applied sciences

MDPI

Article

A Multi-Tier Streaming Analytics Model of 0-Day Ransomware Detection Using Machine Learning

Hiba Zuhair [1], Ali Selamat [2,3,4,*] and Ondrej Krejcar [4]

[1] Department of Systems Engineering, College of Information Engineering, Al-Nahrain University, Baghdad 64074, Iraq; hiba.zuhir@coie-nahrain.edu.iq
[2] School of Computing, Faculty of Engineering, UTM & Media and Games Center of Excellence (MagicX), Universiti Teknologi Malaysia (UTM), Johor Baharu, Johor 81310, Malaysia
[3] Malaysia Japan International Institute of Technology (MJIIT), Universiti Teknologi Malaysia, Jalan Sultan Yahya Petra, Kuala Lumpur 54100, Malaysia
[4] Center for Basic and Applied Research, Faculty of Informatics and Management, University of Hradec Kralove, Rokitanskeho 62, 500 03 Hradec Kralove, Czech Republic; ondrej.krejcar@uhk.cz
* Correspondence: aselamat@utm.my

Received: 25 March 2020; Accepted: 27 April 2020; Published: 4 May 2020

Abstract: Desktop and portable platform-based information systems become the most tempting target of crypto and locker ransomware attacks during the last decades. Hence, researchers have developed anti-ransomware tools to assist the Windows platform at thwarting ransomware attacks, protecting the information, preserving the users' privacy, and securing the inter-related information systems through the Internet. Furthermore, they utilized machine learning to devote useful anti-ransomware tools that detect sophisticated versions. However, such anti-ransomware tools remain sub-optimal in efficacy, partial to analyzing ransomware traits, inactive to learn significant and imbalanced data streams, limited to attributing the versions' ancestor families, and indecisive about fusing the multi-descent versions. In this paper, we propose a hybrid machine learner model, which is a multi-tiered streaming analytics model that classifies various ransomware versions of 14 families by learning 24 static and dynamic traits. The proposed model classifies ransomware versions to their ancestor families numerally and fuses those of multi-descent families statistically. Thus, it classifies ransomware versions among 40K corpora of ransomware, malware, and good-ware versions through both semi-realistic and realistic environments. The supremacy of this ransomware streaming analytics model among competitive anti-ransomware technologies is proven experimentally and justified critically with the average of 97% classification accuracy, 2.4% mistake rate, and 0.34% miss rate under comparative and realistic test.

Keywords: crypto-ransomware; locker-ransomware; static analysis; dynamic analysis; machine learning

1. Introduction

Motivated by fame and illegal profit, cyber-criminals have threatened users' privacy and information systems by ransomware attacks [1]. Thus, different anti-ransomware tools and anti-malware software have been developed to detect ransomware attacks of various ransomware families on desktop and portable platforms [2,3]. Along with them; are the anti-ransomware tools assisted by machine learning that learn ransomware data with a set of static and/or dynamic traits to examine ransomware attacks at their runtime successfully [3,4]. Although they perform better than their competitors, machine learning-based anti-ransomware tools still suffer from late ransomware tackle, somewhat incorrect categorization of a ransomware family among other malware families, variable performance outcomes versus various ransomware families, complex computations, and longtime reaction with heavy use of CPU and memory [4,5]. Accordingly, they still provide a chance of evasion to the cyber-criminals who advance their exploitations to evolve 0-Day versions of new ransomware families every day [6]. Then, more users' data loss, systems' data leakage, and users' money loss would be produced along with other tragic concerns to cyber-security [4–6]. Since data protection, systems' defense, and cyber-space survival is the superior aim of researchers in cyber-security [7]. A more proficient scheme to detect ransomware attacks in their runtime is required to overcome the previous issues of the existing anti-ransomware tools. The required solution should be efficacious with less performance overhead, and adaptive to operate on desktop and portable platforms of servers, PCs, tablets, and smartphones. Furthermore, it should identify the ransomware version among the generic versions of malware and good-ware apps as well as categorize its corresponding family decisively. In addition, it should analyze both *crypto* and *locker* ransomware families automatically to extract their static and dynamic traits that discriminating against them from other malicious families and good-ware apps with light use of CPU and memory.

To this end, we propose a ransomware streaming analytics model by integrating a compact set of 24 static and dynamic traits, a hybrid machine learner, a numeral measurement for ransomware's ancestor family attribution, and a statistic formula for a multi-descent ransomware version via a multi-tiered architecture. The proposed machine learner trains a set of 24 rich traits to characterize 14 ransomware families in a semi-realistic environment. Correspondingly, it identifies the ancestor family as well as the multi-descent ransomware versions among more than 40K of various ransomware, malware, and good-ware versions decisively. Overall, this is done through multiple and synchronous tiers, including ransomware characterization and family attribution tiers, multi-descent ransomware fusion tiers, and then ransomware classification tier. To affirm the efficacy and the supremacy of our proposed approach, an extensive study, test, and benchmarking are conducted across other machine learners and anti-ransomware tools that have been recently adopted in the anti-ransomware domain. Also, a critical qualification is achieved to distinguish and justify their limitations in terms of the type of machine learning algorithms, the employed traits of ransomware, the family attribution, the robustness against big datasets, and the performance outcomes. Hence, the addressed limitations demonstrate what is overlooked, which our proposed model boosts for the best detection performance? Precisely, this paper makes a six-fold contribution as follows:

- Revisiting state-of-the-art anti-ransomware technologies, particularly those assisted by machine learners to highlight the issues that they have overlooked in ransomware detection.
- Deploying an informative compact set of static and dynamic traits to holistically characterize the ransomware versions of both crypto and locker families.
- Enhancing numeral and statistic metrics to attribute a ransomware version to its ancestor family, and to fuse ransomware versions of multi- descent families, respectively.
- Proposing a machine learner that unearths the 0-Day versions of ransomware into a generalized class model.
- Designing a multi-tier streaming analytics model to implement in a realistic environment and operate on manifold platforms by integrating the compact set of traits, the numeral and statistic metrics, and the proposed machine learner.
- Significance of our proposals is manifested through testing, evaluating, and benchmarking on a big dataset consisting of 35,000 ransomware versions of 14 families, 500 versions of 10 malware, and 500 good-ware apps aggregated at a different time from different data archives.

For the aforesaid contribution, the remaining of this paper is organized as follows: Section 2 describes the ransomware life cycle, types, activities, and families. Section 3 revisits the state-of-the-art anti-ransomware tools and appraises them critically. In Section 4, the materials and methodology are elaborated, including the proposed machine learner, numeral, and statistic formulas. Results of the test, evaluation, and benchmarking are presented in Section 5. Then, Section 6 discusses the observations and addresses the pivot issues of the proposed model and machine learner. Finally, Section 7 concludes the overall facets investigated in this paper, along with future outlook.

2. Ransomware Versions and Ransomware Families

Even though, cyberspace provides useful communication media to the users with many services of e-commerce and e-government [1,2], it puts them at the risk of ransomware attacks that cause significant damage to their interconnected information systems and then money loss [2–4]. The risk of ransomware often occurs when the cyber-criminals exploit e-commerce services and e-government applications to access the victims' information systems fully and to gain a ransom from those victims [8]. Furthermore, the advanced electronic payment methods and electronic currencies like bitcoin and de facto payments enable the cyber-criminals to deploy their social engineering technologies to deceive the victims [8,9]. Usually, a ransomware attack is a malicious variant aiming at either locking the victim's information system or encrypting that system and its user files for a ransom acquirement to revive the system and regain the files access [1,4,10]. As we illustrate in Figure 1, cyber-criminals pursue a "recipe-to-success" strategy to deceive the users, intruding their information system on different platforms like Windows and Androids [4,10]. Through the "recipe-to-success" strategy, the victim catches the bait via an email attachment or spoofed link that is under the cyber-criminal's control. Then, the ransomware is disseminated to the victim's information system by exploiting that system's vulnerabilities to infect that system and encrypt particular system files and/or lock particular system's locations. Accordingly, the victim's access to his/her system or files is blocked, and the ransomware attack acquires a ransom from the victim to either decrypt the files or unlock the system [4,10].

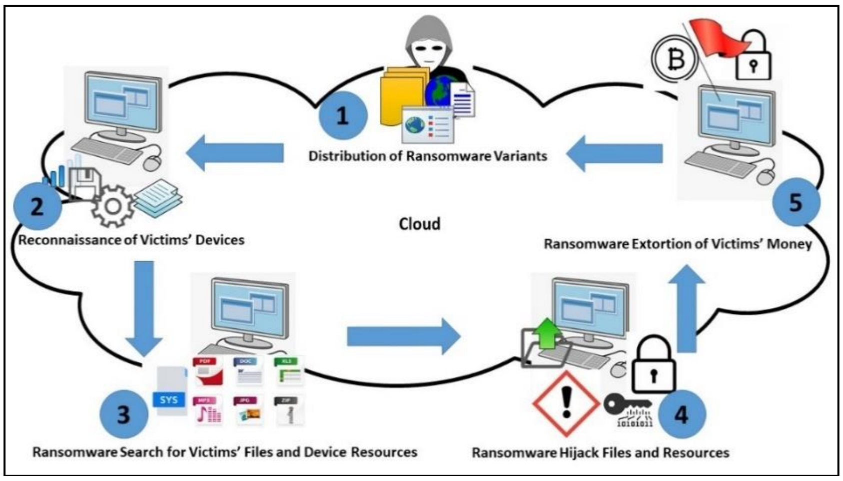

1
Distribution of Ransomware Variants
2
Reconnaissance of Victims' Devices
Cloud
5
Ransomware Extortion of Victims' Money
3
Ransomware Search for Victims' Files and Device Resources
4
Ransomware Hijack Files and Resources

Figure 1. Ransomware routine.

Continually, cyber-criminals evolve ransomware families of either *crypto* or *locker* type, as described in Table 1. *Crypto*-ransomware leverages symmetric and asymmetric ciphering on the user data and system files, whereas *locker*-ransomware hijacks the hosted system's resources and apps to disable the user's access to them [3,4]. As they intend to bypass the existing system defense' settings to cause more potential damage and gain more profit; cyber-criminals used to create many patterns (i.e., versions) belonging to every evolving family [2,3,7]. Generally, ransomware versions run sophisticated intrusion actions and employ advanced exploits that may infect different or similar platforms [8,9]. Additionally, they act similarly to those of other malicious threats and/or they maneuver to those of good-ware apps [8–10]; for example, the ransomware families that are described in Table 1. Hence, ransomware versions and families require different analytics mechanisms assisted by different compact sets of traits to recognize them, among other malware and good-ware apps accurately [4,10].

Table 1. Time-line summary of ransomware families.

Family	Year	Type	Exploits and Actions	Damage (s)
AiDS [10]	1989	Crypto	It was delivered to computer-based information systems via floppy disks	• Leakage of Root Directories • Loss of System Files
GpCode [10,11]	2005	Crypto	It was developed with an asymmetric encryption algorithm to encrypt users' data files	• Holding Up the Banking Information Systems • Loss of System and User Files
Archiveus [10,11]	2006	Crypto	It was applied with the RSA algorithm to encrypt system files	• Ruining the Original Version of Windows Platform • Data Loss • Money Loss
WinLock [10,11]	2010	Locker	It locked the computer system and demanded ransom via sending SMS to the victim's phone number	• Holding Up The Operating System • Loss of Backup Data • Loss of Money
Reveton [4,10,11]	2012	Locker	It impersonated the law enforcement agencies to deceive users with rumor claims	• Holding Up The Operating System • Abusing the Prepaid Electronic Payment Platforms • Loss of Backup Data
Crypto-Locker [4,10,11]	2013	Crypto	It encrypted the file's contents by RSA algorithm with private and public keys	• The Halt of Targeting System • Loss of Backup Data • Loss of Money
Crypto-Wall [4,10,11]	2014	Crypto	It encrypted the system files and injected malicious codes which freezes the system's firewalls	• Leakage of Original System Files • Loss of User Files • Loss of Money in Bitcoins
Ransom as Service (RaaS) [6,10]	2015	Locker	It used the social engineering techniques to impersonate a good-ware website as a malicious website in the dark web	• The Halt of Targeting Systems • Loss of Backup Data • Loss of Money in Bitcoins
Cerber [10,11]	2016	Crypto	It injected the malicious instructions to overwrite an encrypted content onto the original system	• Loss of Original System Files • Deactivation of the system registry
Crysis [10]	2016	Crypto	It encrypted the system files, rewrote their contents, and renamed them	• Leakage of System Files • Loss of User Data • Loss of Backup Data
Locky [4,10,11]	2016	Locker	It used the social engineering techniques to intrude the system through vulnerabilities of system settings, deactivated the registry actions, and removed the backup data	• The Halt of the Targeting System • Loss of Backup Data • Money Loss
WannaCry [4,10]	2017	Crypto	It encrypted the contents of the system files and removed the original system files	• Data Loss • Holding Up the Operating System • Money Loss in Bitcoins
Sopra [12,13]	2017	Crypto	It encrypted the contents of the system files and removed the original system files	• Data Loss • Holding Up the Operating System • Money loss in Bitcoins
Zeus [12,13]	2018	Crypto	It encrypted the contents of systems files and created new files with extensions belonging to different ransomware versions	• The Halt of the Targeting System of Industrial Organization • Backup Data Loss • Money Loss

3. The State-of-The-Art of Anti-Ransomware Technology

To thwart ransomware families and their corresponding versions, researchers have developed various anti-ransomware technologies during the last decades, and they categorized them into misuse-based technology, anomaly-based technology, and machine learning-based technology [6,14–16], as described briefly in Table 2. For a clear description, we illustrate the categories above, along with their advantages, are presented in Figure 2.

Table 2. Brief description of the state-of-the-art anti-ransomware technology.

Table	Description	Demerits	Examples
Misuse-Based Technology [6,14–16]	They analyze ransomware versions to extract cryptographic primitives, suspicious scripts, built-in functions, infected files' paths, and extensions. They achieved moderate accuracy in lightweight performance	-User's data can be lost -They can be obfuscated -They can be defeated against 0-Day ransomware versions -A lot of false alarms -The database must be updated frequently	Bitdefender Kaspersky McAfee Avast
Anomaly-Based Technology [17–23]	They trace ransomware runtime activities, computer processes, CU and memory footprints, server actions and control to detect ransomware versions effectively and efficiently	-The infirm vs. scalable network traffic -They can be defeated vs. 0-Day ransomware versions -Fragile analysis vs. big and imbalanced dataset	R-Locker RansomFlare Poster UNIVEIL Talos
Machine Learning-Based Technology [11,24–32]	They apply machine learning algorithms to classify training and testing sets of ransomware and good-ware instances with a hybrid set of static and dynamic traits. Then, generated class models characterize generic and/or unseen ransomware versions with high detection accuracy, low false alarms, and misclassifications	-They can be evaded by adversarial ransomware traits and newly emerged ransomware families -Simi-real testbed and real-mode conditions are needed -Obsolescent detection vs. various ransomware families	EldeRan ShieldFS 2entFOX GAN NetConverse RansomWall RANDS DRTHIS

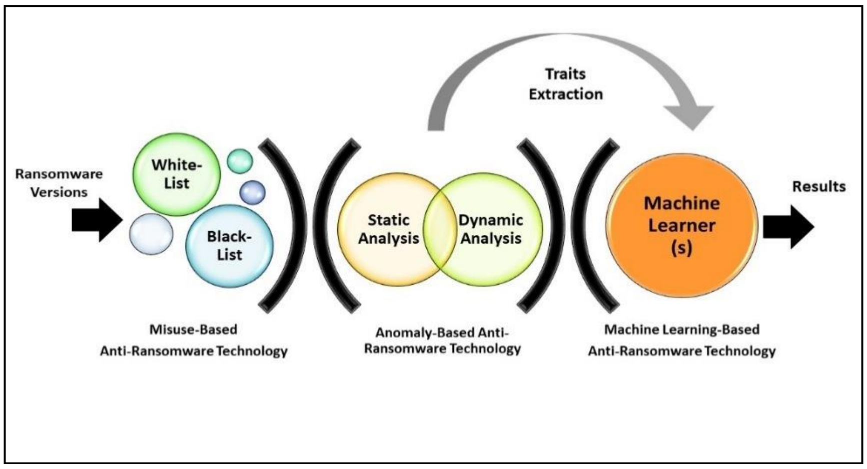

Traits Extraction
Ransomware Versions
White-List
Black-List
Static Analysis
Dynamic Analysis
Machine Learner (s)
Results
Misuse-Based Anti-Ransomware Technology
Anomaly-Based Anti-Ransomware Technology
Machine Learning-Based Anti-Ransomware Technology

Figure 2. Essentials and appendages of the state-of-the-art anti-ransomware technologies.

The misuse-based anti-ransomware technology relies on either archiving the exploits and actions of ransomware families in a black-list or archiving exploits of good-ware apps in a white-list [6,14–16]. However, it can detect the generic versions of the prevalent ransomware families exclusively as if the built-in archives are not frequently updated by the data of 0-Day versions [6,16]. Furthermore, it consumes a longer time and more computer resources as well as human labor to trace ransomware exploits and actions [6,10,14]. Unlikely, the anomaly-based anti-ransomware technology can analyze ransomware' normal behaviors and generic processes statically and/or dynamically to deviate 0-Day versions [4,6] (see Figure 2). Although anomaly-based anti-ransomware technology outperforms the misuse-based technology against 0-Day versions, it is still bypassed by versions of more advanced crypto coding families [6,10,14].

As described in Table 2, machine learning-based anti-ransomware technology applies various machine learners to cope with the problems of the former anti-ransomware technologies at discriminating 0-Day versions of different families expertly [6,24]. Furthermore, the constructed machine learners take the advantages of static and/or dynamic analysis of ransomware traits to extract the vectors of traits and learn them with a compact set of traits [25,26]. Precisely, they rely on different decisive functions, induction parameters, design, and class attributes to promote their discriminating power; for example, naïve bayes (NB), support vector machine (SVM), decision tree (DT), logic regression (LR), bayesian network (BN), neural networks (k-NN), and random forest (RF), etc. as well as hybrid machine learners like (HMLC) that hybridizes multiple and complementary machine learners as presented in Table 3. Therefore, machine learning-based anti-ransomware technologies outperform their competitors [2–4]. However, they still fall short at characterizing ransomware families holistically, attributing ransomware versions to their ancestor families, fusing ransomware versions of multi-descents, learning big data stream, and running on manifold platforms [24], as presented in Table 3. Beyond the above, the next sections will emulate the milestones of our proposed solutions, justify its sufficiency versus the lack of state-of-the-art anti-ransomware technology, qualify its performance, and enumerate its future outlook.

Table 3. Prominent anti-ransomware tools assisted by machine learners.

Existing Tools	Method	Machine Learning Algorithm	Lacks
EldeRan [11]	It detected the updates of API calls, registry key and file system operations via a sandbox versus locker-ransomware families	DT	• Exclusive for locker ransomware • Heavyweight in use • Time-sensitive • High detection faults • Unaware of new ransomware families • Unaware of family attribution • Unaware of multiple datasets • Unaware of manifold platforms
ShieldFS [30]	It detected some crypto-ransomware families that exploit generic ransomware traits like I/O and low-level file system infections, encrypting file contents, and overwriting the original contents by monitoring file system activity and then updating the threat profile overtime on a realistic environment.	DT	• Exclusive for crypto-ransomware • Heavyweight in use • Time-sensitive • High detection faults • Unaware of new ransomware families • Unaware of multiple datasets • Unaware of manifold platforms • Unaware of family attribution
Net-Converse [26]	It used dynamic traits and six machine learning algorithms to detect 0-Day versions of locker-ransomware versions	LR, DT, BN	• Exclusive for locker-ransomware • Performance overhead • Unaware of new ransomware families • Unaware of runtime condition • Unaware of static analysis • Unaware of family attribution • Unaware of multiple datasets • Unaware of manifold platforms
2entFOX [31]	It detected static and dynamic traits of crypto-ransomware by using graph traversal network	BN	• Exclusive for crypto-ransomware • Performance overhead • Heavyweight in use • Unaware of new ransomware families • Unaware of family attribution • Unaware of manifold platforms
GAN [32]	It develops a generative adversarial network for detecting versions of locker-ransomware families with a generic set of dynamic traits	NN, NB, RF, SVM	• Exclusive for locker-ransomware • Performance overhead • Heavyweight in use • Unaware of multi-class decision • Unaware of new ransomware families • Unaware of runtime condition • Unaware of multiple datasets • Unaware of static analysis • Unaware of manifold platforms
Ransom-Wall [33]	A multi-layered tool detects 0-Day versions of crypto-ransomware families by developing a generalized model comprised of static and dynamic traits	LR, SVM, NN, RF	• Exclusive for crypto-ransomware • Time-sensitive • High detection faults • Unaware of multi-class decision • Unaware of family attribution • Unaware of multiple datasets

Table 3. *Cont.*

Existing Tools	Method	Machine Learning Algorithm	Lacks
RANDS [34,35]	A hybrid machine learning-based anti-ransomware tool that detects 0-Day versions of both crypto and locker ransomware families by using dynamic traits on Windows platform	HMLC	• Unaware of static analysis • Unaware of family attribution • Unaware of manifold platforms • Unaware of multiple datasets
DRTHIS [36]	A three-fold machine learner that identifies crypto and locker ransomware versions among malware and good-ware versions using dynamic traits.	NN	• Unaware of static analysis • Unaware of family attribution • Unaware of advanced and new ransomware versions • Time-sensitive • Unaware of manifold platforms

4. Materials and Methods

This section describes the design and emulates the milestones of the proposed ransomware streaming analytics model to justify its sufficiency versus the lacks of the state-of-the-art anti-ransomware technology in terms of the compactness of traits, hybrid machine learner, and decision margins for ransomware characterization and family attribution.

4.1. Ransomware Characterization

Like malware families, ransomware families expose static traits such as digital signatures, built-in scripts, fuzzy functions, and hashes. They exhibit the ten static traits described in Table 4 to intrude the targeting system, infect its firewalls, and disable its restore settings. Also, they utilize static traits to encrypt data, APIs, files' content, and files' paths as well as spoofing the links to particular directories. The ten static traits presented in Table 4, are usually exploited in the versions of 14 ransomware families that this work deploys to implement and testify the proposed model. In the extraction tier that of a semi-realistic environment, the existence of encrypts, packers, hashes, and suspicious scripts are traced to inspect. Furthermore, any infections like altered filenames and directories, file system locations, bootstraps, and registry keys that might be done by a ransomware version are tracked through created trap files. Whereas, the 14 dynamic actions those leveraged by the versions of the 14 ransomware families (as described in Table 4) are traced to state their exploits through the accessing queries of files and directories, read/write/delete operations, edit the system's digital certification, modify system files' headers as well as the entropies of buffering data. Then, the raw data of all traced traits and actions of an examined version are formulated into an input vector of traits along with the class of that version as either ransomware R or non-ransomware R' to be computationally readable by the machine learner. The values of the trait vector are normalized into "0" s and "1" s according to the existence of their traits. The class label R in the trait vector is represented by "1" and the class label R' is represented by "−1". In contrast, any suspicious version that does not belong to R and R' (an imperviously examined version that might be malicious) is remarked as "0".

4.2. Ancestor Family Attribution and Multi-Descent Fusion

Consequently, the attribute of the ransomware family that the examined version might belong to is assumedly represented by two digits ranging from "01" to "99" as the header of the version's trait vector. Since 14 ransomware families are investigated for this work; therefore, the headers would range from "01" to "14". Any additionally adopted ransomware families will be assigned with the remaining digits of the assumed range in the future work. It is noteworthy to mention that the assumption "00" is assigned to the headers of all good-ware trait vectors, and "99" is assumedly assigned to the headers of all malware apps in the dataset. Then, a lookup table of the assigned headers is created as a tracing file for the numeral computation of *Attribution Rate* $(AR_{t_{i,j}})$ of every trait in a trait vector across other trait vectors involved in ransomware vectors and non-ransomware vectors, as in Equation (1). By product, the header data *"h"* of all characterized trait vectors are exploited to identify their relevance to a particular ransomware family among other families.

$$AR_{t_{i,j}} = \sum_{j=1}^{n} \frac{h_{t_{i,j} \rightarrow R} - h_{t_{i,j} \rightarrow R^{\sim}}}{h_{t_{i,j} \rightarrow R} + h_{t_{i,j} \rightarrow R^{\sim}}} \tag{1}$$

where R refers to the ransomware trait vectors and $R^{\sim}$ refers to all non-ransomware trait vectors in the batch of the dataset. Then, $AR_{t_{i,j}}$ is the frequency of each trait t_j belonging to a trait vector $t_{i,\,j}$ across all vectors encompassed in R and $R^{\sim}$. Unlikely, $h_{t_{i,j} \rightarrow R}$ is the frequency of that trait $t_{i,\,j}$ with respect to the headers of trait vectors in R. However, $h_{t_{i,j} \rightarrow R^{\sim}}$ is the frequency of the same trait $t_{i,\,j}$ with respect to all headers of trait vectors in $R^{\sim}$.

Table 4. Potential traits exploited by 0-Day versions of generic ransomware families.

		Ransomware Families													
Traits	**Type**	**Crypto-Wall**	**Win-Lock**	**Reventon**	**Crypto-Locker**	**Archiveus**	**GpCode**	**AiDS**	**RaaS**	**Cerber**	**Locky**	**Crysis**	**Wanna-Cry**	**Sopra**	**Zeus**
Windows API calls	Dynamic	✗	✔	✔	✗	✗	✗	✗	✗	✗	✔	✗	✗	✗	✗
Windows Cryptographic APIs	Dynamic	✔	✗	✔	✔	✔	✔	✗	✗	✔	✗	✔	✗	✔	✗
Registry Key	Dynamic	✔	✗	✔	✔	✗	✗	✗	✗	✗	✔	✗	✗	✗	✗
System File Process	Dynamic	✔	✗	✗	✔	✗	✗	✗	✔	✔	✔	✔	✗	✗	✗
Directory Actions	Dynamic	✔	✔	✗	✔	✗	✗	✗	✗	✗	✗	✔	✔	✔	✔
Application Folders	Dynamic	✔	✔	✗	✔	✗	✗	✗	✗	✗	✔	✔	✔	✔	✔
Control Panel Settings	Dynamic	✔	✗	✔	✔	✗	✗	✗	✗	✗	✔	✗	✔	✗	✔
System File Locations	Dynamic	✔	✗	✗	✔	✗	✗	✗	✔	✔	✔	✔	✗	✗	✗
Pay-loaders/Downloaders	Dynamic	✔	✗	✔	✔	✗	✗	✗	✗	✗	✔	✗	✔	✗	✔
Command and Control Server	Dynamic	✔	✔	✔	✔	✗	✗	✗	✔	✗	✔	✗	✔	✗	✔
Windows Volume Shadow (vssadmin.exe and WMIC.exe)	Dynamic	✗	✔	✔	✗	✗	✗	✗	✔	✗	✔	✗	✔	✗	✔
File Fingerprint	Dynamic	✔	✗	✗	✔	✗	✔	✗	✗	✔	✗	✔	✗	✗	✗
Directory Listing Queries	Dynamic	✔	✗	✗	✔	✗	✗	✗	✔	✔	✔	✔	✗	✗	✗
Windows Safe Mode Booting (bcdedit.exe)	Dynamic	✗	✔	✔	✗	✗	✗	✔	✔	✗	✔	✗	✔	✔	✔
File Extensions	Static	✔	✔	✔	✔	✔	✔	✗	✗	✔	✗	✔	✔	✔	✔
Files Names	Static	✗	✗	✗	✗	✔	✗	✔	✗	✔	✗	✔	✔	✔	✔
Portable Executable Header	Static	✗	✔	✔	✗	✗	✗	✔	✔	✗	✔	✗	✔	✔	✔
Embedded Resources	Static	✗	✔	✔	✗	✗	✗	✔	✔	✗	✔	✗	✔	✔	✔
Packers	Static	✗	✔	✔	✗	✗	✗	✔	✔	✗	✔	✗	✔	✔	✔
Shannon's Entropy	Static	✔	✗	✔	✔	✔	✔	✗	✗	✔	✗	✗	✗	✗	✗
Cryptors	Static	✔	✗	✔	✔	✔	✔	✗	✗	✔	✗	✗	✗	✗	✗
Portable Executable Signature	Static	✗	✔	✔	✗	✗	✗	✔	✔	✗	✔	✗	✔	✔	✔
Embedded Scripts	Static	✗	✔	✔	✗	✗	✗	✔	✔	✗	✔	✗	✔	✔	✔
Fuzzy Hashing	Static	✔	✗	✔	✔	✔	✔	✗	✗	✔	✗	✗	✔	✗	✗

On the other hand, the header data *"h"* of all characterized trait vector *"T_j"* would be checked-up across all extracted trait vectors from the batch of the dataset to categorize its mutual multi-descent to the other families of ransomware among malware and good-ware families as per Equations (2) and (3). This statistic probability is named *Multi* − *Descent Ratio* ($MDR(T_i)$), which prioritizes the ransomware versions that may belong to multi-descent families of ransomware with respect to their relative redundancy in malware and good-ware families on the learned trait vectors (T).

$$P_r(T_i \mid h_i) = \frac{N_{h_i \to R}}{N_{h_i \to R} + N_{h_i \to R^{\sim}}} \tag{2}$$

$$MDR(T_i) = \frac{\sum_{i=1}^{|K|} \Pr(T_i \mid h_i)}{|T_i|} \tag{3}$$

4.3. Ransomware Classification

As it is elaborated in Algorithm 1, two dominant machine learning algorithms, DT and NB, are synchronously hybridized in HML to optimize the adaptive categorization at the moment of tackling 0-Day versions of ransomware, malware, and good-ware. Since DT and NB are complements in their decisive functions and pruning margins, they are therefore hybridized to classify ransomware more accurately [28]. Conceptually, DT carries out a fast classification versus big training data throughout the tree structure such that the predictive classes of the input trait vectors can be arranged as the antecedent nodes, and their traits can be set as leaves of the tree. However, it might be ineffective in predicting the class of imperviously seen and relevant traits [27–29]. On the other hand, NB executes fast training of data; however, it is impractical against a big set of traits and heterogeneous trait values. However, it spends a short computation time in learning the training vectors of traits and predicting their actual classes by using Bayes' probabilistic theorem with the assumption that all the examined traits are independent of each other [27–29]. Thus, NB is applied by HML to trace the predictive classes of all overlooked traits in the indecipherable nodes of DT that optimizes the adaptive classification.

To do so, HML trains the fetching batch of trait vectors through cutting the decision edges of DT with NB pruning margins in an iterative splitting of the training trait vectors into sub-training vectors. Thus, the training trait matrix ($T = \{T_1, \ldots, T_K\}$) is given such that ($T_i = \left\{T_{i,j}\right\}_{i \in K,\ j \in |T_i|}$) with the predictive labels ($P_{class} = \{C_1, C_2\} :\ C_1 = 1,\ and\ C_2 = -1$). Each trait vector can be represented as ($T_i = \left\{C_m,\ T_{i,\ j}\right\}_{i \in |T_K|,\ m \in |C_M|}$. Then, the prior class probability $P(C_m)$ is computed as per Equation (4) to predict how often each class occurs over (T) relatively to the trait vector (T_i); whilst, the conditional probability of (T_i) is computed by Equation (5) to predict the relevance between the predictive class (C_m) and its corresponding trait ($T_{i,\ j}$) as it was indicated by ($P\left(T_{i,j} | C_m\right)$).

$$\mathrm{P}(\mathrm{T_i C_m}) = \mathrm{P}(\mathrm{C_m}) \prod_{\mathrm{e}=1 \to \mathrm{p}} \left(\mathrm{T_{i,j}} | \mathrm{C_m}\right) \tag{4}$$

$$\mathrm{C_m} = \mathrm{C_i} \to \mathrm{P_{ms}}(\mathrm{T_i},\ \mathrm{C_m}) \tag{5}$$

4.4. Structure of the Ransomware Multi-Tier Streaming Analytics Model

Figure 3 illustrates the structure of the proposed ransomware streaming analytics model. It consists of a trait extraction tier, an ancestor-family attribution tier, a multi-descent fusion tier, and a learning tier. The extraction tier works on the semi-realistic environment to analyze datasets (i.e., versions of ransomware, malware, and good-ware) into class-labeled trait space (i.e., a set of trait vectors with their class label). The semi-realistic environment is a virtual testbed used to create trap files (s) with realistic conditions, and it disseminates them into particular system directories. Once the trap (s) are exploited by ransomware, that ransomware runs its malfunctions and downloads its *crypto*-or *locker*-gadgets to either encrypt system data or halt the system. Correspondingly, the extracted trait vectors are attributed to their ancestor ransomware/non-ransomware families in the family attribution tier by normalizing the header data of each trait vector with respect to its family. Given that they assigned to their family attributes, all trait vectors are examined versus the case of a multi-descent family in the multi-descent fusion tier.

Algorithm 1

```
Definition of Semantic Codes
Let     $S$ the stream of ransomware and non-ransomware versions such that $S = \{S_m\}_{m\in|S|}$
        $T$ the compact set of traits
        $T_{space}$ the generated space of extracted traits where $T_{space} = \{T_i\}_{i\in|T|}$
        $T_i$ a trait vector included in $T_{space}$
        $TreeNode$ the decision tree of $T_{space}$
        $T_{sub}$ the splitting space of $T_{space}$ such that $T_{space} = \{T_{sub}\}_{sub\in|sub|}$
        $C_m$ the class model of $T_i$, where $C = \{C_m\}_{m\in M}$, and $M$ is the number of predictive classes
        $R$ the ransomware trait vectors (i.e., ransomware versions)
        $R^{\sim}$ are the non-ransomware trait vectors (i.e., non-ransomware versions)
        $F$ the trace file
        $AR_{t_{i,j}}$ the Attribution Rate of a trait vector
        $MDR(T_i)$ the multi-descent ratio of a trait vector
Input:  $S$ and $T$
Output: $R$ and $R'$
Begin
  1.  Generate $T_{space}$ from $S$ with headers
  2.  Repeat (3) to (c)
        a. Create $TreeNode$
  3.    b. IF (all $\{T\ i\}_{i\in|T|}$ in $T_{space}$ have similar class $C_m$) THEN $TreeNode \leftarrow LeafNode$
        c. Until $T = \{\}$ THEN attach $TreeNode$ to the majority class model $C_m$
  4.  For each $T_i$ in $T_{space}$
        a. find prior probability $C_j$ over $T_{space}$ by Equation (4)
        b. Find the conditional probability of $t_{i,\ j}$ about $C_j$ over $T_{space}$ by Equation (5)
        c. Update $T_i$ in $T_{space}$ with the maximal $P\left(t_{i,j}\middle|C_m\right)$ such that $P\left(C_{jm}\middle|t_{i,j}\right)$; $C_m \rightarrow P_{ml}\left(C_m\middle|t_{i,j}\right)$
        d. Partition $T_{space}$ into $T_{space} = \{T_{sub}\}_{sub\in|T|}$ and $T_{space} \leftarrow \{T_{sub}\}_{sub\in|T|}$
  5.  Repeat (6) Until ($T_{space} \neq \{\}$) AND ($T \neq \{\}$)
  6.  Keep all computed probabilities in $R$ and $R^{\sim}$ for classification decision
      For each $T_i$ in $R$ and $R^{\sim}$
        a. find $AR_{t_{i,j}}$
  7.    b. find $MDR(T_i)$
        c. keep $AR_{t_{i,j}}$ and $MDR(T_i)$ in the trace file $F_i$
End
```

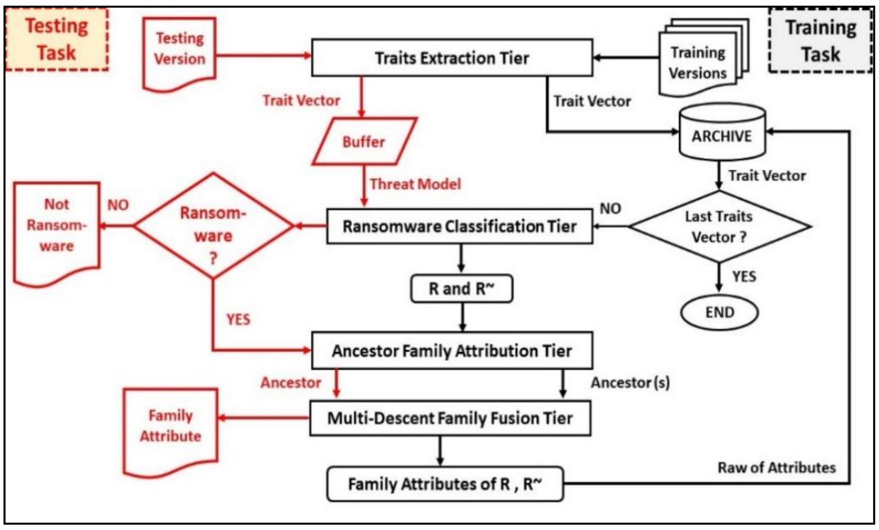


Figure 3. Structure of multi-tier ransomware streaming analytics model.

The diagnoses of ancestor family and multi-descent families that are pursued by family attribution and multi-descent fusion tiers are implemented synchronously along with learning done by the hybrid machine learner in the learning tier. So far, the proposed ransomware streaming analytics model carries out the aforesaid multi-tiers in a multi-disciplinary manner across data, trait space, header data, and predictive classes during both the training and testing tasks, as shown in Figure 3. Thus, the input (unknown) ransomware version is analyzed on its runtime during the testing task of the proposed multi-tiered streaming analytics architecture as the same as the data of the training corpus but in a different order. The training task can be traced by the black path, whereas the testing task is tracked throughout the red path, as it can be seen in Figure 3.

5. Experiments and Results

This section elaborates the experimental workflow that is conducted for the purpose of performance assessment along with the description of data corpora and evaluation metrics as well as the discussion of obtained results. As illustrated in Figure 4, the experimental workflow is conducted through several tasks, including a collection of data, construction of data into training corpus and testing corpus, implementation of the benchmarking machine learners as well as the proposed machine learner (HML). In addition, the experimental workflow pursues the implementation task of the benchmarking anti-ransomware tools and the proposed ransomware multi-tiered streaming analytics model. Then, the evaluation task is carried out to qualify the overall experimental results and to justify the asserted findings.

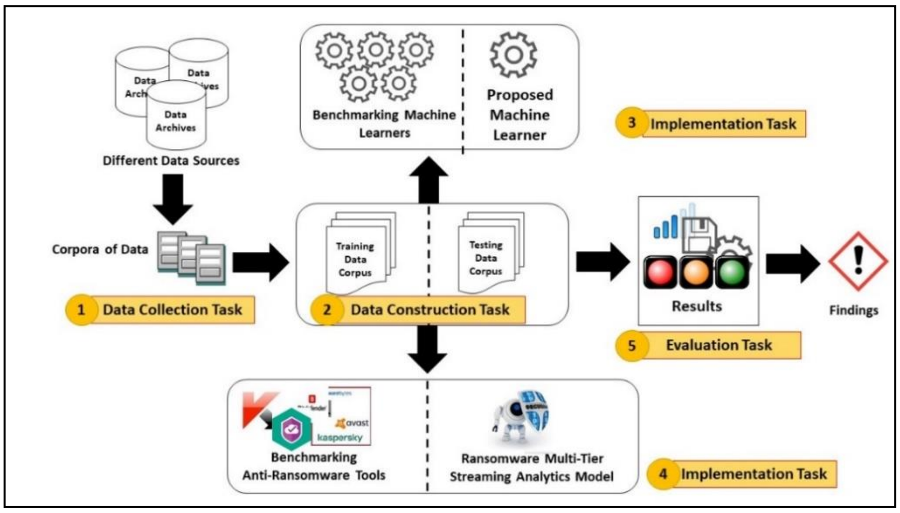


Figure 4. Experimental workflow.

5.1. Runtime Test Routine

Figure 5 shows how the proposed multi-tier streaming analytics model runs its run-time test routine to detect a ransomware version in a realistic environment. In Figure 5, a suspicious version downloads itself onto the targeting computer system when the user browses the web. To do so, the suspicious attack uses many toolkits for this purpose, such as Trojans, spoofing links, and downloading software and apps. Meanwhile, the suspicious version tackles the trap files created by the proposed model during the extraction tier. It activates its static and dynamic traits to infect the trap files. Then, a scan disk is implemented to check-up the actions and infections on the computer system. Whenever infections are observed, they are analyzed statically and characterized dynamically. Accordingly, the ancestor families are attributed, and multi-descent ransomware versions are fused to be ready for further learning by the hybrid machine learner. Consequently, the class label of the fetched version is predicted throughout the learning tier. By tracking and analyzing the observed actions, the proposed model attempts to determine to what ancestor family this version does belong? Thus, it affirms that the suspicious version is either ransomware, or malware, or good-ware. Finally, it either warns the user by a popped up message or; it pops-up a safe acknowledgment to the user's screen.

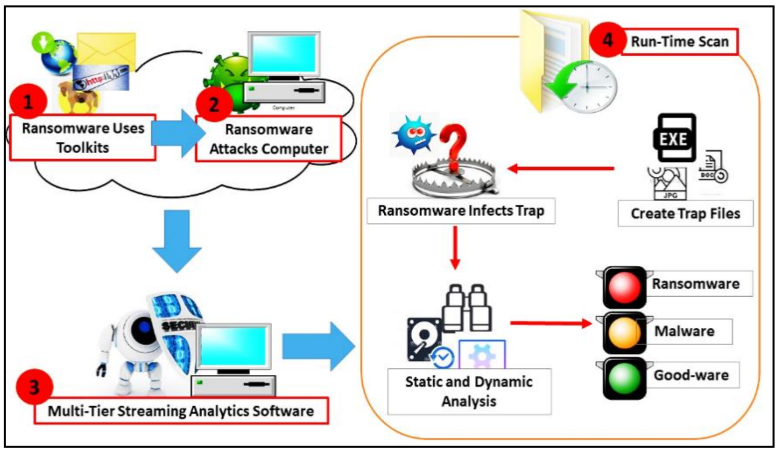


Figure 5. The run-time test routine of the multi-tier streaming analytics model.

5.2. Data Collection and Evaluation Metrics

By searching publically used archives like the Virus Total Intelligence Platform and Virus Share, the executable and portable ransomware and malware versions are aggregated to construct data corpora. Exclusively, our data aggregation targets those versions that are submitted at least three times to the aforesaid data archives between 1 January 2019 and 1 September 2019, and they are still undetected by any existing antivirus software and/or anti-ransomware tools. Similarly, the good-ware instances are aggregated and merged randomly to the aforesaid data corpora based on particular heuristics of benign software apps. As described in Table 5, the aggregated data corpora are divided randomly into $\frac{2}{3}^{nd}$ and $\frac{1}{3}^{rd}$ as training data corpus and testing data corpus to be used in training and testing tasks, respectively. On the other hand, the rates of true positive (TPR), false positive (FPR), false negative (FNR), and classification accuracy as well as mistake rate, miss rate, and elapsed time; all are used as standard performance evaluation metrics for the evaluation task in recently published works [11,27–29,34,35]. TPR, FPR, and FNR are derived from the confusion matrix calculations as follows:

$$TPR = \frac{TP}{TP + FN} \tag{6}$$

$$FPR = \frac{FP}{TN + FP} \tag{7}$$

$$FNR = \frac{FN}{TP + FN} \tag{8}$$

where TPR refers to the rate of correctly classified ransomware data, FPR indicates the rate of wrongly classified good-ware data as ransomware, and FNR indicates the rate of wrongly labeled ransomware data as good-ware data, respectively. While TP is the number of good-ware samples classified as ransomware, FN is the number of ransomware samples classified as good-ware, TN is the number of ransomware samples that correctly classified, and FP is the number of good-ware samples that classified as ransomware; respectively.

Based on the above mentioned standard metrics, the classification accuracy rate (ACCR) is calculated to validate the effectiveness of the applied machine learner and/or anti-ransomware tool at detecting valid ransomware (TP) and valid good-ware samples (TN) relatively to the whole data corpora as follows:

$$ACCR = \frac{TP + TN}{TP + FP + TN + FN} \tag{9}$$

Accordingly, mistake rate is computed to qualify the abilities of the comparable anti-ransomware tools as well as the proposed ransomware multi-tier streaming analytics model on how they rationally detect the valid ransomware versions with least false classifications as per Equation (10); whereas, the miss rate is computed to qualify their abilities to rationally detect valid ransomware versions with least misclassification cost as per Equation (11).

$$\text{Mistake Rate} = \frac{\text{FP}}{\text{N}_\text{G}} \tag{10}$$

$$\text{Miss Rate} = \frac{\text{FN}}{\text{N}_\text{R}} \tag{11}$$

where, N_G and N_R are the number of good-ware data and ransomware data, respectively.

In addition, the Elapsed Time plays an important to assert how long the comparable machine learners and anti-ransomware tools spend to execute and respond by examining a batch of data with a nominal cost of computations [35–37]. On the other hand, the proposed HML is assessed against its competitor machine learners in terms of AUC against the up-to-date ransomware, malware, and good-ware data during the real test. AUC calculation is widely used to justify the performance of a machine learning algorithm on experimental data by devoting the scalar value of the receiver operating characteristic curve (ROC), which is a plot of TPR versus FPR [35,37,38]. If the AUC score closes to 0.9, then it signifies an excellent performance in the realistic practice; while score values of 0.8, 0.7, and less could signify good, moderate, and then poor performance [38,39].

Table 5. The corpora of data.

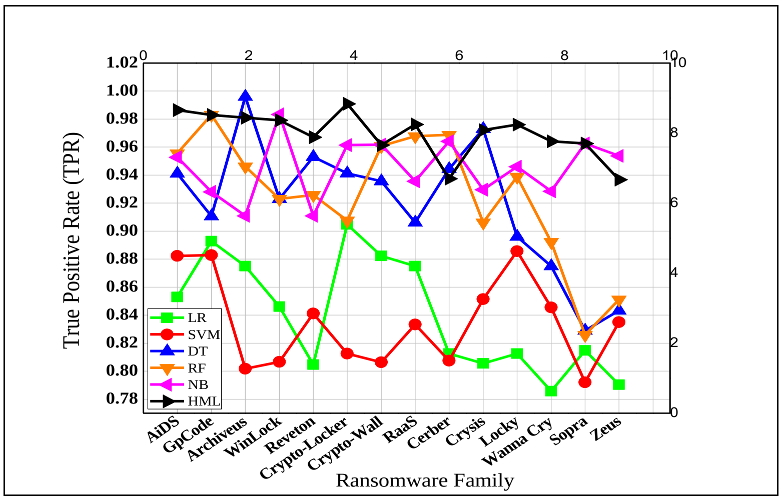
Description		Corpus
Number of Valid Ransomware Versions		35,000
Number of Malware Versions		500
Number of Good-ware Versions		500
Data Archives		[40–42]
Aggregation Time		1/1/2019–1/9/2019
Training Data Corpus		17,332
Testing Data Corpus		17,666
Ransomware Families	AiDS	4000
	GpCode	8000
	Archiveus	1500
	WinLock	3620
	Reveton	2400
	CryptoLocker	1720
	CryptoWall	3250
	RaaS	1300
	Cerber	1535
	Locky	2000
	Crysis	1320
	WannaCry	1300
	Sopra	1570
	Zeus	1500

Thus, overall performance metrics are used during performance evaluation task that involves comparative experiments, and realistic experiments. Such experiments quantify how often and how long the proposed machine learner and the multi-tier ransomware streaming analytics model takes for ransomware detection and ransomware families' categorization versus the dominant machine learners and the benchmarking anti-ransomware tools. Both comparative and real-time experiments are evaluated by using the corpora of training and testing data. It is worthy of mention that simulated machine learners in Weka, as well as soft computing by Python, are used for the conducted experiments.

5.3. Comparative Experiment

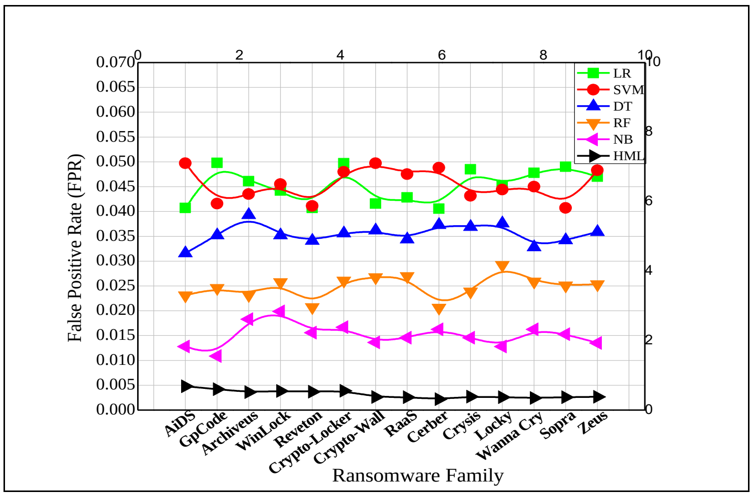
Three comparative experiments are conducted to address several problematic issues of machine learners-based anti-ransomware technology like the rich set of traits to classify ransomware, ransomware family attribution, and learn big and imbalanced corpora of data. The first comparative experiment is conducted in between to validate the efficacy of the proposed hybrid machine learner (HML) against state-of-the-art machine learners, including LR, SVM, RF, DT, and NB. On the other hand, the second experiment is conducted to compare the proposed ransomware streaming analytics model versus the most salient signature-based anti-ransomware tools like *BitDefender*, misuse-based anti-ransomware tools like *R-Locker*, and the machine learner-based anti-ransomware tools such as *EldeRan* and *RANDS*. As shown in Figures 6 and 7, HML outperforms significant TPRs, FPRs, and FNRs versus all ransomware versions of the 14 families.

Consequently, HML contributes the proposed ransomware streaming model progressively to classify ransomware versions effectively, among other examined anti-ransomware tools. Plots of Figures 6 and 7 demonstrate how the previous questionable issues can be improved to enrich ransomware classification throughout (i) handling variety, heterogeneity, and quantity of the employed set of traits (static and dynamic traits), (ii) leveraging the commonness among ransomware families, (iii) learning a scalable and variable corpus of data adaptively, and (iv) affecting by different emerging and aggregation time of the input versions. Furthermore, plots of Figures 6d and 7d demonstrate the efficiency of the proposed HML and then the proposed ransomware streaming model within its multi-tiered design in time, and light-weight use of computer resources. The attitudes above could be decisive factors to escalate the detection accuracy and lessen the false detections against ransomware versions.

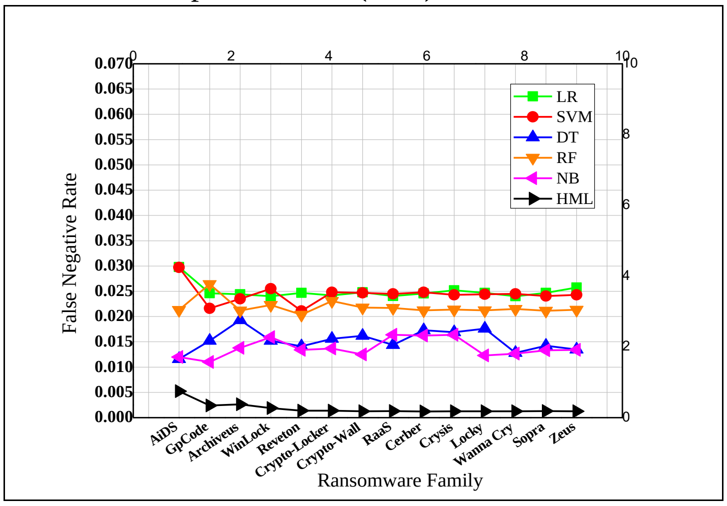


(a) True positive rate (TRR) vs. ransomware families

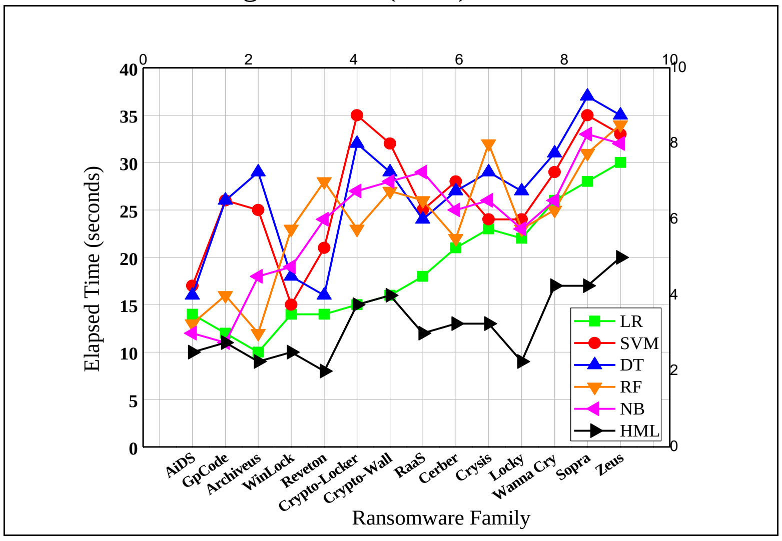
Figure 6. *Cont.*



(b) False positive rate (FPR) vs. ransomware families

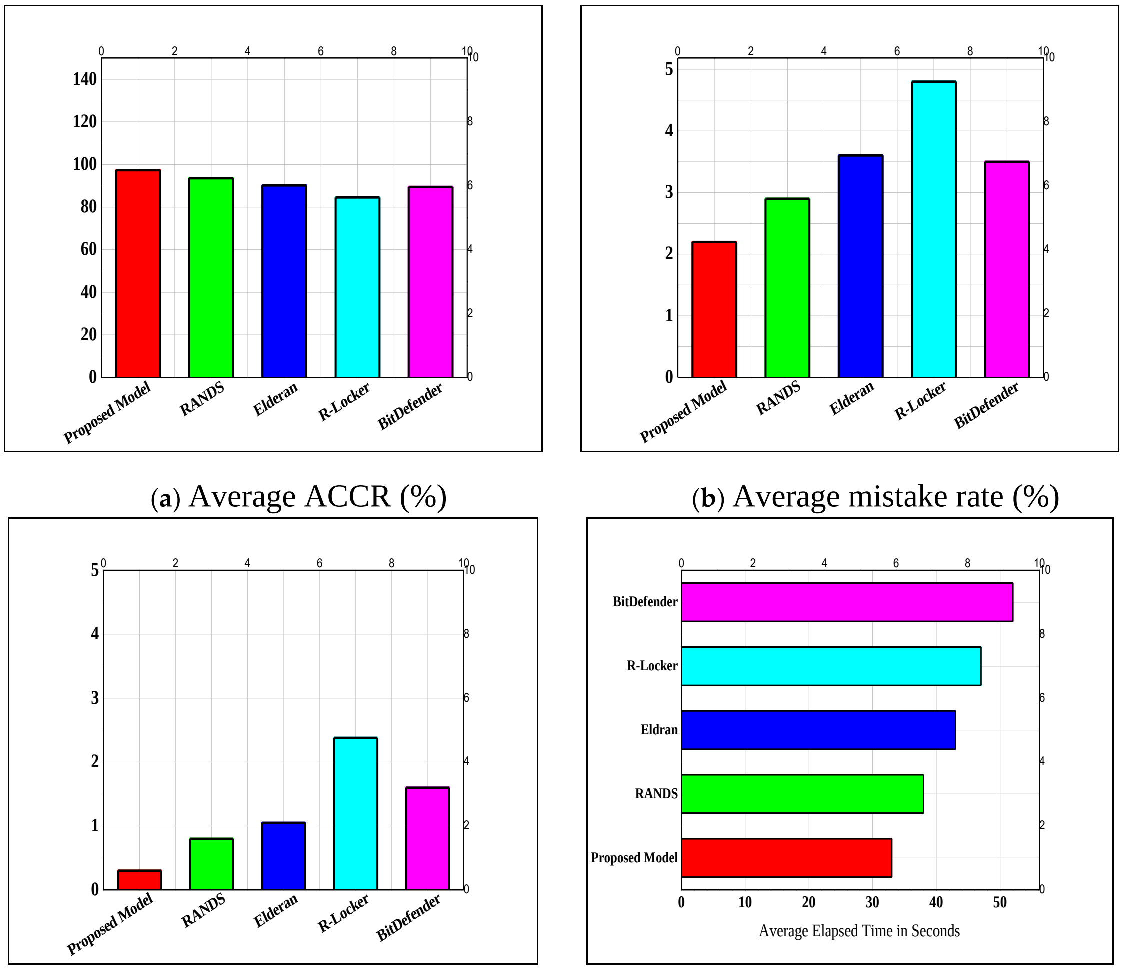


(c) False negative rate (FNR) vs. ransomware families



(d) Cost of elapsed time vs. ransomware families

Figure 6. Evaluation of the proposed machine learner against state-of-the-art machine learners.



(**a**) Average ACCR (%) (**b**) Average mistake rate (%)

(**c**) Average miss rate (%) (**d**) Elapsed time consumption

Figure 7. Evaluation of the proposed ransomware streaming analytics model versus different examples of anti-ransomware tools.

So far, we attempt to investigate how do state-of-the-art machine learners, as well as the proposed HML, can leverage the issues above in the realistic environment against 0-Day ransomware versions. Such investigation points out the effects of real-life data corpus that might contain many different and daily emerged versions of cyber-attacks, including ransomware, malware, and good-ware. Almost cyber-attacks are of a multi-descent family; for example, malicious and ransomware attacks. The existence of cyber-attacks causes an abundance of attacks' versions, imbalance in those versions' classes, variety in versions' ancestor families, and versions' multi-descent families as well as their commonness in their latent traits and dynamic behaviors that are suggested by this work. Thus, a third comparative experiment is conducted in the realistic mode during one month (specifically from 1 November to 30 November 2019), and its outcomes are evaluated by the scores of AUC as they are plotted in Figure 8.

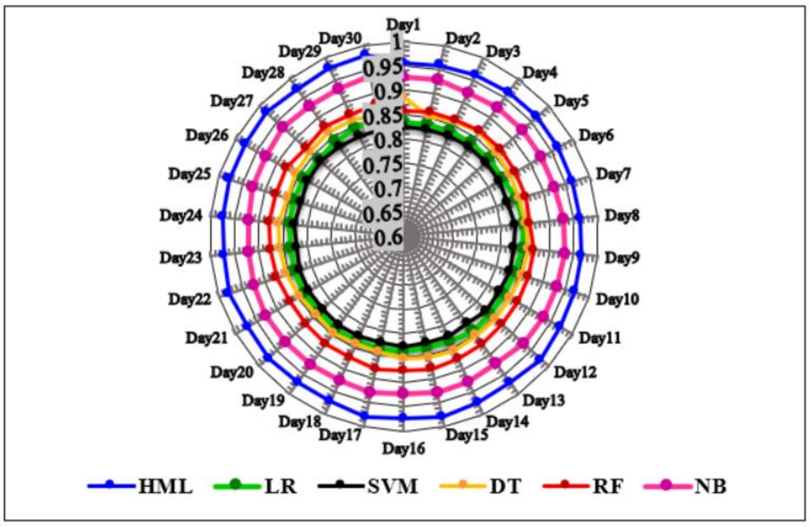
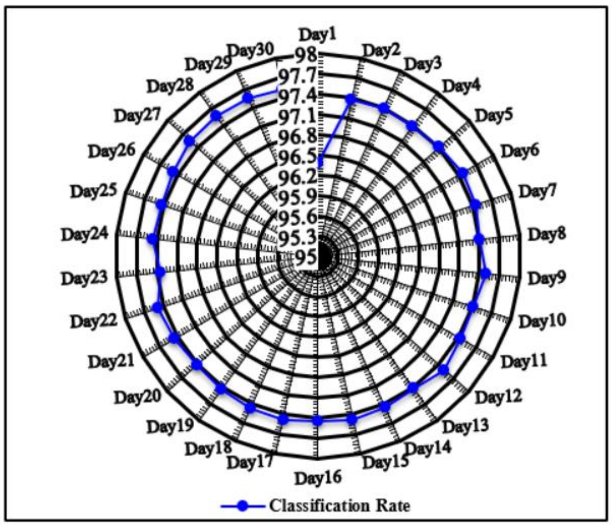


Figure 8. Outcomes of the third comparative experiment in a realistic environment with respect to AUC scores.

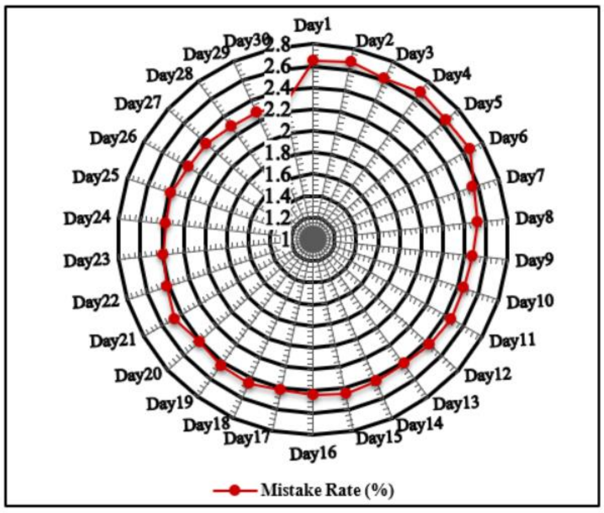
Unlike the proposed HML, the state-of-the-art machine learners achieve approximately high to moderate scores of AUC at classifying 0-Day ransomware versions among other cyber-versions during the month of the test (see Figure 8). AUC scores state that the examined machine learners require more crucial functions with probably regulating margins to adapt 0-Day versions involved in such real-life data corpus. That is due to some cases like that any suspicious version might be classified as invalid ransomware, or invalid good-ware, or invalid malware, or a new version of ransomware inaccurately. Furthermore, the examined machine learners show their consumption of changeable periods of time and complex computations to charactserize every version as well as recognizing its identity and its probability of a multi-descent family.

5.4. *Realistic Experiment*

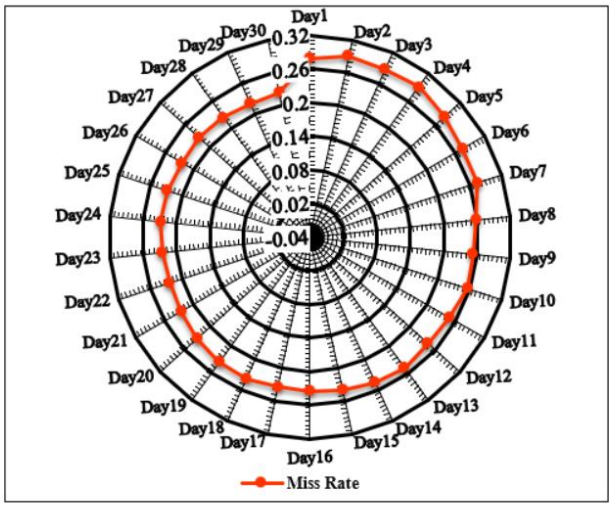
Besides the previous comparative experiments, a daily based test is conducted in a realistic environment to evaluate how the proposed ransomware streaming analytics model manifests its efficacy against different and 0-Day ransomware versions of those of different ancestor-families and multi-descent family. The plotted charts of Figure 9 demonstrate the detection ability and holistic characterization of the proposed ransomware streaming analytics model with minimal performance overhead through its multi-tiered design versus daily escalating/deescalating and/or imbalanced corpus of data during one month. Unlike the anti-ransomware tool *RANDS*, which is devoted to our previous work [34,35], a minor escalation or de-escalation of efficacy's trend line is reported by our proposed ransomware streaming analytics model at certain days (see Figure 9). This is due to its attitude in solving the problems of the multi-descent family case as well as ancestor family attribution by using numeral and statistic metrics that are pursued by the proposed HML. Furthermore, the proposed ransomware streaming analytics model achieves shorter elapsed time than that of *RANDS* in [34], as shown in Figure 9d, due to its minor consumption of CPU and memory.



(**a**) Classification Accuracy Rate

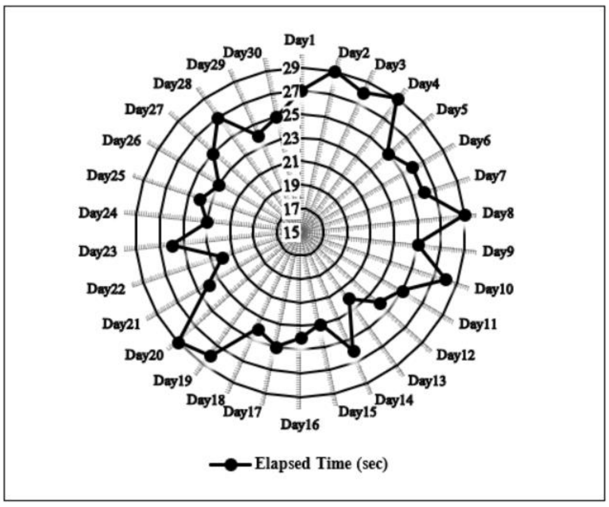


(**b**) Mistake Rate



(**c**) Miss Rate

Figure 9. *Cont.*



(**d**) Elapsed Time

Figure 9. Evaluation outcomes of the proposed ransomware streaming analytics model in the realistic test.

6. Discussion

The previous experiments have pointed out "why the state-of-the-art anti-ransomware tools those assisted by machine learners are still insufficient to detect ransomware versions of different ancestor families and multi-descent families in the realistic environment?" The answers to this question can be summarized as follows:

- Shortage in static analysis. Not all the state-of-the-art machine learners leverage static traits to characterize versions of all-inclusive ransomware families. Indeed, many ransomware families might exploit more adversarial scripts to intrude on their target operating systems [6,24]. Thus, the examined machine learners showed their defeatism against static exploits of ransomware, and they achieved low to sensible TPRs along with nontrivial FPRs and FNRs, as shown in Figures 6 and 8. In addition, some of the examined anti-ransomware tools revealed partial characterization of such static exploits on data corpora, as shown in Figure 7.
- Shortage in dynamic analysis. Tracking the behavioral traits, executed server controls, I/O resources, buffers, system activity, file events, CPU processes, and memory usage; all together, these are more distinctive to classify ransomware versions than the static exploitations [13,15,21]. However, their analysis and extraction require a virtual testbed which must be designed under realistic conditions to trigger ransomware payloads, linking libraries, and access permissions. As shown in Figure 6, the state-of-the-art machine learners fall short in tracing a ransomware's infection chain, and they produced trivial FPRs because they are mostly devoted to semi-realistic environments and lack the necessary triggers. Then, they are rendered vulnerable against any 0-Day ransomware versions during the second comparative experiment in realistic practice (see Figure 7).
- Elapsed time-accuracy trade-off. The trade-off between elapsed time and classification accuracy is a double-edges sword in machine learning [37]. Almost all, the examined machine learners and anti-ransomware tools achieved acceptable accuracy rates but at unacceptable elapsed time for detecting an incremental stream of training and testing corpora (see Figures 6 and 7). Elapsed time is a crucial factor to consider in ransomware detection since its computation depends on the traits' relevance and redundancy to the data corpus, the number of traits to be extracted from data corpus, the dimensionality of data corpus to learn, the extensive utilization of device's resources like CPU and memory and the computation complexity of the machine learning algorithm during both learning and testing tasks. Thus, some machine learners showcase high accuracy rates, but they spend a long elapsed time to obtain the detection results.

- Unaware of multi-descent versions of ransomware. The training and testing corpora of versions might involve versions of other malware and scareware families that probably share similar and mutually inter-related static exploits and dynamic behaviors and functionalities [6,24]. This may confuse machine learners, which leverage binary-class inductive function rather than a multi-class inductive function [36,37]. As a result, almost examined machine learners achieved high to moderate false classifications versus valid ransomware versions; those are descents of multiple ransomware and/or malware families, as shown in Figures 6–8. Hence a manifold induction could be attained by hybridizing multiple machine learners and/or statistic pruning formularization as it is adopted in HML
- Imbalanced corpora of data. Experimentally, the state-of-the-art machine learners produced their own predictions by learning the employed set of traits on the corpus of training data that was (i) various in the versions of other cyber-attacks like scareware and malware, etc. (ii) sub-optimally representative of ransomware detection, (iii) imbalanced in ransomware family population and ransomware-type distribution (i.e., crypto and locker), and (iv) varied in aggregation time. Altogether, it caused substantial FPRs, FNRs, and insignificant AUCs, as shown in Figures 6 and 7, respectively. Similarly, the examined anti-ransomware tools achieved incompetent outcomes versus the aforesaid corpus of data, as shown in Figure 8.
- Unreliable source of data. Most of the popular archives of ransomware fall short in providing reliable and unique ransomware datasets that is a complementary factor of boosting the implicit ransomware classification against 0-Day ransomware versions and imperviously detected ransomware families [4,6,7,12]. It is observed in Figures 6 and 7 that predicting valid ransomware versions across unreliable training corpus of data could be crucial to adjust the decision margins of the examined machine learners. Hence, resembling a reliable training corpus by re-learning the default predictions would solve the problem of substantial FPRs [37,38].
- Unaware of family attribution. Escalated ransomware streams from 2005 to 2019 yield more than 30 different ransomware families of both locker and crypto types [21,24,34,35]. This is attributed to the availability of e-services, social engineering, and primitive ciphering technologies that enable cyber-criminals to advance their ransomware without sophisticated knowledge [21,24]. However, the big stream of ransomware versions might share mutually inter-related traits that cause overfitting to decisive margins of machine learners and then limited detection against some ransomware families with high rates of FNR, as shown in Figure 6.
- Realistic and semi-realistic environment. The comparable machine learners, except HMLC that is devoted by [34,35], are still inactive to learn the misclassified versions of ransomware, and they achieve sub-optimal AUCs (see Figure 9). This is attributed to their decisive default settings, which fall short in minimizing the future cases of their predictions, which rely on their high rates of falsely detected versions versus their low rates of the truly detected versions. By product, the examined machine learners are still unaware of imperviously classified ransomware families on chronologically increasing ransomware stream.

The aforesaid observations raise another question to answer: "Why and how did our proposed HML and then our proposed multi-tier ransomware streaming analytics model outperform the state-of-the-art machine learners and anti-ransomware technologies?" The answer is summarized as follows:

- Set of traits. Great care was put on exploring and utilizing the most generic and evolutionary traits of ransomware families. Twenty-four different traits of both static-type and dynamic-type are used to provide a holistic characterization of ransomware versions among the versions of other cyber-attacks. Thus, our proposed work was privileged at classifying ransomware in the semi-realistic and realistic modes as it is observed from the experimental outcomes in Figures 6–9.
- Hybrid machine learner. The evaluation outcomes in Figures 6–9 restated that the proposed HML was more competent among its competitors. This was attributed to (i) its dual inductive function that is a hybrid of NB and DT functions, and (ii) its adjustable decisive boundary that is

complementary of numeral and statistic metrics, *AR* and *MDR*. Accordingly, it achieved higher detection rates than others against the versions of ever-seen and/or never-seen ransomware families on big and different corpora of data.

- Training and testing corpora of data. The corpora of data were collected during a one-year period of time at different runtimes Ŧ. Due to this periodic aggregation of data corpora every Ŧ, the training, and testing tasks would be less biased versus the implicit and explicit class distribution problem in the examined data stream. Moreover, the data corpora were privileged in terms of quantity and variety of versions 45 K of ransomware, malware, and good-ware versions), quantity and variety of families (14 ransomware families), the difference of aggregation time, the difference of data archives, and imbalance of versions belong to common ancestors as well as the multi-descent versions (see Figures 6–9).
- Realistic conditions. To customize the overall performance, the proposed ransomware streaming analytics model with the presence of the proposed HML was run with the assumption of learning a corpus of data at its actual runtime Ŧ. Hence, a suspicious version *R* that was inspected at time *Ŧ* could be classified as a ransomware version in the future runtime iteration of (Ŧ + Δ). That, in turn, would control the trade-off between the elapsed time, which relied on both CPU and memory consumption of the hosted system and the rate of detection accuracy in the realistic environment (see Figures 8 and 9).

7. Conclusions and Future Work

By studying the performance trade-offs across the state-of-the-art machine learners-based anti-ransomware tools experimentally, this paper affirms that they were computationally insufficient to classify 0-Day versions of different ransomware families on 40K corpora of data throughout in semi-realistic and realistic environments. They were still limited in static and dynamic analyses, set of traits to extract, type of ransomware to examine, the number of versions to learn, variety of ancestor ransomware families to attribute, multi-descent versions to recognize, the significant and imbalanced data stream to analyze, semi-realistic and realistic testbeds, decisive margins to prune, and inductive functions to adjust. Thus, this paper devotes a multi-tier ransomware streaming analytics model, which is empowered by a rich set of 24 static and dynamic traits, a novel machine learner, statistic, and numeral rates of family attribution and multi-descent family fusion those overlooked by the state-of-the-art anti-ransomware technologies. The proposed solution pursues four tiers of traits extraction, ransomware classification, ancestor family attribution, and multi-descent family fusion to discriminate ransomware versions from malware and good-ware versions. Experiments have qualified how the proposed solution enriches the accuracy, reduces the mistakes and misclassifications, and shortens the elapsed time versus escalating, big, lifelike, and imbalanced corpora of data. Overall results that averaged by 97% of accuracy rate, 2.4% of mistake rate, and 0.34% of miss rate; justify its maximal efficacy and cost-efficiency among its competitors.

For future improvement, it is recommended to explore more distinctive traits to classify the imperviously seen ransomware families by investigating versions of other ransomware families. Herewith, 14 *crypto*-type, and *locker*-type ransomware families are investigated with 24 traits solely. Furthermore, a hybrid machine learner of other base machine learners rather than NB and DT could be designed as either a single-based learner or an ensemble-based learner for pivoting an ideal decision that could satisfy the aforesaid cases.

Author Contributions: Conceptualization, H.Z.; methodology, H.Z.; software, H.Z.; validation, H.Z. and A.S.; formal analysis, A.S., and O.K.; investigation, A.S. and O.K.; resources, H.Z.; data curation, H.Z.; writing—original draft preparation, H.Z.; writing—review and editing, A.S. and O.K.; visualization, H.Z.; supervision, A.S. and O.K.; project administration, A.S. and O.K.; funding acquisition, A.S. and O.K. All authors have read and agreed to the published version of the manuscript.

Funding: The Article Processing Charge was funded by the SPEV project 2102/2020, Faculty of Informatics and Management, University of Hradec Kralove.

Acknowledgments: This research was partially funded by Universiti Teknologi Malaysia (UTM) under Research University Grant Vot-20H04, Malaysia Research University Network (MRUN) Vot 4L876, and the Fundamental Research Grant Scheme (FRGS) Vot 5F073 supported under the Ministry of Education Malaysia. The work is partially supported by the SPEV project, (ID: 2102-2020), "Smart Solutions in Ubiquitous Computing Environments" Faculty of Informatics and Management, University of Hradec Kralove. We are also grateful for the support of Ph.D. students Michal Dobrovolny and Ayca Kirimtat in consultations regarding application aspects from Hradec Kralove University, Czech Republic.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Bhardwaj, A.; Avasthi, V.; Sastry, H.; Subrahmanyam, G.V.B. Ransomware digital extortion: A rising new age threat. *Indian J. Sci. Technol.* **2016**, *9*, 1–5. [CrossRef]
2. Richardson, R.; North, M. Ransomware: Evolution, mitigation and prevention. *Int. Manag. Rev.* **2017**, *13*, 10–21.
3. Azmoodeh, A.; Dehghantanha, A.; Conti, M.; Choo, K.K.R. Detecting crypto-ransomware in IoT networks based on energy consumption footprint. *J. Ambient Intell. Humaniz. Comput.* **2018**, *9*, 1141–1152. [CrossRef]
4. Al-rimy, B.A.S.; Maarof, M.A.; Shaid, S.Z.M. Ransomware threat success factors, taxonomy, and countermeasures: A survey and research directions. *Comput. Secur.* **2018**, *74*, 144–166. [CrossRef]
5. Tailor, J.P.; Patel, A.D. A comprehensive survey: Ransomware attacks prevention, monitoring and damage control. *Int. J. Res. Sci. Innov.* **2017**, *4*, 2321–2705.
6. Kok, S.; Abdullah, A.; Jhanjhi, N.; Supramaniam, M. Ransomware, threat and detection techniques: A review. *Int. J. Comput. Sci. Netw. Secur.* **2019**, *19*, 136.
7. Yaqoob, I.; Ahmed, E.; Rehman, M.H.; Ahmed, A.I.A.; Al-Garadi, M.A.; Imran, M.; Guizani, M. The rise of ransomware and emerging security challenges in the Internet of Things. *Comput. Netw.* **2017**, *129*, 444–458. [CrossRef]
8. Pathak, P.B.; Nanded, Y.M. A dangerous trend of cybercrime: Ransomware growing challenge. *Int. J. Adv. Res. Comput. Eng. Technol.* **2016**, *5*, 371–373.
9. Herrera Silva, J.A.; Barona López, L.I.; Valdivieso Caraguay, Á.L.; Hernández-Álvarez, M. A survey on situational awareness of ransomware attacks—detection and prevention parameters. *Remote Sens.* **2019**, *11*, 1168. [CrossRef]
10. Zavarsky, P.; Lindskog, D. Experimental analysis of ransomware on windows and android platforms: Evolution and characterization. *Proced. Comput. Sci.* **2016**, *94*, 465–472.
11. Sgandurra, D.; Muñoz-González, L.; Mohsen, R.; Lupu, E.C. Automated dynamic analysis of ransomware: Benefits, limitations and use for detection. *arXiv* **2016**, arXiv:1609.03020.
12. Kok, S.H.; Abdullah, A.; Jhanjhi, N.Z.; Supramaniam, M. Prevention of crypto-ransomware using a pre-encryption detection algorithm. *Computers* **2019**, *8*, 79. [CrossRef]
13. Morato, D.; Berrueta, E.; Magaña, E.; Izal, M. Ransomware early detection by the analysis of file-sharing traffic. *J. Netw. Comput. Appl.* **2018**, *124*, 14–32. [CrossRef]
14. Sihwail, R.; Omar, K.; Ariffin, K.A.Z. A survey on malware analysis techniques: Static, dynamic, hybrid and memory analysis. *Int. J. Adv. Sci. Eng. Inf. Technol.* **2018**, *8*, 1662. [CrossRef]
15. Stiborek, J.; Pevný, T.; Rehák, M. Probabilistic analysis of dynamic malware traces. *Comput. Secur.* **2018**, *74*, 221–239. [CrossRef]
16. Cybersecurity, K.E. The Protection Technologies of Kaspersky Endpoint Security. Available online: https://mediacircle.de/pdf/Protection_Technologies_Whitepaper.pdf (accessed on 3 March 2020).
17. Kharraz, A.; Kirda, E. Redemption: Real-time protection against ransomware at end-hosts. In *International Symposium on Research in Attacks, Intrusions, and Defenses*; Springer: Cham, Switzerland, 2017; pp. 98–119.
18. Kharaz, A.; Arshad, S.; Mulliner, C.; Roberson, W.K.; Krida, E. UNVEIL: A large scale, automated approach to detecting ransomware. In Proceedings of the 2017 IEEE 24th International Conference on Software Analysis, Evolution and Reengineering (SANER), Klagenfurt, Austria, 20–24 February 2017; pp. 757–772.
19. Gómez-Hernández, J.A.; Álvarez-González, L.; García-Teodoro, P. R-Locker: Thwarting ransomware action through a honeyfile-based approach. *Comput. Secur.* **2018**, *73*, 389–398. [CrossRef]
20. Cabaj, K.; Gregorczyk, M.; Mazurczyk, W. Software-defined networking-based crypto ransomware detection using HTTP traffic characteristics. *Comput. Electr. Eng.* **2018**, *66*, 353–368. [CrossRef]

21. Hampton, N.; Baig, Z.; Zeadally, S. Ransomware behavioural analysis on windows platforms. *J. Inf. Secur. Appl.* **2018**, *40*, 44–51. [CrossRef]
22. Feng, Y.; Liu, C.; Liu, B. Poster: A new approach to detecting ransomware with deception. In Proceedings of the 38th IEEE Symposium on Security and Privacy, San Jose, CA, USA, 22–24 May 2017.
23. Cimitile, A.; Mercaldo, F.; Nardone, V.; Santone, A.; Visaggio, C.A. Talos: No more ransomware victims with formal methods. *Int. J. Inf. Secur.* **2018**, *17*, 719–738. [CrossRef]
24. Zhang, H.; Xiao, X.; Mercaldo, F.; Ni, S.; Martinelli, F.; Sangaiah, A.K. Classification of ransomware families with machine learning based on N-gram of opcodes. *Future Gener. Comput. Syst.* **2019**, *90*, 211–221. [CrossRef]
25. Alhawi, O.M.; Baldwin, J.; Dehghantanha, A. Leveraging machine learning techniques for windows ransomware network traffic detection. In *Cyber Threat Intelligence*; Advances in Information Security (ADIS, Volume 70); Springer: Cham, Switzerland, 2018; pp. 93–106.
26. Bae, S.I.; Lee, G.B.; Im, E.G. Ransomware detection using machine learning algorithms. *Concurr. Comput. Special Issue* **2016**. [CrossRef]
27. Aburomman, A.A.; Reaz, M.B.I. A survey of intrusion detection systems based on ensemble and hybrid classifiers. *Comput. Secur.* **2017**, *65*, 135–152. [CrossRef]
28. Tsai, C.-F.; Hsu, Y.-F.; Lin, C.-Y.; Lin, W.-Y. Intrusion detection by machine learning: A review. *Expert Syst. Appl.* **2009**, *36*, 11994–12000. [CrossRef]
29. Shabtai, A.; Moskovitch, R.; Elovici, Y.; Glezer, C. Detection of malicious code by applying machine learners on static features: A state-of-the-art survey. *Inf. Secur. Tech. Rep.* **2009**, *14*, 16–29. [CrossRef]
30. Continella, A.; Guagnelli, A.; Zingaro, G.; De Pasquale, G.; Barenghi, A.; Zanero, S.; Maggi, F. ShieldFS: A self-healing, ransomware-aware filesystem. In Proceedings of the 32nd Annual Conference on Computer Security Applications, ACM, New York, NY, USA, 5–8 December 2016; pp. 336–347.
31. Ahmadian, M.M.; Shahriari, H.R. 2entFOX: A framework for high survivable ransomwares detection. In Proceedings of the 13th International Iranian Society of Cryptology Conference on Information Security and Cryptology (ISCISC), Tehran, Iran, 7–8 September 2016; pp. 79–84.
32. Zimba, A. Malware-free Intrusion: A novel approach to Ransomware infection vectors. *Int. J. Comput. Sci. Inf. Secur.* **2017**, *15*, 317.
33. Shaukat, S.K.; Ribeiro, V.J. RansomWall: A layered defense system against cryptographic ransomware attacks using machine learning. In Proceedings of the 10th International Conference on Communication Systems & Networks (COMSNETS), Bengaluru, India, 3–7 January 2018; pp. 356–363.
34. Zuhair, H.; Selamat, A. RANDS: A machine learning-based anti-ransomware tool. In *Advancing Technology Industrialization through Intelligent Software Methodologies, Tools and Techniques, In Proceedings of the 18th International Conference on New Trends in Intelligent Software Methodologies, Tools and Techniques (SoMeT2019), Kuching, Sarawak, Malaysia, 23–25 September 2019*; IOS Press: Amsterdam, The Netherlands, 2019; pp. 573–587. [CrossRef]
35. Zuhair, H.; Selamat, A. An Intelligent and Real-Time Ransomware Detection Tool Using Machine Learning Algorithm. *J. Theor. Appl. Inf. Technol.* **2019**, *97*, 3448–3461.
36. Homayoun, S.; Dehghantanha, A.; Ahmadzadeh, M.; Hashemi, S.; Khayami, R.; Choo, K.K.R.; Newton, D.E. DRTHIS: Deep ransomware threat hunting and intelligence system at the fog layer. *Future Gener. Comput. Syst.* **2019**, *90*, 94–104. [CrossRef]
37. Krawczyk, B.; Minku, L.L.; Gama, J.; Stefanowski, J.; Woźniak, M. Ensemble learning for data stream analysis: A survey. *Inf. Fusion* **2017**, *37*, 132–156. [CrossRef]
38. Huang, G.; Huang, G.B.; Song, S.; You, K. Trends in extreme learning machines: A review. *Neural Netw.* **2015**, *61*, 32–48. [CrossRef]
39. Kwon, O.; Sim, J.M. Effects of data set features on the performances of classification algorithms. *Expert Syst. Appl.* **2013**, *40*, 1847–1857. [CrossRef]
40. Benign Software. Available online: http://software.informer.com/software/ (accessed on 4 April 2019).
41. *Virus Share*, "Malware Repository". Available online: https://virusshare.com (accessed on 13 January 2019).
42. Virus Total-Intelligence Search Engine, "Free Online Virus, Malware URL Scanner". Available online: https://www.virustotal.com (accessed on 21 August 2019).