A Proactive Approach for Detecting Ransomware based on Hidden Markov Model (HMM)

Mohammed A. Saleh
Computer Science Department, College of Sciences and Arts in Ar Rass
Qassim University, Kingdom of Saudi Arabia (KSA)

Abstract

A ransomware is the most hazardous kind of computer malware that causes a huge devastation to the computer systems, so that detecting it is highly required at the moment. Truthfully, several prior researchers addressed Markov Model and its variants, like Hidden Markov Model, to detect a malware, but none of them addressed the detection of ransomware through Assembly language instructions. In this paper, a new proactive approach for detecting ransomware based on Hidden Markov Model (HMM) is proposed in order to detect and classify ransomware. In addition, new datasets that comprises of various benign and ransomware are generated and collected. The proposed approach utilized Hidden Markov Model (HMM) for analyzing the generated and collected datasets from benign and ransomware samples, and it detected and classified all samples correctly with 73% accurate testing samples emissions sequence.

1. Introduction

In recent times, ransomware attacks form the extremely active and widely spread waves of malware attacks [1-2]. Ransomware is a kind of malware, which is maliciously attacks the victim and encrypts his digital objects, like files and folders, and enforces and extorts the victim to pay a ransom, such as a money or bitcoins, in order to decrypt his digital objects, and therefore; the victim gets his files and folders back [3-5]. Although a ransomware is one type of the malware, it differs than other types of malware in many features, such as the huge amount of digital objects, files and folders, processing for encrypting them and wiping the original copies completely, and eventually, it orders a ransom to the victim. Indeed, a ransomware constitutes an extreme havoc to computers systems, hence; it should be detected and classified preciously in order to avoid its negative effects [6-7]. In reality, a ransomware infects victims through a bug, like zero-days vulnerabilities, or social engineering tactics [8-10].

In this paper, a new proactive approach for detecting ransomware based on Hidden Markov Model (HMM) is proposed to tackle ransomware in order to detect and classify them properly [11]. The proposed approach is a proactive, since it is crucial to classify the sample before the execution under a computer system. In case the approach detects and classifies it as a ransomware, it blocks it immediacy, or otherwise allows it to execute [12]. Besides that, the approach utilizes reinforcement of machine learning, namely Hidden Markov Model (HMM), to train and test the generated and collected datasets [13-15].

This research paper is organized as follows: section 1 presents introduction. Sections 2 discusses the literature review, which involves definitions and theory of Hidden Markov Model (HMM), and previous related works. Section 3 demonstrates generating and collecting training and testing datasets. Section 4 explains a proactive approach for detecting ransomware based on Hidden Markov Model (HMM). Section 5 presents the empirical results and discusses their analyses accordingly. Finally, section 6 presents the conclusion and future work.

2. Literature review

2.1. Hidden Markov Model (HMM)

A Markov Model is a stochastic process that describes the probabilities of sequences of random variables and states [11][14]. It makes very strong assumption that when predicting the future, it only depends on the present, and the past does not matter at all, as labelled in the next equation 1.

$$P(S_{ik} \mid S_{i1}, S_{i2}, \dots, S_{ik-1}) = P(S_{ik} \mid S_{ik-1}) \quad (1)$$

where {S1, S2, …, SN} are a set of states. In many cases, the variables and states (classes) that we are interested in are not observable directly, which are hidden, and therefore; Hidden Markov Model is used, since it used to describe the both: observable and hidden variables and states (classes). According to [16-17] the Hidden Markov models is characterized by three essential problems:

- Problem 1 (Evaluation/Likelihood): aims to determine the evaluation (likelihood) based on the following equations: equation 2, equation 3, and equation 4.

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i) \quad (2)$$

$$\alpha_t(j) = \sum_{i=1}^{N} \alpha_{t-1}(i)\, a_{ij} b_j(o_t) \quad (3)$$

$$\alpha_1(j) = \pi_j b_j(o_1) \quad (4)$$

- Problem 2 (Decoding): seeks to discover the best hidden state sequence, and it uses the succeeding equations: equation 5, equation 6, and equation 7.

$$\text{Best Score } P * = \max_{i=1} N\, v_T(i) \quad (5)$$

$$v_t(j) = \max_{i=1} N\, v_{t-1}(i) a_{ij} b_j(o_t) \quad (6)$$

$$v_1(j) = \pi_j b_j(o_1) \quad (7)$$

- Problem 3 (Learning): trains HMM parameters by calculating the following equations: equation 8, equation 9, and equation 10.

$$P(O|\lambda) = \sum_{i=1}^{N} \pi_j\ b_j(o_1)\, \beta_1(j) \quad (8)$$

$$\beta_t(j) = \sum_{i=1}^{N} a_{ij} b_j(o_{t+1})\, \beta_{t+1}(j) \quad (9)$$

$$\beta_T(i) = 1 \quad (10)$$

where: λ: HMM Model, O: Observation Sequence, πi: Initial Probability Distribution, αt(j): Forward Path Probability, αt-1(j): Previous Forward Path Probability, aij: Transition Probability Matrix, bj(ot): Observation Likelihoods, vt-1: Previous Viterbi Path Probability, βT(j): Backward Probability.

2.2. Previous related works

A research done by [13] proposed a classifier based on Hidden Markov Model (HMM) in order to identify the family that a virus belongs to. The research covered Viruses, but not ransomware. Another research conducted by [18] examined Hidden Markov Model (HMM) for four different compilers, hand-written assembly code, three virus construction kits, and a metamorphic virus to state similarities and dissimilarities in the hidden states of the HMM. As well, it developed the dueling HMM Strategy for the creation of improved virus detection tools based on HMMs. The paper covered three virus construction kits and a metamorphic virus, but did cover ransomware. A different research accomplished by [19] applied Hidden Markov Model (HMM) to differentiate between a cyber-security attack and no attack. It breaks the data into three clusters using Fuzzy K mean (FKM), after it that labels a small data manually (Analyst Intuition), and lastly it uses HMM state-based approach. The study tackled merely network attacks. A research prepared by [12] created a HMM-based models for each malware family based on its sequence of system calls. The research treated almost all types of malware, except ransomwares. Another relevant research [20] compared API call sequences and opcode sequences using the HMM learning model in order to detect malware. The paper did not study any ransomware sample. A related research work [21] proposed a novel malware classification scheme that is based on Hidden Markov Models (HMMs) and discriminative classifiers. The proposed scheme takes the sequences of system calls that are generated by malware during execution as observation sequences to train the HMMs. The scheme was not examined and tested towards any ransomware sample. Two similar researches [14][22] introduced a new approach based on machine learning methods with n-gram model for detection malwares. It used Markov blanket method as feature selection technique, reduced size of features. The introduced new approach was not examined and tested against any ransomware sample. A research established by [23] presented a classification technique based on Hidden Markov Model (HMM) in order to classify computer viruses. It tested the presented technique towards used various virus construction tools, and none of them generate ransomware.

3. Generating and collecting training and testing datasets

In this paper, datasets are generated and collected from a combination of benign and ransomware samples; 23 benign samples and 22 ransomware samples. The research collected 23 benign samples randomly from a fresh 32-bit and 64-bits Windows operating systems. In the meanwhile, it collected 22 ransomware samples from online malware repositories like [24] and [25]. Subsequently, the whole samples, which involves benign and ransomware samples, are investigated and analyzed in terms of Assembly language instructions in order to explore the sharable Assembly language instructions among the entire samples. The most common sharable Assembly language instructions among the complete samples as discovered by the research are 59 instructions, as follows: mov lea xchg lods pop push call ret leave hlt int add sub div mul inc dec or and xor shr shl test cmp jo jno js jns je jz jne jnz jb jnae jc jnb jae jnc jbe jna ja jnbe jl jnge jge jnl jle jng jg jnle jp jpe jnp jpo jcxz jecxz jmp loop nop.

Afterward, these discovered 59 Assembly language instructions are grouped and clustered according to the type of instruction operation, which includes 5 different operation types, namely Data Processing Instructions (Opcodes), Process Instructions (Opcodes), Arithmetic Instructions (Opcodes), Logic Instructions (Opcodes), and Control Flow Instructions (Opcodes). The 59 Assembly language instructions are grouped and

clustered based on its equivalent operation type as the following: 6 Data Processing Instructions (Opcodes) involve mov lea xchg lods pop push, 5 Process Instructions (Opcodes) include call ret leave hlt int, 5 Arithmetic Instructions (Opcodes) contain add sub div mul inc dec, 6 Logic Instructions (Opcodes) encompass or and xor shr shl test, and 36 Control Flow Instructions (Opcodes) comprise cmp jo jno js jns je jz jne jnz jb jnae jc jnb jae jnc jbe jna ja jnbe jl jnge jge jnl jle jng jg jnle jp jpe jnp jpo jcxz jecxz jmp loop nop.

The benign and ransomware samples, executable Windows software, are converted to Assembly language instructions (Opcodes), and then occurrences of the Opcodes are counted according to the operation types that are listed above. Table 1 and Table 2, as shown in see Appendix 1 and Appendix 2, present the overall generated and collected datasets from benign and ransomware samples, respectively.

4. A proactive approach for detecting ransomware based on Hidden Markov Model (HMM)

Practically, the proposed proactive approach trains Hidden Markov Model (HMM) throughout Problem 3, Problem 1, and Problem 2 of the HMM model using the benign and ransomware training datasets, which are presented in Table 3 and Table 4, in order to identify benign profile and ransomware profile. The benefit of these profiles is that they will be used as benchmarks for classifying testing dataset later on. The overall datasets, which are shown in Table 1 and Table 2, are fragmented randomly into training dataset and testing dataset based on 80% for training dataset and 20% for testing dataset. The following Table 3 displays the training dataset from benign samples, while Table 4 presents the training dataset from ransomware samples. Table 5 shows the testing dataset from benign and ransomware samples. The training and testing processes of Hidden Markov Model (HMM) are demonstrated in Figure 1 and Figure 2, correspondingly.

Table 3. Training dataset from benign samples

No	Benign Software/Program	Data Processing Opcodes	Process Opcodes	Arithmetic Opcodes	Logic Opcodes	Control Flow Opcodes	Total
1	ComputerDefaults.exe	2217	906	3894	784	1801	9602
2	DisplaySwitch.exe	27806	9648	59453	8947	24779	130633
3	Magnify.exe	59064	15748	42508	22518	31588	171426
4	Narrator.exe	72352	4144	202347	47781	69287	395911
5	calc.exe	16631	3391	7405	5010	4079	36516
6	clipbrd.exe	11993	3670	8097	4608	6315	34683
7	cmd.exe	20227	4498	13926	6514	17038	62203
8	dvdplay.exe	428	311	1028	291	506	2564
9	freecell.exe	5632	1195	6428	2645	2750	18650
10	klist.exe	3034	1148	2264	1150	2059	9655
11	label.exe	1250	468	1050	435	806	4009
12	mstsc.exe	79785	29979	85167	29352	46704	270987
13	notepad.exe	15627	3503	10828	6927	11635	48520
14	ntprint.exe	4208	1060	3692	2001	3030	13991
15	osk.exe	41958	17077	77010	12778	24615	173438
16	syskey.exe	2028	1021	2316	1101	1480	7946
17	taskmgr.exe	15092	4456	14411	5737	7730	47426
18	winhlp32.exe	517	281	911	353	463	2525
19	write.exe	357	264	821	315	555	2312
Total		380206	102768	543556	159247	257220	1442997
Ratio		0.2635	0.0712	0.3767	0.1104	0.1783	1.0000

Table 4. Training dataset from ransomware samples

No	Ransomware	Data Processing Opcodes	Process Opcodes	Arithmetic Opcodes	Logic Opcodes	Control Flow Opcodes	Total
1	cerber.exe	27087	5423	24903	16281	155477	229171
2	cryptowall.exe	21993	1505	50955	13102	16373	103928
3	locky.exe	15425	5158	16462	10435	12260	59740
4	mamba.exe	233132	53856	232065	154467	133771	807291
5	matsnu.exe	9755	2653	13174	4838	5824	36244
6	petrwrap 1.exe	42976	8020	21662	19668	22007	114333
7	petrwrap 2.exe	60185	8278	103806	34585	41059	247913
8	petrwrap.exe	42976	8020	21662	19668	22007	114333
9	petya 1.exe	96530	23008	54206	35338	45482	254564
10	petya.exe	96530	23008	54206	35338	42333	251415
11	radaman_UPX.ViR.exe	0	0	0	1	0	1
12	satana.exe	8538	443	9593	3906	4516	26996
13	teslacrypt 1.exe	26309	5063	21494	10583	13831	77280
14	teslacrypt 2.exe	40067	9083	15219	14111	18316	96796
15	vipasana 1.exe	55799	10582	25264	26035	22068	139748
16	vipasana 3.exe	51241	9714	22437	22000	19819	125211
17	wannacry.exe	393909	68355	205262	179467	211577	1058570
Total		1516765	292154	1059744	734662	946823	4550148
Ratio		0.3333	0.0642	0.2329	0.1615	0.2081	1.0000

Table 5. Testing dataset from benign and ransomware samples

No	Benign Software/Program	Data Processing Opcodes	Process Opcodes	Arithmetic Opcodes	Logic Opcodes	Control Flow Opcodes	Total
1	Defrag.exe	16424	4116	10009	7056	9654	47259
	Ratio	0.3475	0.0871	0.2118	0.1493	0.2043	1.0000
2	petya 2.exe	32682	6592	17918	10897	13303	81392
	Ratio	0.4015	0.0810	0.2201	0.1339	0.1634	1.0000
3	colorcpl.exe	6861	1885	5526	3246	4324	21842
	Ratio	0.3141	0.0863	0.2530	0.1486	0.1980	1.0000
4	teslacrypt 3.exe	26848	5073	21489	10788	14089	78287
	Ratio	0.3429	0.0648	0.2745	0.1378	0.1800	1.0000
5	wannacry+.exe	294313	49985	167374	134839	160103	806614
	Ratio	0.3649	0.0620	0.2075	0.1672	0.1985	1.0000
6	resmon.exe	8543	2199	5670	4151	5261	25824
	Ratio	0.3308	0.0852	0.2196	0.1607	0.2037	1.0000
7	mblctr.exe	65404	19910	62871	26866	46543	221594
	Ratio	0.2952	0.0898	0.2837	0.1212	0.2100	1.0000
8	vipasana 2.exe	55622	10551	25397	26058	22199	139827
	Ratio	0.3978	0.0755	0.1816	0.1864	0.1588	1.0000

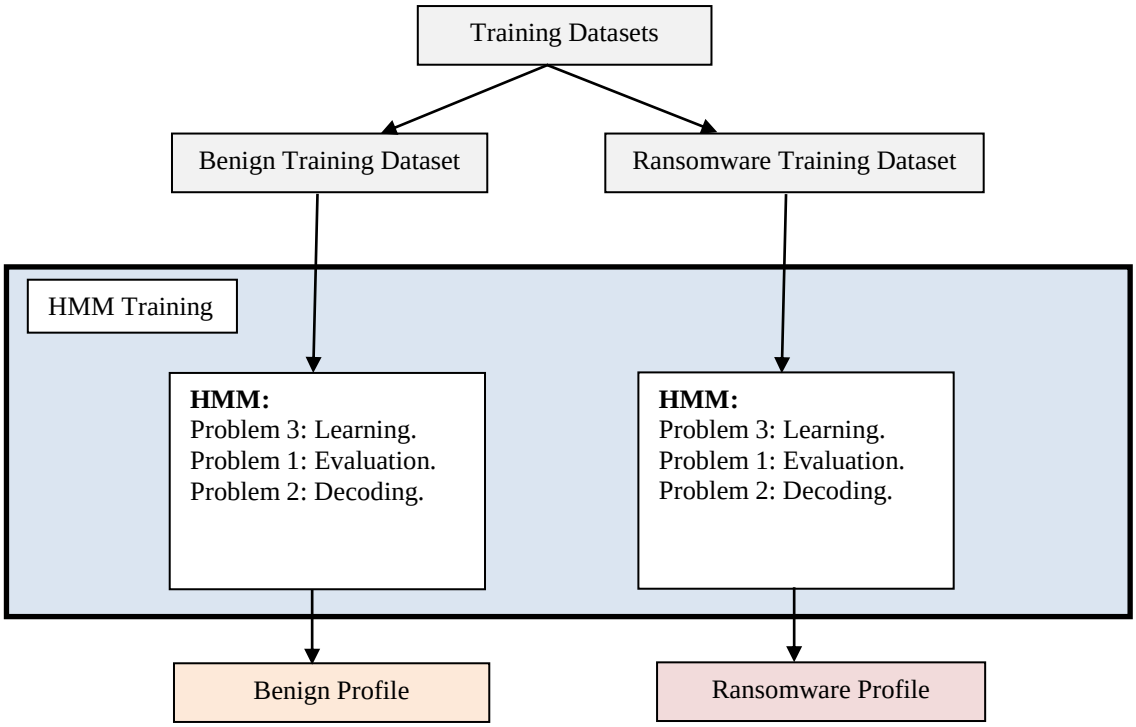

Figure 1. Training processes of Hidden Markov Model (HMM)

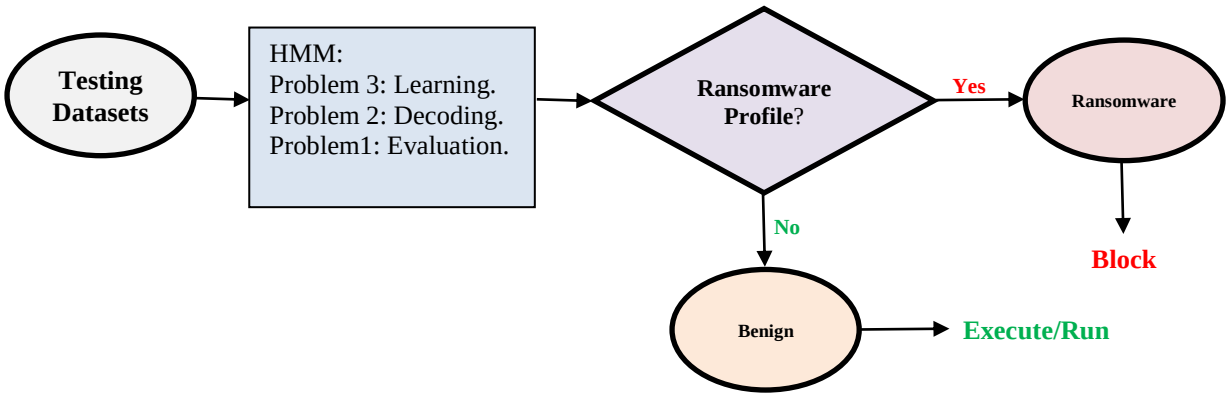

Figure 2. Testing processes of Hidden Markov Model (HMM)

5. Empirical results and discussions

The empirical for testing the proposed proactive approach for detecting ransomware based on Hidden Markov Model (HMM) are conducted on MATLAB software [26] for the sake of analysing training and testing datasets, which are generated and collected from benign and ransomware samples. First, the empirical computed Problem 3, which is training HMM model, twice for both training datasets of benign samples, in Table 3, and ransomware samples, in Table 4, as depicted in Figure 1. The results are shown as the following:

$$Ransomware\ Transition\ Matrix = \begin{bmatrix} 0.9000 & 0.1000 \\ 0.0500 & 0.9500 \end{bmatrix}$$

$$Ransomware\ Emissions\ Matrix = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0.3333 & 0.0642 & 0.2329 & 0.1615 & 0.2081 \end{bmatrix}$$

$$Estimated\ (Trained)\ Ransomware\ Transition\ Matrix = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}$$

$$Estimated\ (Trained)\ Ransomware\ Emissions\ Matrix = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0.2000 & 0.2000 & 0.2000 & 0.2000 & 0.2000 \end{bmatrix}$$

$$Ransomware\ Emissions\ Sequence = [1 \quad 3 \quad 5 \quad 4 \quad 2]$$

$$Benign\ Transition\ Matrix = \begin{bmatrix} 0.9000 & 0.1000 \\ 0.0500 & 0.9500 \end{bmatrix}$$

$$Benign\ Emissions\ Matrix = \begin{bmatrix} 0.2635 & 0.0712 & 0.3767 & 0.1104 & 0.1783 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$Estimated\ (Trained)\ Benign\ Transition\ Matrix = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$Estimated\ (Trained)\ Benign\ Emissions\ Matrix = \begin{bmatrix} 0.2000 & 0.2000 & 0.2000 & 0.2000 & 0.2000 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$Benign\ Emissions\ Sequence = [3 \quad 1 \quad 5 \quad 4 \quad 2]$$

Second, the empirical calculated Problem 1, which is an evaluation and a likelihood, and Problem 2, which is a decoding, of the HMM model twice for both training datasets of benign samples, in Table 3, and ransomware samples, in Table 3, as illustrated in Figure 1. The aim of these two steps is to construct benign profile and ransomware profile, which they will be used as benchmarks in testing process later on. The results are revealed in the next:

$$Ransomawre\ Posterior\ State\ Probabilities = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

$$Ransomawre\ Best\ State\ Path\ (Viterbi) = [2 \quad 2 \quad 2 \quad 2 \quad 2]$$

$$Benign\ Posterior\ State\ Probabilities = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$Benign\ Best\ State\ Path\ (Viterbi) = [1 \quad 1 \quad 1 \quad 1 \quad 1]$$

Finally, the empirical computed Problem 3, Problem 2, and Problem 1 of the HMM model for testing dataset of benign and ransomware samples in Table 5 randomly, as explained in Figure 2. Then, the gained result is compared to the ransomware profile, and in case it matches the profile, the sample is classified as a ransomware, or otherwise is classified as a benign software. The proposed proactive approach for detecting ransomware based on Hidden Markov Model (HMM) detected and classified all ransomware samples in the testing dataset precisely with 85% accurate ransomware testing samples emissions sequence. As well, it detected and classified all benign samples in the testing dataset exactly with 60% accurate benign testing samples emissions sequence, with overall 73% accurate testing samples emissions sequence, as shown in Table 6 and Table 7.

Table 6. The results of Hidden Markov Model (HMM) on testing dataset samples

No	Program	Posterior State Probabilities	Emissions Sequence	Best State Path (Viterbi)	Classification
1	Defrag.exe	[1 1 1 1 1]	[1 3 5 4 2]	[1 1 1 1 1]	Benign
2	petya 2.exe	[2 2 2 2 2]	[1 3 5 4 2]	[2 2 2 2 2]	Ransomware
3	colorcpl.exe	[1 1 1 1 1]	[1 3 5 4 2]	[1 1 1 1 1]	Benign
4	teslacrypt 3.exe	[2 2 2 2 2]	[1 3 5 4 2]	[2 2 2 2 2]	Ransomware
5	wannacry+.exe	[2 2 2 2 2]	[1 3 5 4 2]	[2 2 2 2 2]	Ransomware
6	resmon.exe	[1 1 1 1 1]	[1 3 5 4 2]	[1 1 1 1 1]	Benign
7	mblctr.exe	[1 1 1 1 1]	[1 3 5 4 2]	[1 1 1 1 1]	Benign
8	vipasana 2.exe	[2 2 2 2 2]	[1 4 3 5 2]	[2 2 2 2 2]	Ransomware

Table 7. The percentage of detection and classification for testing dataset samples

No	Classification	Detection and Classification Percentage	Emissions Sequence Percentage	
1	Benign	100%	60%	73%
2	Ransomware	100%	85%	

6. Conclusion

This research paper proposed a new proactive approach for detecting ransomware based on Hidden Markov Model (HMM). The approach is a proactive, since it firstly analyzes the samples, and in case it classifies it as a ransomware, it will detect and eliminate it before the execution. It generated and collected training and testing datasets from various benign and ransomware samples. In the meantime, it investigated and analyzed the samples in terms of Assembly language instructions, or Opcodes, in order to explore the sharable Assembly language instructions among the entire samples. The most common sharable Assembly language instructions among the complete samples as discovered by the research were 59 instructions, which were grouped and clustered according to the type of instruction operation, which includes 5 different operation types, namely Data Processing Instructions (Opcodes), Process Instructions (Opcodes), Arithmetic Instructions (Opcodes), Logic Instructions (Opcodes), and Control Flow Instructions (Opcodes). The approach identified and constructed benign profile and ransomware profile, which will be used as benchmarks for classifying testing dataset later on. Empirically, the proposed approach detected and classified all ransomware samples in the testing dataset precisely with 85% accurate ransomware testing samples emissions sequence. As well, it detected and classified all benign samples in the testing dataset exactly with 60% accurate benign testing samples emissions sequence, with overall 73% accurate testing samples emissions sequence. Hopefully the future works will expand this research to involve more ransomware samples.

7. References

[1] P. Paganini, "Thousands of servers infected with the Lilocked Ransomware," *securityaffairs.co*, 07-Sep-2019.

[2] C. Cimpanu, "Ransomware gang wanted $5.3 million from US city, but they only offered $400,000," *www.zdnet.com*, 2019. [Online]. Available: https://www.zdnet.com/article/ransomware-gang-wanted-5-3-million-from-us-city-but-they-only-offered-400000/. [Accessed: 13-Sep-2019].

[3] C. Cimpanu, "Thousands of servers infected with new Lilocked (Lilu) ransomware," *www.zdnet.com*, 2019. [Online]. Available: https://www.zdnet.com/article/thousands-of-servers-infected-with-new-lilocked-lilu-ransomware/. [Accessed: 13-Sep-2019].

[4] I. Ilascu, "Fake PayPal Site Spreads Nemty Ransomware," *www.bleepingcomputer.com*, 2019. [Online]. Available: https://www.bleepingcomputer.com/news/security/fake-paypal-site-spreads-nemty-ransomware/. [Accessed: 13-Sep-2019].

[5] S. Morgan, "Global Ransomware Damage Costs Predicted To Exceed $5 Billion In 2017," May-2017.

[6] K. P. Subedi, D. R. Budhathoki, and D. Dasgupta, "Forensic Analysis of Ransomware Families Using Static and Dynamic Analysis," *2018 IEEE Secur. Priv. Work.*, pp. 180–185, 2018.

[7] E. Kirda, "UNVEIL: A large-scale, automated approach to detecting ransomware (keynote)," *2017 IEEE 24th Int. Conf. Softw. Anal. Evol. Reengineering*, pp. 1–1, 2017.

[8] S. Poudyal, K. P. Subedi, and D. Dasgupta, "A Framework for Analyzing Ransomware using Machine Learning," in *2018 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2018, pp. 1692–1699.

[9] D. Y. Kim, G. Y. Choi, and J. H. Lee, "White list-based ransomware real-time detection and prevention for user device protection," *2018 IEEE Int. Conf. Consum. Electron. ICCE 2018*, vol. 2018-Janua, pp. 1–5, 2018.

[10] M. M. Ahmadian and H. R. Shahriari, "2entFOX: A framework for high survivable ransomwares detection," *13th Int. ISC Conf. Inf. Secur. Cryptology, Isc. 2016*, pp. 79–84, 2016.

[11] S. K. Sasidharan and C. Thomas, "A Survey on Metamorphic Malware Detection based on Hidden Markov Model," *2018 Int. Conf. Adv. Comput. Commun. Informatics, ICACCI 2018*, pp. 357–362, 2018.

[12] R. Pranamulia, Y. Asnar, and R. S. Perdana, "Profile Hidden Markov Model for Malware Classification - Usage of System call Sequence for Malware Classification," 2017.

[13] S. P. Thunga and R. K. Neelisetti, "Identifying metamorphic virus using n-grams and Hidden Markov Model," *2015 Int. Conf. Adv. Comput. Commun. Informatics, ICACCI 2015*, pp. 2016–2022, 2015.

[14] B. Pechaz, M. V. Jahan, and M. Jalali, "Malware detection using hidden markov model based on markov blanket feature selection method," *2nd Int. Congr. Technol. Commun. Knowledge, ICTCK 2015*, no. Ictck, pp. 558–563, 2016.

[15] K. Xin, G. Li, Z. Qin, and Q. Zhang, "Malware detection in smartphone using Hidden Markov Model," *Proc. - 2012 4th Int. Conf. Multimed. Secur. MINES 2012*, pp. 857–860, 2012.

[16] D. Jurafsky and J. H. Martin, *An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, Third Edit. 2018.

[17] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition.," in *Proceedings of the IEEE, 77(2)*, 1989, pp. 257–286.

[18] T. H. Austin, E. Filiol, S. Josse, and M. Stamp, "Exploring hidden Markov models for virus analysis: A semantic approach," *Proc. Annu. Hawaii Int. Conf. Syst. Sci.*, pp. 5039–5048, 2013.

[19] T. T. Teoh, Y. Y. Nguwi, Y. Elovici, N. M. Cheung, and W. L. Ng, "Analyst intuition based Hidden Markov Model on high speed, temporal cyber security big data," *ICNC-FSKD 2017 - 13th Int. Conf. Nat. Comput. Fuzzy Syst. Knowl. Discov.*, pp. 2080–2083, 2018.

[20] S. Alqurashi and O. Batarfi, "A comparison between API call sequences and opcode sequences as reflectors of malware behavior," *2017 12th Int. Conf. Internet Technol. Secur. Trans. ICITST 2017*, pp. 105–110, 2018.

[21] M. Imran, M. T. Afzal, and M. A. Qadir, "Similarity-Based Malware Classification Using Hidden Markov Model," *Proc. - 4th Int. Conf. Cyber Secur. Cyber Warf. Digit. Forensics, CyberSec 2015*, pp. 129–134, 2016.

[22] H. Divandari, B. Pechaz, and M. V. Jahan, "Malware detection using Markov Blanket based on opcode sequences," *2nd Int. Congr. Technol. Commun. Knowledge, ICTCK 2015*, no. Ictck, pp. 564–569, 2016.

[23] F. Rezaei, M. Hamedi-Hamzehkolaie, S. Rezaei, and A. Payandeh, "Metamorphic viruses detection by hidden Markov models," *7'th Int. Symp. Telecommun.*, pp. 821–826, 2015.

[24] Y. Nativ, "theZoo project." [Online]. Available: https://github.com/ytisf/theZoo. [Accessed: 15-Aug-2019].

[25] CrowdStrike, "reverse." [Online]. Available: https://www.reverse.it/. [Accessed: 15-Aug-2019].

[26] MathWorks, "MATLAB." MathWorks, 2019.

Appendix 1

Table 1. The overall generated and collected dataset from benign samples

No	Benign Software/Program	Data Processing Opcodes	Process Opcodes	Arithmetic Opcodes	Logic Opcodes	Control Flow Opcodes	Total
1	ComputerDefaults.exe	2217	906	3894	784	1801	9602
2	Defrag.exe	16424	4116	10009	7056	9654	47259
3	DisplaySwitch.exe	27806	9648	59453	8947	24779	130633
4	Magnify.exe	59064	15748	42508	22518	31588	171426
5	Narrator.exe	72352	4144	202347	47781	69287	395911
6	calc.exe	16631	3391	7405	5010	4079	36516
7	clipbrd.exe	11993	3670	8097	4608	6315	34683
8	cmd.exe	20227	4498	13926	6514	17038	62203
9	colorcpl.exe	6861	1885	5526	3246	4324	21842
10	dvdplay.exe	428	311	1028	291	506	2564
11	freecell.exe	5632	1195	6428	2645	2750	18650
12	klist.exe	3034	1148	2264	1150	2059	9655
13	label.exe	1250	468	1050	435	806	4009
14	mblctr.exe	65404	19910	62871	26866	46543	221594
15	mstsc.exe	79785	29979	85167	29352	46704	270987
16	notepad.exe	15627	3503	10828	6927	11635	48520
17	ntprint.exe	4208	1060	3692	2001	3030	13991
18	osk.exe	41958	17077	77010	12778	24615	173438
19	resmon.exe	8543	2199	5670	4151	5261	25824
20	syskey.exe	2028	1021	2316	1101	1480	7946
21	taskmgr.exe	15092	4456	14411	5737	7730	47426
22	winhlp32.exe	517	281	911	353	463	2525
23	write.exe	357	264	821	315	555	2312
Total		477438	130878	627632	200566	323002	1759516

Appendix 2

Table 2. The overall generated and collected dataset from ransomware samples

No	Ransomware	Data Processing Opcodes	Process Opcodes	Arithmetic Opcodes	Logic Opcodes	Control Flow Opcodes	Total
1	cerber.exe	27087	5423	24903	16281	155477	229171
2	cryptowall.exe	21993	1505	50955	13102	16373	103928
3	locky.exe	15425	5158	16462	10435	12260	59740
4	mamba.exe	233132	53856	232065	154467	133771	807291
5	matsnu.exe	9755	2653	13174	4838	5824	36244
6	petrwrap 1.exe	42976	8020	21662	19668	22007	114333
7	petrwrap 2.exe	60185	8278	103806	34585	41059	247913
8	petrwrap.exe	42976	8020	21662	19668	22007	114333
9	petya 1.exe	96530	23008	54206	35338	45482	254564
10	petya 2.exe	32682	6592	17918	10897	13303	81392
11	petya.exe	96530	23008	54206	35338	42333	251415
12	radaman_UPX.ViR.exe	0	0	0	1	0	1
13	satana.exe	8538	443	9593	3906	4516	26996
14	teslacrypt 1.exe	26309	5063	21494	10583	13831	77280
15	teslacrypt 2.exe	40067	9083	15219	14111	18316	96796
16	teslacrypt 3.exe	26848	5073	21489	10788	14089	78287
17	vipasana 1.exe	55799	10582	25264	26035	22068	139748
18	vipasana 2.exe	55622	10551	25397	26058	22199	139827
19	vipasana 3.exe	51241	9714	22437	22000	19819	125211
20	wannacry+.exe	294313	49985	167374	134839	160103	806614
21	wannacry.exe	393909	68355	205262	179467	211577	1058570
22	wannacryPlus.exe	294313	49985	167374	134839	160103	806614
Total		1926230	364355	1291922	917244	1156517	5656268