, i.e., a domain name, is questionable and may cause high false positive rates. This method requires the building and maintenance of Markov-chain models for many non-English languages, which would be an expensive process. Aragorn Tseng et al. [17]] investigated different signatures for several ransomware families and proposed a deep learning based detection approach. They achieved 93.92% accuracy, although this was highly dependent on HTTP and TCP protocols.

Jones and Shashidhar [18] analysed the WannaCry ransomware in a Win32 environment using static and dynamic host-based analysis techniques. They also presented prototype software that can prevent WannaCry from releasing its payload. However, their research does not tackle the network activities of WannaCry with an aim to extract behavioural features, providing only limited information about its network communications.

Cabaj et al. [19] presented a dedicated Software-Defined Networking (SDN) based approach to detect and mitigate a ransomware attack, focusing on the analysis of HTTP traffic. The experimental results of the detection rate were of 97-98% accuracy with 1-5% false positives. However, this method

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2907485, IEEE Access

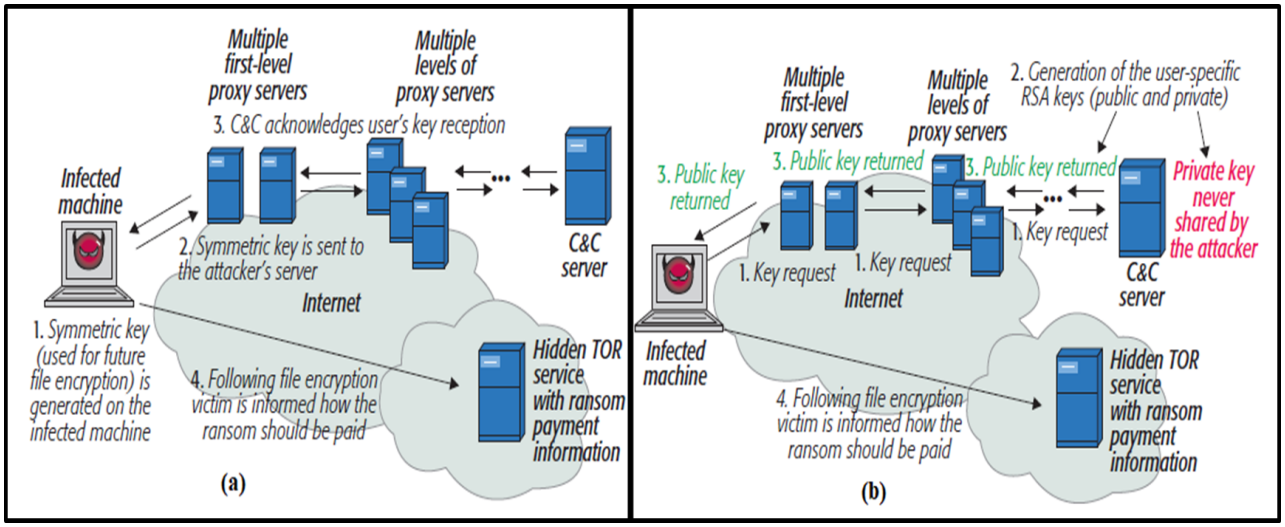


FIGURE 1: Crypto ransomware network communications [13] (a) Symmetrical encryption , (b) Asymmetrical encryption

is based only on the size of the data inserted by the victim into the outgoing sequences of the HTTP POST messages. Greg Cusack et al. [20] used a recent property of networking hardware called programmable forwarding engines (PFEs) to monitor the network traffic between an infected computer and the C&C server. Twenty-eight flow features were extracted from this traffic and they employed machine learning methods to detect the ransomware. However, this method presented moderate accuracy (86%) and high false negative rate (above 10%). Moreover, the detection was based on network traffic signatures and does not take the UDP protocol features into consideration.

The main gap in the literature is that there is little research focused on performing network-based analysis or detection of crypto ransomware [21] to extract potential behavioral features that can help to detect the attack before unleashing its payload. Conversely, research such as [22] [23] [24] [25] [26] only deal with extracting host-based features to detect the ransomware. In this paper, a thorough dynamic analysis of network traffic is carried out using Wireshark, and 18 network features from several protocols are extracted. A multi-classifier intrusion detection system with a new topology is proposed and implemented, using machine learning techniques to detect ransomware network activities. Moreover, it employs a language-independent algorithm to detect gibberish domain names based on general sonic axioms. The use-case study is composed of various experiments conducted on a dedicated test-bed using a public dataset. The experimental results clearly confirm that the approach proposed in this paper significantly enhances the detection of ransomware communications.

III. SETTING UP THE ANALYSIS ENVIRONMENT

This section describes the preparation of an analysis environment to run and analyze the network activities of Locky ransomware.

A. MALWARE TESTBED ENVIRONMENT

Dynamic analysis of ransomware requires a safe environment to execute and monitor the ransomware's network activities. We have built a dedicated testbed that consists of three real computers and two virtual machines as depicted in Fig. 2. The purpose of this testbed is to execute samples of ransomware and capture their network traffic. These PCAP files are then analyzed to extract a set of network features that describe the communication behavior between the ransomware and its C&C server (i.e., attacker). The details of the testbed's components are summarized below:

- PC1 is used as a victim machine, where the ransomware is injected.
- PC2 and its hosted virtual machines VPC1 and VPC2, are three clean machines on the network representing the non-infected nodes. Analyzing their network traffic allows investigation of how ransomware tries to propagate over a network.
- PC3 captures the network traffic and stores it in PCAP files using Wireshark for further analysis. For that, its NIC is set to work in promiscuous mode. As it is important to keep PC3 protected from the ransomware's propagation attempts, Ubuntu Linux was used as the OS, as the selected ransomware samples do not infect Linux systems. Furthermore, we have setup a firewall with a strict rule to drop any packet targeting PC3 explicitly.
- This testbed is isolated from the university's main campus network to protect the main network from the ransomware propagation attempts. All of the testbed machines are connected to the Internet, as often required by the ransomware to communicate with its C&C server. The external ISP providing the dedicated Internet connection to the experimental-lab was informed in advance about an infected PC with ransomware on their

public network.

B. RANSOMWARE DATASET

Having a rich dataset is crucial in allowing accurate analysis and the extraction of informative network features. The acute shortage in ransomware datasets [21] was a key challenge in this research work. We have examined several benchmark datasets such as [27] [28]:

- KDDCUP 99
- NSL-KDD (2009)
- UNSW-NB15 (2015)
- Canadian Institute for Cybersecurity (CIC) datasets

Unfortunately, these datasets do not include records of any ransomware attack, although the first attack was discovered in 1989 [29]. Therefore, we built our own dedicated testbed to create a new dataset taking the Locky ransomware as a case study as outlined above.

However, we have observed that our testbed-generated dataset only contains unidirectional traffic, i.e., the traffic originated from Locky (at the victim's end) attempting to find the C&C server, without any response traffic originating from the attackers' side. This can be due to the likelihood of the C&C servers being down after the campaign was over, their IP addresses were blacklisted, or the attacker being able to recognize our environment as a testbed, rather than a real victim.

Therefore, we have carried out an extensive search to find a reliable dataset containing bidirectional traffic captured when the campaign was still active. We have found a dataset created within a research center in the Czech Technical University (CTU), the Malware Capture Facility Project (MCFP) dataset [30]. In this project, real malware was executed within a real network environment for up to several weeks, around the first appearance of Locky. It also contains a significant amount of benign traffic, the size of which is much larger than the malicious traffic, as it contains all traffic generated by the different benign services running on the system. Conversely, the malicious dataset contains the traffic generated only by Locky, which deliberately tries to send a small amount of traffic over time in order to avoid detection by IDS systems. Table 1 shows the details of the MCFP portion used in this research.

C. LOCKY RANSOMWARE SAMPLE COLLECTION

Locky samples have been collected from the following open source communities:

- virusshare.com
- malware-traffic-analysis.net

The hash values and types of the samples were checked using virustotal.com to ensure that the samples are valid and have the correct labels.

IV. LOCKY NETWORK TRAFFIC ANALYSIS AND FEATURES EXTRACTION

In this section, we have analyzed the network traffic of Locky ransomware to extract a number of informative network features. Locky often spreads over spam emails that contain malicious attachments in the form of macro-enabled office documents. This attachment runs a script, called a downloader, to download Locky's executable file from a URL and install it on the victim's system. When fully installed, Locky tries to find and contact its C&C server(s) to exchange encryption keys and execute the intended malicious actions via one of the following strategies:

- It uses an encrypted list of hardcoded IP addresses to establish a TCP session with its C&C server(s).
- If these hardcoded addresses are all unreachable, i.e., blacklisted, or the session was disrupted, Locky tries to find its C&C server(s) by running the Domain Generation Algorithm (DGA) that periodically generates a large number of pseudo-random domain names. Locky keeps sending DNS requests about these names until the actual C&C server(s) is found.
- If the DGA method also fails, Locky uses the NetBIOS Name Service (NBNS) protocol hoping to find a previously infected machine on the local network that has already resolved the C&C server name.

If Locky successfully establishes a TCP session with any of its C&C servers, the attacker can use it to govern the payload execution. Conversely, Locky uses the HTTP post request method to send information back to the attacker. As stated above, Locky uses a wide range of network protocols that can be useful to extract potential network features. Table 2 shows the DNS, NBNS, and HTTP protocols' statistics within the collected PCAP files.

Using Python scripts and MATLAB, the PCAP files are investigated to extract informative network-based features. The analysis shows that identified features can be classified based on two aspects producing four different types of features: the behavioral and the non-behavioral features, and the detectable and the non-detectable features.

A behavioral feature is a feature that cannot be suppressed by any future malware variant, if it is built upon the same underlying technology, e.g., DGA. While a non-behavioral feature is a feature that appears within the traffic being analyzed, however, its absence is conceivable in future variants even if they use the same underlying technology.

A detectable feature is a feature of the malicious traffic that can still be measured in the mixed traffic, i.e., the traffic of an infected machine where both the benign and the malicious streams coexist. A non-detectable feature is one that appears in the malicious traffic alone, however, the malware was able to obfuscate it within the normal traffic stream so that it becomes indistinguishable. The following subsections illustrate these features in greater detail, supported by experiments and statistics.

A. TCP TRAFFIC ANALYSIS

1) Reset connections

It can be observed that there are numerous TCP reset (RST, ACK) packets in Locky's traffic used to terminate the malicious TCP connections abnormally. They are recognizable

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2907485, IEEE Access

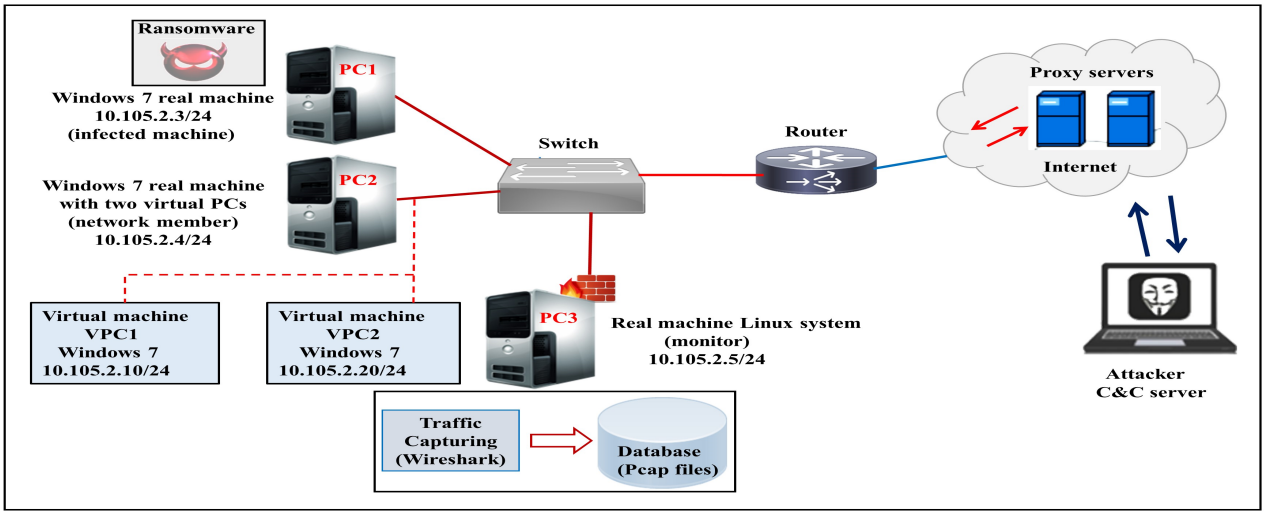


FIGURE 2: The testbed environment

TABLE 1: MCFP collected Locky and benign PCAP files

No.	MCFP ID	File type	Size (KB)	Hash value
1	CTU-Malware-Capture-Botnet-183-1	Locky ransomware	1097	eda7667e1081397ba43ccc5d6ba6d76434e05403
2	CTU-Malware-Capture-Botnet-214-1	Locky ransomware	265636	2ddd654437a48974f241f81a9d645a7374b82bec
3	CTU-Malware-Capture-Botnet-221-1	Locky ransomware	299	d99e90f8fdc5a47bbc7efa9caf8aefdc718cfcbe
4	CTU-Malware-Capture-Botnet-236-1	Locky ransomware	176	8856cc5561a0598340f8be3de37ac39ac5e76e2d
5	CTU-Normal-13	benign	495	N/A
6	CTU-Normal-20	benign	275797	N/A
7	CTU-Normal-23	benign	287794	N/A
8	CTU-Normal-24	benign	206399	N/A
9	CTU-Normal-25	benign	493700	N/A
10	CTU-Normal-26	benign	115669	N/A
11	CTU-Normal-28	benign	140264	N/A
12	CTU-Normal-29	benign	446690	N/A
13	CTU-Normal-30	benign	852486	N/A
14	CTU-Normal-32	benign	842056	N/A

TABLE 2: Network protocols' statistics

Protocol Name	Percentage (%)
HTTP	3.77
DNS	7.25
NBNS	6.05

when compared to the regular traffic. Fig. 3 shows the ratio of reset connections in the overall malicious traffic compared to that of the benign traffic.

The reset connection ratio (%RSTConn) attribute, appears to be an informative feature. However, after observing the time-based rate of the malicious (RST, ACK) packets, they are found to be well distributed over time. This allows the malicious packets to blend in with the normal (RST, ACK) packets without a significant increase in the time-based rate. In other words, it appears that the malware intentionally does not create a lot of connections within a short period of time to avoid detection. Fig. 4 illustrates this by comparing the frequency of the malicious reset packets alongside the normal reset packets every minute. It is apparent that this feature is not easily detectable when the mixed stream of packets is monitored as a whole, i.e., the benign and the malicious streams.

Computing the IP-wise reset connections ratio, (%RSTConnIPwise), can give a distinguishable result, in that, connections with the remote attack servers are expected to preserve the same reset ratio found in the overall malicious traffic, i.e., about 99%. Surprisingly, we found that in most time-frames, e.g., a 15-minute time-frame, there are few

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2907485, IEEE Access

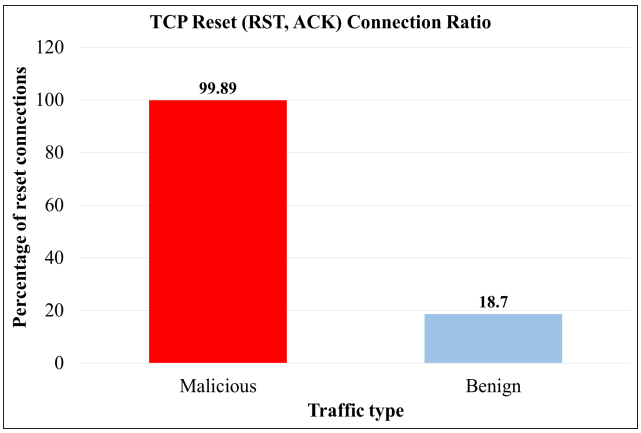


FIGURE 3: The reset connection ratio

connections in the normal stream with a set of destination IP addresses that have a similar reset ratio to that of the malicious stream making this feature a non-detectable feature also. Fig.5 shows an example 15-minute time-frame. It can be observed that there are few benign IP addresses (Fig. 5 (b)) that have a reset connection ratio as high as that of the malicious IP addresses (Fig.5 (a)).

However, these sets of IP addresses in the normal stream vary across different time-frames, while the same set of IP addresses repeat in the malicious stream. Fig. 6(b) shows how the set of benign IP addresses taken in a subsequent 15-minute time-frame is different from that shown in Fig. 5(b), while subsequent time-frames share the same set of malicious IP addresses for the malicious traffic, see Fig. 5(a) and Fig. 6 (a). This led us to define a detectable feature of having the same set of IP addresses with a high reset connection ratio repeated almost every time-frame (%RSTConnRepIPSet).

On the other hand, these RST-based features are all classified as non-behavioral features, since seemingly nothing can prevent future variants of Locky from terminating their connections normally, i.e., using TCP FIN without RST.

2) HTTP traffic

Attackers are continually building robust C&C infrastructures and devising stealthy communication channels for their malware to bypass intrusion detection systems. Therefore, ransomware uses the common ports that are usually open for normal traffic, e.g., ports 80 and 443, to connect with their intermediate proxy server(s) (Fig. 1). After analyzing the captured PCAP files generated by Locky, it can be observed that it was employing Trojan-like behavior. It creates a short TCP session with a remote server, where it uses a HTTP post method to send data to that server. Locky recreates a new connection with the same hardcoded address after a period of time repeating the previous activities (Fig. 7).

It is observable that, unlike the normal traffic, the majority of Locky's HTTP requests were POST methods. Fig. 8 represents the POST methods ratio found in the malicious traffic compared to that of the normal traffic with respect to other

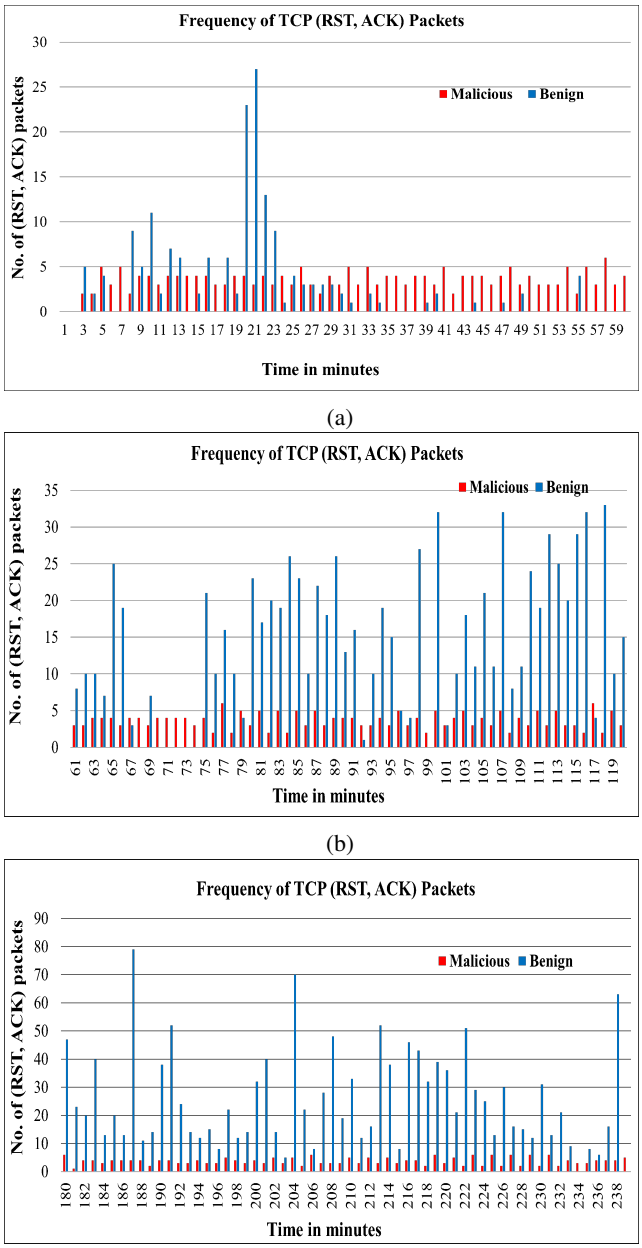
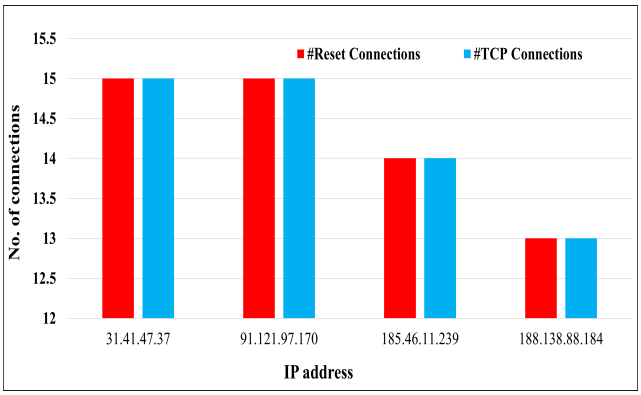


FIGURE 4: The frequency of the TCP reset packets (a) within the 1^{st} hour (b) within the 2^{nd} hour (c)within the 3^{rd} hour

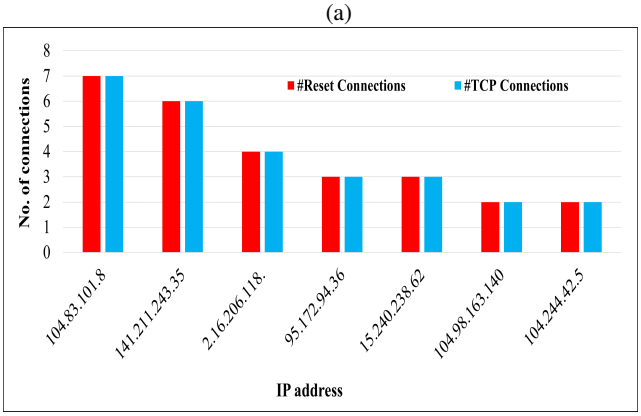
HTTP request methods, e.g., GET method.

This attribute (#HTTP-POSTs) appears to be an informative feature, however the time-based frequency of this attribute needs investigation to determine if it is a detectable or non-detectable feature. Fig. 9 shows the frequency of HTTP-POSTs taken over 15 subsequent time-frames, each of 15 minutes long. It is obvious that Locky significantly increases the number of HTTP-POSTs within the traffic stream, compared to the normal traffic. Thus, #HTTP-POSTs is classified as a detectable feature. It is further classified as a behavioral feature, considering that Locky, as a Trojan, is

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2907485, IEEE Access

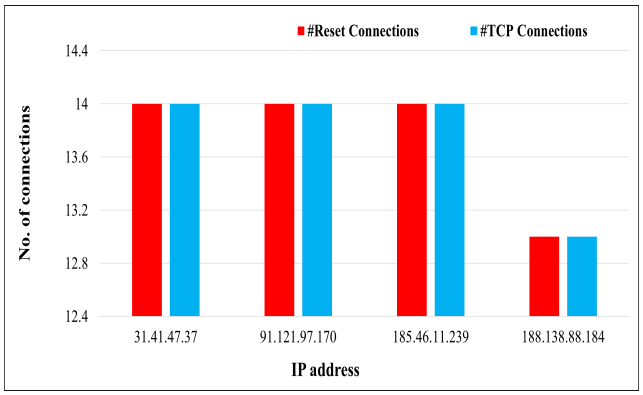


(a)

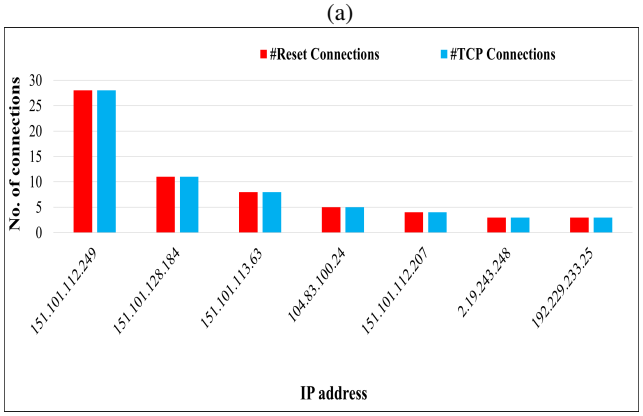


(b)

FIGURE 5: Samples of the IP-wise reset connections ratio (a) malicious samples (b) benign samples



(a)



(b)

FIGURE 6: Samples of the IP-wise reset connections ratio in a subsequent time-frame (a) malicious samples (b) benign samples

expected to POST information to its remote server with a higher frequency than GET requests.

Most of the collected Locky variants use POST methods without specifying the User-Agent, whereas there was no POST method found without a User-Agent in the normal traffic. Therefore, another detectable feature can be defined as Nil-User-Agent. However, this feature is not applicable to all Locky variants, as several were found to be impersonating the Mozilla/4.0 user agent. Therefore, it is classified as a non-behavioral feature.

B. DNS TRAFFIC ANALYSIS

DNS has been widely used by cyber attackers for several purposes:

- Avoiding the need for a hard-coded list of servers' IP addresses to be injected within the malware file, as they can be easily discovered and blacklisted. Instead, DGA is used to generate pseudo-random domain names for which DNS requests are continuously generated until the actual server is located [31].
- Sending instructions to the malware by bringing a particular domain name up or down. The best example for this use case is the Kill Switch domain name of the WannaCry ransomware [22]. If WannaCry finds this domain name up, it will immediately terminate the attack.
- DNS can sometimes be used to transfer data out of a secure network [32] where no outbound TCP connectivity is allowed. UDP is also limited to allow DNS queries only. So, the malware can divide the file data into several DNS requests sent to a fake DNS server residing at the attacker's end. These DNS requests are parsed by the fake DNS server to extract the data chunks and concatenate them, reforming the original data.

Based on this knowledge, we have analyzed the DNS packets of the collected PCAP files and extracted some informative features as illustrated in the following subsections.

1) DNS name error

Numerous DNS name error packets were observed in the Locky traffic. This is due to the use of the DGA algorithm that generates lots of pseudo-random domain names, for which Locky has to send the corresponding DNS requests until the server is found. Fig. 10 displays samples of these name errors found in the Locky's traffic. Fig. 11 shows the statistics of the number of DNS name errors found in the malicious traffic compared to that of the benign traffic.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2907485, IEEE Access

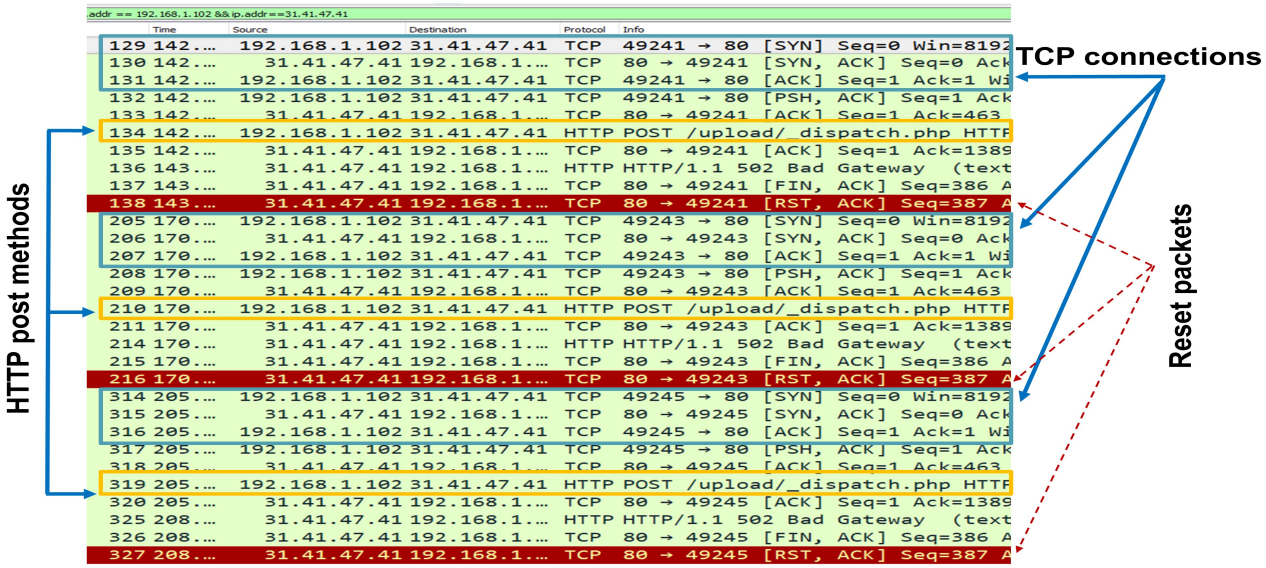


FIGURE 7: Locky's malicious channels

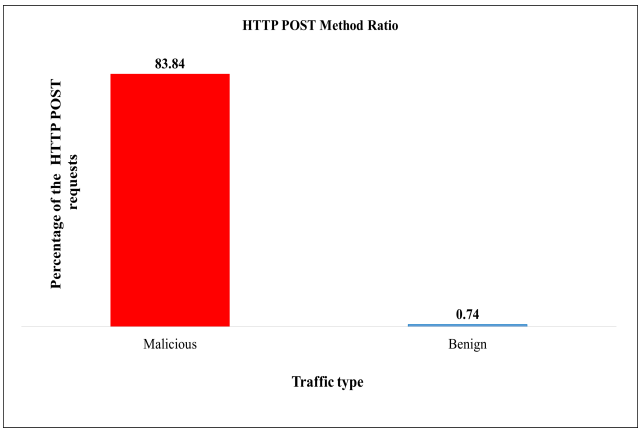


FIGURE 8: HTTP POST method ratio Malicious Vs Benign

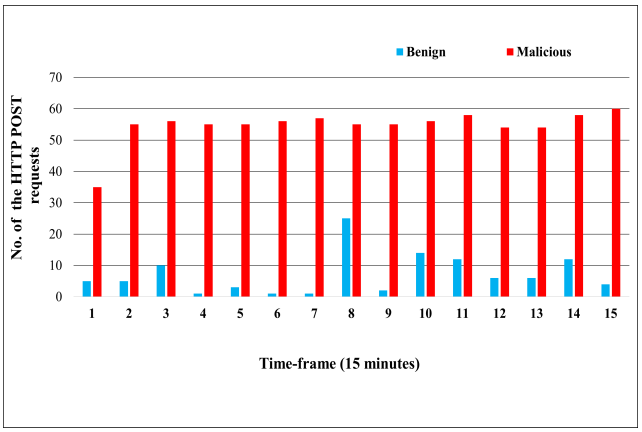


FIGURE 9: HTTP POST method frequencies taken over subsequent time-frames

This significant difference between the benign and malicious streams may produce an informative feature, based on the number of DNS name errors (#DNS-NE). However, by applying the same logic discussed earlier in the previous section, it is also important to analyze the time-based value distribution of this feature. For that, the frequency of the DNS name errors is computed over various time-frames as depicted in Fig. 12. It is obvious that the malware produces a large volume of DNS name errors, which are detectable even within the mixed traffic. We believe that any future variant of the Locky ransomware cannot suppress this feature as far as it is still based on the same underlying technology, i.e., DGA. For that we have classified this feature (#DNS-NE) to be a behavioral feature.

On the other hand, we believe that any future variant of the Locky ransomware cannot suppress this feature as far as it is still based on the same underlying technology, i.e., DGA. For that we have classified this feature (#DNS-NE) to be a behavioral feature.

2) Meaningless domain name

As stated previously, Locky sends a large amount of DNS requests regarding the domain names generated by DGA. These names are pseudo-random, in that they are incomprehensible to humans. Table 3 shows some examples of these meaningless domain names found within Locky traffic. Such ambiguous names were absent in the normal traffic, which makes this feature ((Meaningless Domain Name (MDN)) detectable. However, it requires an algorithm that can recognize such names accurately.

It is also apparent that Locky tries to resolve these names several times in case of failure. Fig. 13 shows the top frequencies of the DNS queries sent by a Locky sample for different

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2907485, IEEE Access

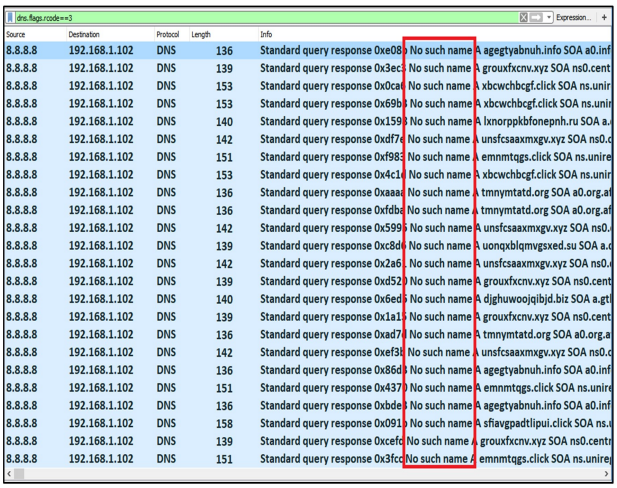


FIGURE 10: Samples of the name errors within the Locky's traffic

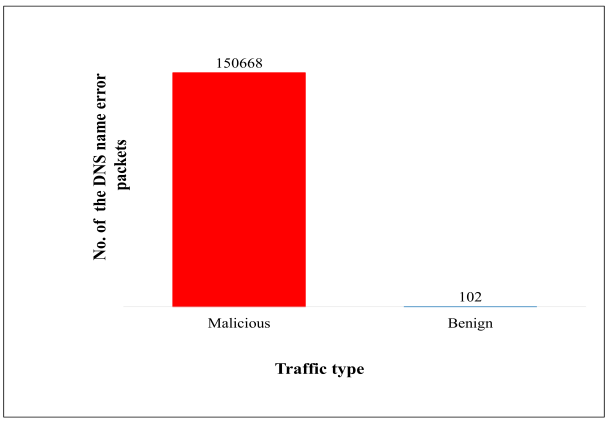


FIGURE 11: Number of the DNS name errors in the malicious vs the benign traffic

meaningless domain names. Similar to the #DNS-NE feature, this feature (MDN) is classified as a behavioral feature.

Identifying the meaningless domain names was not a straightforward task. A combination of two algorithms was adopted. The first one uses the Shannon Entropy metric to measure the distribution randomness of the domain name [33]. It can be calculated using the following equation (1):

$$H = -\Sigma_{i=1}^{n} p_i log p_i \quad (1)$$

where p_i is probability distribution $p_n = (p_1, ..., p_n$ with $p_i \geq 0$ for $i = 1, ..., n$ and $\Sigma_{i=1}^{n} p_i = 1$

The entropy was only calculated for the first label of every domain name, as the second label usually represents the root DNS servers, i.e., org, com, net, etc. Table 4 shows selected examples of domain names and their calculated entropy values.

As depicted in Table 4, entropy alone may not be enough to measure the randomness of a domain name. For that, we

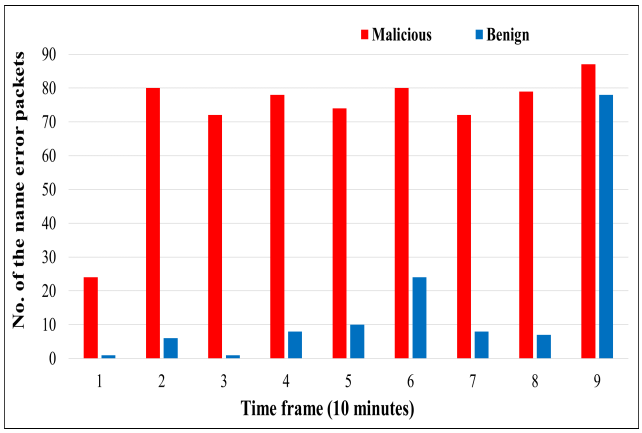


FIGURE 12: DNS name error frequencies taken over subsequent time-frames

TABLE 3: Examples of the meaningless domain names

No.	Domain name	Domain length
1	lxnorppkbfonepnh.ru	19
2	unsfcsaaxmxgv.xyz	17
3	sfiavgpadtlipui.click	21
4	oaedvogymkqaivswf.xyz	21
5	qpxyvqkebnhosrs.pw	18
6	xakckpsovshuyxl.biz	19
7	lussbshcfbhqypv.info	20
8	fvxmecnhgihtd.org	17
9	hldorpxdaaxshby.nl	18
10	kwijpuwsviahhs.ru	17
11	mcyjduv.pm	10

have written an auxiliary Python function, named unpronounceable. This helps to determine the randomness of a domain name, based on the fact that it is unusual to observe several subsequent constant or vowel letters within a normal domain name. This function returns the maximum number of sequential consonant and vowel letters in the first label of the domain name as illustrated in Table 4. For example, the maximum numbers of the sequential consonant and vowel letters for the domain name grouxfxcnv.xyz are 6 and 2 respectively. Generally, the letter 'y' can be regarded as a vowel or a consonant, however, the experiments conducted on our dataset show that considering 'y' as a vowel improves the accuracy. Fig. 14 illustrates our algorithm that combines the previously discussed two methods, i.e., the entropy and unpronounceable, to determine the randomness of a domain name. The threshold values for entropy and unpronounceable were determined experimentally to obtain high accuracy.

3) Further features of DNS

Having several DNS-based behavioral and detectable features prompted us to investigate more DNS features using machine learning. For that, a total of 9 DNS raw features were extracted and fed into a feature selection engine using

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2907485, IEEE Access

TABLE 4: Examples of the entropy, max. no. of sequential consonants and vowels computed for some domain names

Domain name	Domain name Type	Entropy	Max. no. of sequential consonants letters	Max. no. of sequential vowel letters
dnfyupcnwghbqao.be	malicious	3.7736	8	2
qkbwinwgdvbufu.de	malicious	3.3788	6	1
unsfcsaaxmxgv.xyz	malicious	3.2389	5	2
uonqxblqmvgsxed.su	malicious	3.6402	11	2
ggvuecpdljaidm.uk	malicious	3.5216	5	2
tiles.services.mozilla.com	normal	2.3219	1	1
apis.google.com	normal	2	1	1
platformdl.adobe.com	normal	3.1219	4	1
platform-eb.twitter.com	normal	3.4594	2	1
connect.facebook.net	normal	2.2359	2	1

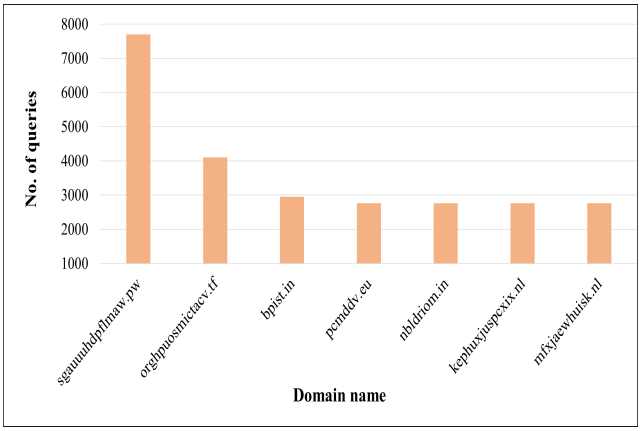


FIGURE 13: The top frequencies of meaningless DNS domain names

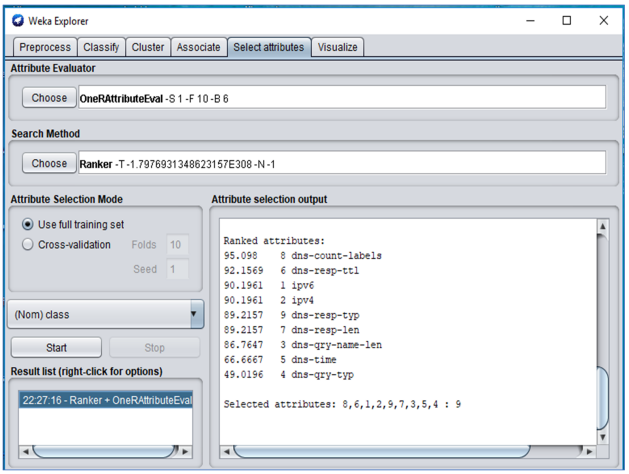


FIGURE 15: Attributes selection using Weka

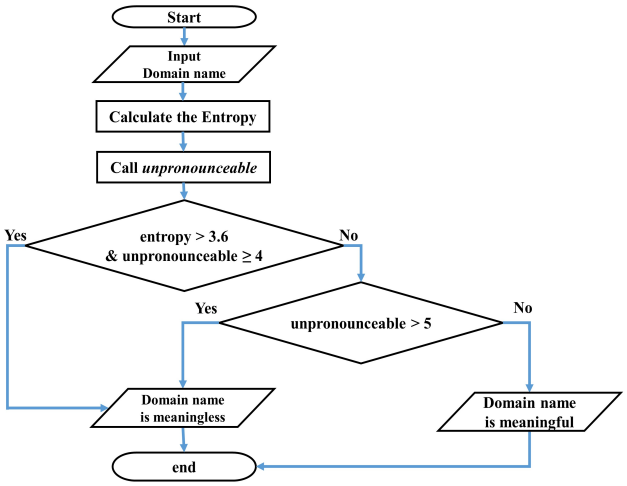


FIGURE 14: Domain name randomness measuring algorithm

Weka, a popular machine learning tool [34], following the configurations shown in Fig. 15. Table 5 shows a list of the ranked attributes extracted by Weka.

We have analyzed the PCAP files against the first attribute in table 5 i.e., dns-count-labels. It is found that Locky commonly uses DNS names of only two labels. Benign traffic uses DNS names with several labels' counts distributed mostly around 3 and 4, see Fig. 16. Therefore, a feature of the DNS labels' count ratio (%dns-count-labels) can be derived to distinguish between the benign and malicious traffic, based on the labels' count as suggested by Weka. This feature is also classified as a detectable and non-behavioral feature, as nothing prevents future variants of Locky from using DNS names with several labels' counts as that of the benign traffic.

C. NETBIOS NAME SERVICE (NBNS)

Network Basic Input / Output System (NetBIOS) is an OSI session protocol that allows the host computers to communicate over a local network and the Internet. It provides a connection (session) and connectionless (datagram) services and supports broadcast and multicast communications. NetBIOS name service (NBNS) is a part of the NetBIOS-over-TCP protocol suite and is similar to DNS, in that it maps NetBIOS names to IP addresses. Typically, NBNS uses UDP port 137 and can also use TCP port 137 [32].

We have observed from Table 2 that NBNS packets form 6.05% of the total malicious traffic, while they were almost

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2907485, IEEE Access

TABLE 5: The ranked attributes by Weka

No.	Attribute name	Rank	Description
1	dns-count-labels	1	The number of labels of a DNS query name
2	dns-resp-ttl	2	Time to live of the DNS response packet
3	dns-ipv6	3	Communication with the DNS server using IPv6 address
4	dns-ipv4	3	Communication with the DNS server using IPv4 address
5	dns-resp-typ	4	DNS response packet type
6	dns-resp-len	5	The length of the DNS response packet
7	dns-qry-name-len	6	The length of a DNS query name
8	dns-time	7	The time between the DNS query and response packets
9	dns-qry-typ	8	DNS query type

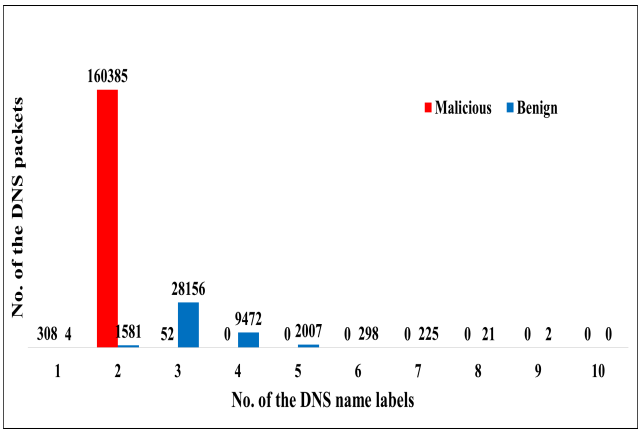


FIGURE 16: The DNS labels' count: Malicious Vs Benign

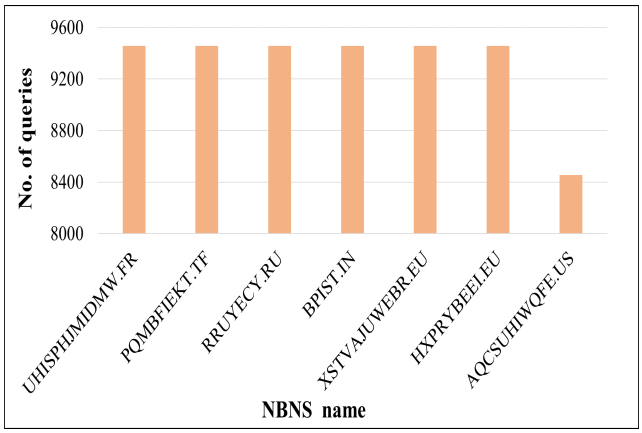


FIGURE 17: The top frequencies of NBNS queries within the malicious traffic

absent from the benign traffic. It is also evident that when Locky receives a DNS name error packet about a particular domain name, it tries to resolve this name again using NBNS, expecting other pre-infected nodes on the network to have an answer for it. Fig. 17 shows the statistics for a number of suspicious NetBIOS queries found within the malicious stream.

Similar to the DNS features, this feature (MNBNS) is classified to be detectable and behavioral. Table 6 summarizes all the extracted network features classified accordingly. Those who fall into the Behavioral and Detectable categories are considered to be class I features. Conversely, the features can be further classified into being packet-level vs flow-level and Simple (easy to measure) Vs Complex as depicted in Table 7. Class I features that were also classified as Simple are the most favorable ones.

TABLE 6: Summary of the extracted features classified into: Behavioral vs Non-behavioral & Detectable vs Non-detectable

	Behavioral	Non-behavioral
Detectable	#HTTP-POSTs, DNS-NE, #DNS-NE, MDN, #MDN, MNBNS	%RSTConnRepIPSet, Nil-User-Agent, dns-ipv6, dns-ipv4, %dns-count-labels, dns-count-labels, dns-resp-len, dns-resp-ttl, dns-resp-typ, dns-qry-typ, dns-qry-name-len, dns-time
Non-detectable	–	%RSTConn, %RSTConnIPwise

V. BUILDING A MULTI-CLASSIFIER INTRUSION DETECTION SYSTEM

In this section, we present a multi-classifier intrusion detection system, which is specifically designed to track ransomware network activities based on the obtained features. The system has two independent binary classifiers: packet-based classifier (C_1)) and flow-based classifier (C_2), working in parallel to detect the packet-level and the flow-level features depicted earlier in Table 7. Fig. 18 shows the architecture of the multi-classifier detection system. It consists of two main modules: The Feature Extraction Module and the Decision Making Module.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2907485, IEEE Access

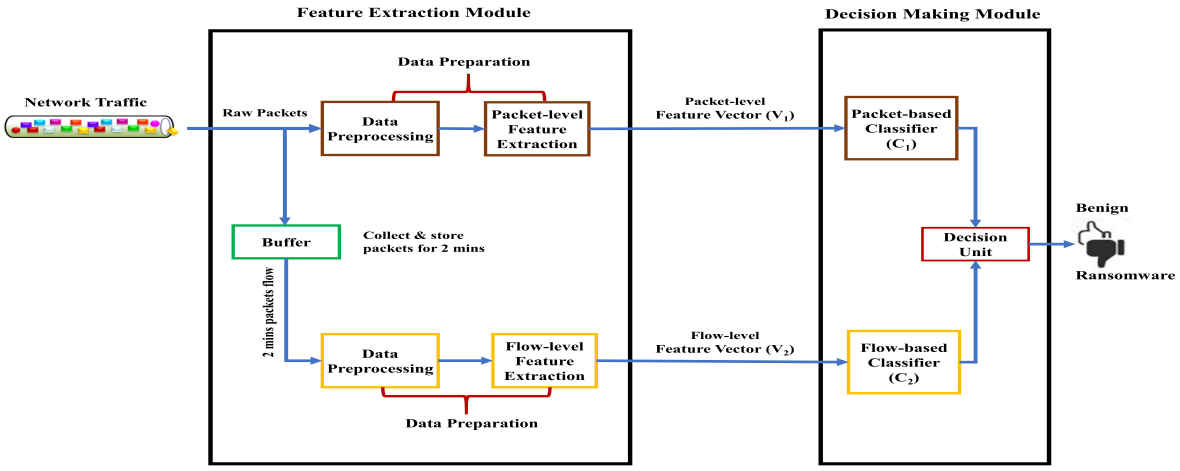


FIGURE 18: The architecture of the multi-classifier detection system

TABLE 7: Summary of the extracted features classified into: Packet-level vs Flow-level & Simple vs Complex

	Packet-level	Flow-level
Simple	Nil-User-Agent, DNS-NE, dns-count-labels dns-ipv6, dns-ipv4, dns-resp-len, dns-resp-ttl, dns-resp-typ, dns-qry-typ, dns-qry-name-len, dns-time	%RSTConn, %RSTConnIPwise, %RSTConnRepIPSet, #HTTP-POSTs, #DNS-NE, %dns-count-labels
Complex	MDN, MNBNS	#MDN

TABLE 8: The selected packet-level and flow-level features

Type	Features
Packet-level	MDN
	DNS-NE
	MNBNS
	dns-ipv6
	dns-ipv4
	dns-count-labels
	dns-resp-len
	dns-resp-ttl
	dns-resp-typ
	dns-qry-typ
	dns-qry-name-len
	dns-time
Flow-level	#MDN
	#DNS-NE
	#HTTP-POSTs
	%RSTConnRepIPSet

The incoming packets are sent to the packet-level data preparation unit to extract the packet-level feature vector (V_1). In addition, a buffer unit is used to collect and store the incoming packets temporarily. Every two minutes, the buffer's data is forwarded to the flow-level data preparation unit to extract the flow-level feature vector (V_2). The two vectors V_1 and V_2 are sent to classifier C_1 and C_2 respectively to detect any ransomware network activity. Finally, a simple OR-based decision unit is used to trigger an alarm if any of the classifiers detects malicious activity.

A. DATA PREPARATION

The data preprocessing unit accepts the raw packet data as an input, identifying and handling any missing and outlier values, then prepares it for feature extraction. For example, it extracts the domain name out of the incoming packet and calculates its entropy value. The feature extraction unit accepts the data preprocessing output and extracts the corresponding features accordingly. For example, it runs the algorithm depicted in Fig. 14 to determine the randomness of the extracted domain name. As discussed earlier, we have two levels of data preparation, the packet-level and the flow-level. Table 8 represents the selected features for our experiments from Table 7. The data preprocessing and feature extraction units were built in Python, using the libraries dpkt, socket, numpy, and csv. Each feature extraction unit prepares the feature vector for the corresponding classifier in a csv format.

B. MODELS BUILDING

The raw PCAP files shown in Table 1 were used to build the packet-based and flow-based labeled datasets to train and test classifiers C_1 and C_2 respectively as depicted in Fig. 19. Then, each dataset was randomly divided into two disjointed subsets: the training and the testing datasets. Since there is no deterministic rule to specify the percentage of the training and testing datasets [35], we have adopted the percentages shown in Table 9.

Weka tool was used to train the classifiers on the training datasets using the cross-validation option that can both

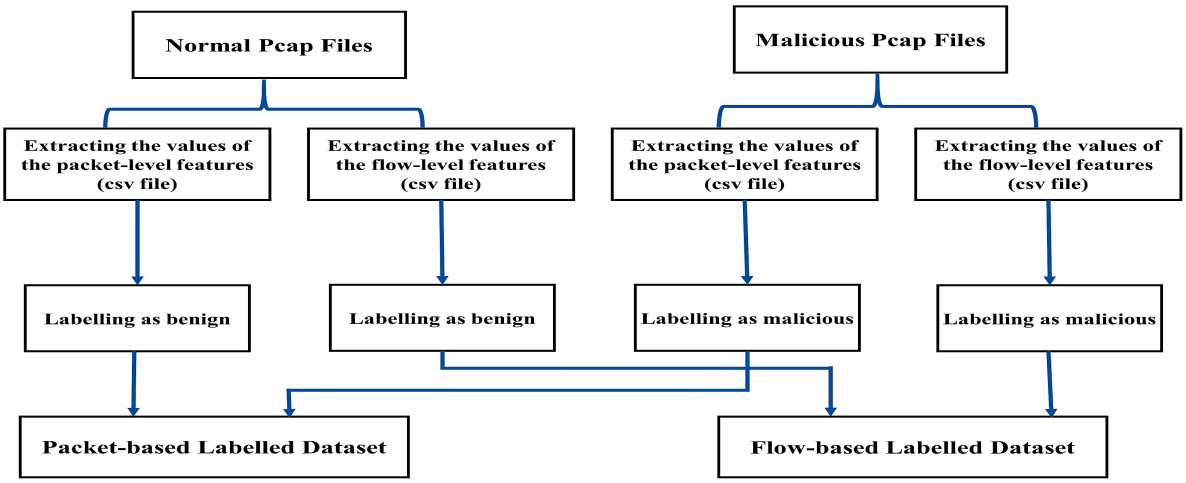


FIGURE 19: Building labelled datasets out of the raw PCAP files

TABLE 9: Percentages of the training and testing datasets

Type	Training dataset	Testing dataset
Packet-based	70%	30%
Flow-based	85%	15%

TABLE 10: Confusion matrix

	Predicted (ransomware)	**Predicted (benign)**
Actual (ransomware)	True Positive (TP)	False Negative (FN)
Actual (benign)	False Positive (FP)	True Negative (TN)

validate the model and detect overfitting. Although applying the cross validation procedure is considered enough [36] to estimate the performance of the model, however, it is also recommended to evaluate the trained model using the testing dataset which was not seen by the model during the learning process.

Several popular classification learning algorithms such as Random Forest, LibSVM, Bayes Net, and Random Tree were selected to build each classifier C_1 and C_2 autonomously. Finally, the algorithm that provides the highest accuracy was adopted by each classifier accordingly.

C. MODELS EVALUATION

Several classification metrics [37] [38], i.e., accuracy, False Positive Rate (FPR), precision, recall and F1 score, were used to evaluate the performance of each model. These metrics are derived from the confusion matrix (Table 10) as indicated by equations (2), (3), (4), (5), and (6) respectively.

Where:

True positive (TP): Number of ransomware samples that were classified as ransomware (correct prediction). False Positive (FP): Number of benign samples that were classified as ransomware (incorrect prediction). True negative (TN): Number of benign samples that were classified as benign (correct prediction). False Negative (FN): Number of ransomware samples that were classified as Benign (incorrect prediction).

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{2}$$

$$FPR = \frac{FP}{FP + TN} \tag{3}$$

$$Precision = \frac{TP}{TP + FP} \tag{4}$$

$$Recall = \frac{TP}{TP + FN} \tag{5}$$

$$F1score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{6}$$

Tables 11 and 12 indicate the evaluation results for the packet-based and the flow-based models respectively. Fig. 20 and Fig. 21 show the values of some performance metrics. It is observed that for the selected dataset, classifier C_1 produces the highest performance when it uses the Random Tree algorithm, while classifier C_2 produces the highest performance when it uses Bayes Net algorithm.

VI. CONCLUSION

Building upon the state-of-the-art research on network-based ransomware detection, this paper provides a thorough analysis of crypto ransomware network traffic and proposes an advanced ransomware detection method, using Locky as a case study. A dedicated testbed environment was built and PCAP files were captured and analyzed. Furthermore, Locky's PCAP files of the MCFP dataset were also collected and analyzed carefully. The analysis indicates that Locky has several network activities that can be used to extract potential

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2907485, IEEE Access

TABLE 11: Packet-based classifier evaluation

Dataset Type	Algorithm name	Accuracy (cross validation 10 folds)	Accuracy (testing)	FPR	Precision	Recall	F1 score
Packet-based	Random Forest	97.45	95.83	0.042	0.962	0.958	0.958
	LibSVM	96.15	91.67	0.083	0.917	0.917	0.917
	Bayes Net	95.51	93.75	0.063	0.938	0.938	0.937
	Random Tree	**98.72**	**97.92**	**0.021**	**0.979**	**0.979**	**0.979**

TABLE 12: Flow-based classifier evaluation

Dataset Type	Algorithm name	Accuracy (cross validation 10 folds)	Accuracy (testing)	FPR	Precision	Recall	F1 score
Flow-based	Random Forest	99.74	96.49	0.035	0.965	0.965	0.965
	LibSVM	99.56	94.74	0.053	0.947	0.947	0.947
	Bayes Net	**99.83**	**97.08**	**0.029**	**0.972**	**0.971**	**0.971**
	Random Tree	99.74	95.91	0.041	0.959	0.959	0.959

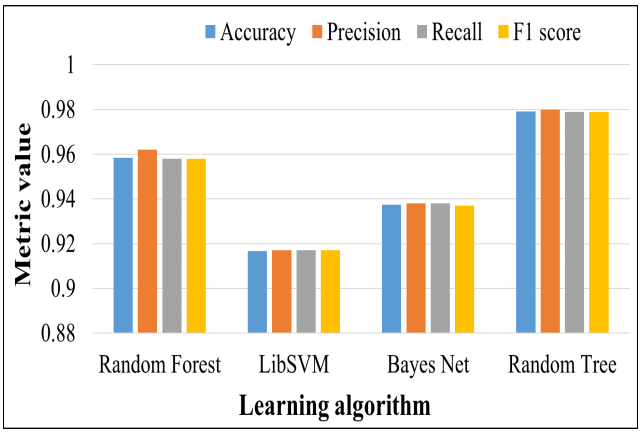


FIGURE 20: Performance metrics of each algorithm for the packet-based classifier

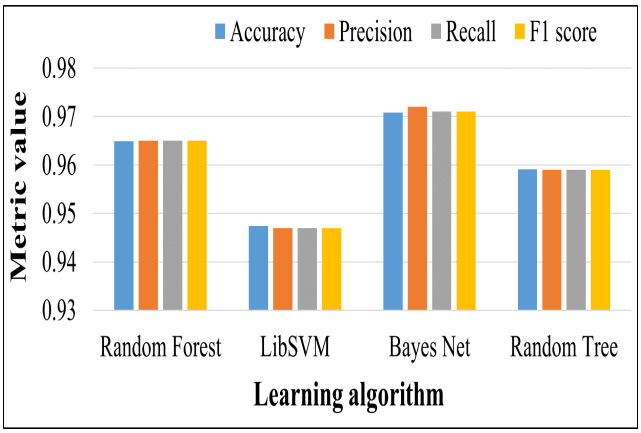


FIGURE 21: Performance metrics of each algorithm for the flow-based classifier

behavioral features. A total of 18 features were extracted using TCP, HTTP, DNS, and NBNS traffic. These features are informative and can clearly differentiate traffic generated by a compromised host.

Moreover, a multi-classifier network-based ransomware detection method is proposed and prototyped, working on two different levels: the packet-level and the flow-level. Detailed experimental analyses clearly demonstrate a high detection accuracy for each level: 97.92% and 97.08% respectively validating the effectiveness of the extracted features. Our discovery of these new features significantly contributes to the field of research in network-based malware detection, providing a strong foundation for the advancements of next generation network-based malware detection systems.

ACKNOWLEDGMENT

I would like to express my special thanks of gratitude to University of Baghdad which supported me during this research work. Also, I like to express my sincere gratitude to Dr. Harsha Kumara Kalutarage, a Lecturer at Robert Gordon University for the assistance he has provided during this research work. Finally, I want to thank Dr. Domhnall Carlin, a Research Fellow in CSIT, ECIT, Queen's University Belfast, for proofreading the paper.

REFERENCES

[1] J. K. Lee, S. Y. Moon, and J. H. Park, "Cloudrps: a cloud analysis based enhanced ransomware prevention system," The Journal of Supercomputing, vol. 73, no. 7, pp. 3065–3084, 2017.

[2] P. Mishra, E. S. Pilli, V. Varadharajan, and U. Tupakula, "Intrusion detection techniques in cloud environment: A survey," Journal of Network and Computer Applications, vol. 77, pp. 18–47, 2017.

[3] M. P. Sharma, M. S. Zawar, and S. B. Patil, "Ransomware analysis: Internet of things (iot) security issues, challenges and open problems inthe context of worldwide scenario of security of systems and malware attacks," Int. J. Innov. Res. n Sci. Eng, vol. 2, no. 3, pp. 177–184, 2016.

[4] G. O'Gorman and G. McDonald, Ransomware: A growing menace. Symantec Corporation, 2012.

[5] D. O'Brien, "Internet Security Threat Report Ransomware 2017," Tech. Rep., 2017.

[6] Barkely, "Ransomware Protection A Healthcare IT Handbook," Barkly, Tech. Rep., 2016. [Online]. Available: https://www.barkly.com/

[7] Sonicwall, "2017 Annual Threat Report," p. 23, 2017. [Online]. Available: https://bluekarmasecurity.net/wp-content/uploads/2017/06/SonicWall-2017-Annual-Threat-Report.pdf

[8] K. lab, "KSN Report: Ransomware and malicious cryptominers 2016-2018," Tech. Rep., 2018.

[9] Symantec, "Ransomware and businesses," p. 30, 2016. [Online]. Available: https://www.symantec.com/content/dam/symantec/docs/security-center/white-papers/ransomware-and-businesses-16-en.pdf

[10] R. Richardson and M. North, "Ransomware: Evolution, mitigation and prevention," International Management Review, vol. 13, no. 1, pp. 10–21, 2017.

[11] A. Liska and T. Gallo, Ransomware: Defending against digital extortion. " O'Reilly Media, Inc.", 2017.

[12] M. M. Ahmadian, H. R. Shahriari, and S. M. Ghaffarian, "Connection-monitor & connection-breaker: A novel approach for prevention and detection of high survivable ransomwares," in Information Security and Cryptology (ISCISC), 2015 12th International Iranian Society of Cryptology Conference on. IEEE, 2015, pp. 79–84.

[13] K. Cabaj and W. Mazurczyk, "Using software-defined networking for ransomware mitigation: the case of cryptowall," IEEE Network, vol. 30, no. 6, pp. 14–20, 2016.

[14] BLEEPINGCOMPUTER, "Locky Ransomware Information, Help Guide, and FAQ," 2016. [Online]. Available: https://www.bleepingcomputer.com/virus-removal/locky-ransomware-information-help

[15] K. Cabaj, P. Gawkowski, K. Grochowski, A. Nowikowski, and P. Żórawski, "The impact of malware evolution on the analysis methods and infrastructure," in Computer Science and Information Systems (FedCSIS), 2017 Federated Conference on. IEEE, 2017, pp. 549–553.

[16] K. Cabaj, P. Gawkowski, K. Grochowski, and D. Osojca, "Network activity analysis of cryptowall ransomware," Przeglad Elektrotechniczny, vol. 91, no. 11, pp. 201–204, 2015.

[17] A. Tseng, Y. Chen, Y. Kao, and T. Lin, "Deep learning for ransomware detection," IEICE Tech. Rep., vol. 116, no. 282, pp. 87–92, 2016.

[18] J. Jones and N. Shashidhar, "Ransomware analysis and defense wannacry and the win32 environment," INTERNATIONAL JOURNAL OF INFORMATION SECURITY SCIENCE, vol. 6, no. 4, pp. 57–69, 2017.

[19] K. Cabaj, M. Gregorczyk, and W. Mazurczyk, "Software-defined networking-based crypto ransomware detection using http traffic characteristics," Computers & Electrical Engineering, vol. 66, pp. 353–368, 2018.

[20] G. Cusack, O. Michel, and E. Keller, "Machine learning-based detection of ransomware using sdn," in Proceedings of the 2018 ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization. ACM, 2018, pp. 1–6.

[21] B. A. S. Al-rimy, M. A. Maarof, and S. Z. M. Shaid, "Ransomware threat success factors, taxonomy, and countermeasures: A survey and research directions," Computers & Security, 2018.

[22] A. Zimba, L. Simukonda, and M. Chishimba, "Demystifying ransomware attacks: Reverse engineering and dynamic malware analysis of wannacry for network and information security," Zambia ICT Journal, vol. 1, no. 1, pp. 35–40, 2017.

[23] D. Sgandurra, L. Muñoz-González, R. Mohsen, and E. C. Lupu, "Automated dynamic analysis of ransomware: Benefits, limitations and use for detection," arXiv preprint arXiv:1609.03020, 2016.

[24] A. Kharraz, S. Arshad, C. Mulliner, W. K. Robertson, and E. Kirda, "Unveil: A large-scale, automated approach to detecting ransomware." in USENIX Security Symposium, 2016, pp. 757–772.

[25] O. Hachinyan, "Detection of malicious software on based on multiple equations of api-calls sequences," in Young Researchers in Electrical and Electronic Engineering (EIConRus), 2017 IEEE Conference of Russian. IEEE, 2017, pp. 415–418.

[26] H. Daku, P. Zavarsky, and Y. Malik, "Behavioral-based classification and identification of ransomware variants using machine learning," in 2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE). IEEE, 2018, pp. 1560–1564.

[27] N. Moustafa and J. Slay, "Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set)," in Military Communications and Information Systems Conference (MilCIS), 2015. IEEE, 2015, pp. 1–6.

[28] Canadian Institute for Cybersecurity, "Datasets," 2017. [Online]. Available: https://www.unb.ca/cic/datasets/index.html

[29] K. A. Gandhi et al., "Survey on ransomware: a new era of cyber attack," International Journal of Computer Applications, vol. 168, no. 3, 2017.

[30] S. Garcia, M. Grill, J. Stiborek, and A. Zunino, "An empirical comparison of botnet detection methods," computers & security, vol. 45, pp. 100–123, 2014.

[31] L. Vu Hong, "Dns traffic analysis for network-based malware detection," 2012.

[32] J. Gardiner, M. Cova, and S. Nagaraja, "Command & control: Understanding, denying and detecting-a review of malware c2 techniques, detection and defences," arXiv preprint arXiv:1408.1136, 2014.

[33] D. K. Vishwakarma, "Domain name generation algorithms," 2017.

[34] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal, Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann, 2016.

[35] S. Gollapudi, Practical machine learning. Packt Publishing Ltd, 2016.

[36] S. Shalev-Shwartz and S. Ben-David, Understanding machine learning: From theory to algorithms. Cambridge university press, 2014.

[37] D. Sarkar, R. Bali, and T. Sharma, "Practical machine learning with python," 2018.

[38] S. Kumar, A. Viinikainen, and T. Hamalainen, "Machine learning classification model for network based intrusion detection system," in Internet Technology and Secured Transactions (ICITST), 2016 11th International Conference for. IEEE, 2016, pp. 242–249.

AHMAD O. ALMASHHADANI : Received his M.Sc. degree in Control and Computer Engineering from the Department of Electrical Engineering, University of Baghdad, Iraq in 2011. He is currently pursuing his Ph.D. degree in Computer Engineering at the Center for Secure Information Technologies, Queen's University Belfast, UK. His research interests include Computer Networks, Network Security, and Malware Analysis.

MUSTAFA KAIIALI : Completed his B.E. degree in Computer Science at Aleppo University, Syria in 2003. Then he obtained his M.Tech and Ph.D. degrees from the Department of Computer and Information Sciences (DCIS), University of Hyderabad, India in 2008 and 2012, respectively. His areas of expertise are: Cloud Computing, Information Security, and Networking. He also passed the test of Cisco Certified Network Professional in Security in 2012. He has several publications in well-reputed journals and international conferences. Currently, he is with the Center for Secure Information Technologies (CSIT), ECIT, Queen's University Belfast (QUB), United Kingdom as a Research Fellow in Cloud Security.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2907485, IEEE Access

SAKIR SEZER : Received his Dipl. Ing. in Electrical and Electronic Engineering from RWTH Aachen University, Germany, and his PhD in 1999 from Queen's University Belfast, UK. He holds the Chair for Secure Communication Technologies at Queen's University Belfast and is the Director and Head of Network and Cyber Security Research at the Centre for Secure Information Technologies (CSIT) at Queen's University Belfast. Prof Sezer is a world renowned authority in high-performance network processing and Internet security technologies. His research is leading major advances in the field of high-performance IP packet, content and security processing and is currently commercialised by a number of Silicon Valley corporations. For his achievements Professor Sezer has been awarded a number of prestigious awards including the InvestNI Enterprise Fellowship, and Enterprise Ireland and Intertrade Ireland Innovation and Enterprise awards. Professor Sezer is also co-founder and CTO of Titan IC Systems and is a member of various research and executive committees.

PHILIP OKANE : Received the B.Eng. degree in electronic systems and the M.Sc. degree in informatics from the University of Ulster in 1994 and 2009, respectively, and the Ph.D. degree from Queen's University Belfast with a focus on the detection of obfuscated malware. His career started as an embedded Software Engineer in the Telecoms industry and later moved to application development for the Finance industry. He is currently a Lecturer with the Center for Secure Information Technologies, Queen's University Belfast. He has 20 years of software development in industry. His interests include the low-level analysis and detection of malware.