Contents lists available at ScienceDirect

Computers & Security

journal homepage: www.elsevier.com/locate/cose

Zero-Ran Sniff: A zero-day ransomware early detection method based on zero-shot learning

Mingcan Cen[a], Xizhen Deng[b,c], Frank Jiang[a,*], Robin Doss[a]

[a] Centre for Cyber Resilience and Trust (CREST), Deakin University, 221 Burwood Highway, Geelong, 3125, Australia
[b] Zibo Big Data Center, Shandong, 255000, China
[c] Guangxi Normal University, Guilin, Guangxi, 541004, China

A R T I C L E I N F O

Keywords:
Ransomware
Ransomware early detection
Cybersecurity
Malware
Encrypted attacks

A B S T R A C T

Ransomware attacks, which blackmail victims into paying a ransom by locking their devices or encrypting their files, have become one of the major threats to network security. Conventional anti-ransomware tools often fail to detect zero-day ransomware attacks due to the inability to obtain zero-day ransomware signatures in advance to train detection models. In addition, zero-day ransomware attacks often use sophisticated encryption techniques to launch attacks on new vulnerabilities, and these encryption attacks cause irreversible damage to victims' digital files even if they choose to pay a ransom. Hence, it is imperative and urgent to detect unknown ransomware attacks at the earliest possible stage, ideally before the encryption phase. To this end, this paper proposes Zero-Ran Sniff (ZRS), an early zero-day ransomware detection method based on zero-shot learning, which can detect zero-day ransomware attacks in the early stage. ZRS leverages the portable executable header (PE header) feature from executable files to identify ransomware. It comprises two stages: an auto-encoding network-based core attribute learning (AE-CAL) stage and a self-attentive mechanism-based convolutional neural network inference Stage (SA-CNN-IS). During the AE-CAL stage, the core features of known and unknown classes of ransomware are extracted using self-encoding networks, and the SA-CNN-IS phase identifies ransomware. To the best of our knowledge, we are the first to explore the use of zero-shot learning for zero-day ransomware early detection. Experimental results demonstrate that the proposed ZRS outperforms traditional machine learning methods. Compared to previous zero-day detection work, ZRS achieves a recall of 98.47% and an accuracy of 96.31%

1. Introduction

Ransomware is a form of malware that continues to evolve, presenting a significant threat to cybersecurity (Cen et al., 2024). This type of malware works by encrypting the victim's data, restricting the user from logging onto the system or accessing files, and establishing links to command and control servers (C&C) to blackmail the victim to pay a ransom to recover the compromised data. While the origins of ransomware can be traced back to the AIDS Trojan virus (Ganta et al., 2020) that emerged in 1989, in recent years, ransomware attacks have become increasingly sophisticated, displaying polymorphism and metamorphic characteristics that make detection and scanning by traditional anti-malware tools challenging. To obtain more ransom, ransomware attacks have shifted from targeting individuals to critical infrastructures, including hospitals, government departments, banks, and large commercial companies (Wade, 2021). Attackers also used new untraceable technologies, such as anonymity, peer-to-peer (P2P) networks, and decentralized cryptocurrencies (e.g., Ethereum, Bitcoin), making it even more difficult for authorities to track them.

In the typical process of ransomware encryption aimed at compromising a victim's files and resources, several critical steps are involved (Cen et al., 2024; Moussaileb et al., 2021, 2018), as delineated in Fig. 1. The attack often commences with the user being deceived into interacting with a malicious email or downloading a hazardous payload. Upon successful infiltration into the system, the ransomware deploys necessary attack mechanisms. It then employs file-sharing protocols for lateral movement to propagate the infection and initiates a systematic search for files of significant value. In phase 5 of Fig. 1, the crypto-ransomware establishes communication with C&C and generates encryption keys locally or remotely. Following this is the destruction phase, where the ransomware encrypts the most important user files on hard drives, removable drives, and mapped network shares for extor-

* Corresponding author.
E-mail address: frank.jiang@deakin.edu.au (F. Jiang).

https://doi.org/10.1016/j.cose.2024.103849
Received 2 May 2023; Received in revised form 2 January 2024; Accepted 11 April 2024
Available online 17 April 2024

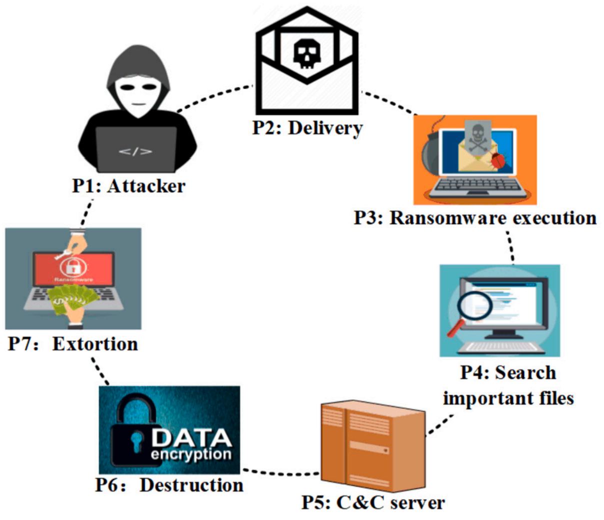


Fig. 1. Cryto-ransomware attack process.

tion (Ahmed et al., 2020). The final phase involves extortion, wherein the attacker demands a ransom within a specified timeframe to decrypt the victim's data. According to the phases listed above, identifying ransomware as early as possible is crucial for effectively stopping it from causing damage because ransomware encryption techniques can cause irreparable damage to the victim's data, regardless of whether the victim pays the ransom (Kok et al., 2022). Young and Yung have proved that if ransomware is identified on a particular system, reversing its impact on the host system may be computationally intractable (assuming strong asymmetric encryption) (Ahmadian et al., 2015; Young and Moti, 1996). Furthermore, typical users often lack the necessary expertise to recognize potentially malicious applications, and by the time a ransomware attack becomes apparent to them, it is usually too late for effective intervention. Hence, to effectively protect users' digital assets, ransomware attacks must be identified in the early phase, i.e. pre-encryption phase (McIntosh et al., 2022; Al-rimy et al., 2021). In this paper, pre-encryption refers to the point in time before the target files are encrypted, when a ransomware detection system is capable of accurately and reliably identifying ransomware threats.

Traditional ransomware detection tools typically rely on signature-based methods to identify ransomware by matching signatures extracted from suspicious samples with those of previously identified ransomware samples (Rezaei et al., 2021). These methods limit the number of files lost after the encryption phase by blocking any process with similar behavior, and characteristics to ransomware (API calls, registry keys, embedded strings in binaries, etc. Moussaileb et al., 2018). However, ransomware has developed countermeasures to evade antivirus tools. For instance, it often employs techniques like encrypting its true signature to evade system detection. Moreover, ransomware exhibits a plethora of variants, rendering signature-based detection methods ineffective in identifying novel or previously unseen ransomware, commonly referred to as zero-day ransomware attacks (Masdari and Khezri, 2020). As a result, many researchers (Feng et al., 2017; Deng et al., 2023; Chen et al., 2021; Guo et al., 2020; Zhu et al., 2022) have turned to machine learning (ML) and deep learning (DL) techniques, utilizing static or dynamic features, to detect ransomware. Machine learning-based techniques have a high potential for generalization and can detect unintuitive patterns in previously unseen ransomware samples, thus detecting those with new behavior similar to previous ransomware samples. Advances in deep neural networks have demonstrated their powerful representational learning capabilities, enabling more efficient, higher-level data representations through layered training from the lowest to the highest levels (Rezaei et al., 2021).

However, it is important to note that dynamic feature-based detection methods typically require files to be executed in an isolated environment to extract features. This not only demands substantial time and computational resources but also increases the risk of ransomware leakage, consequently hindering rapid ransomware detection. Therefore, developing a swift and effective method for the rapid detection of zero-day ransomware is a pressing challenge in the field of ransomware detection. Against this backdrop, this work proposes a static feature-based zero-shot learning method to achieve early detection of zero-day ransomware. It aims to address the aforementioned challenges and detect zero-day ransomware attacks in the pre-encryption phase by analyzing PE header static features extracted from ransomware samples. The proposed method comprises three parts: pre-processing, attribute learning, and inference. The pre-processing phase obtains the raw PE header feature vector of the ransomware, including MS-DOS, COFF, and Optional header parts. In the attribute learning phase, a self-encoding network extracts the core attribute features of the PE header, which is a convolutional neural network classification model with a self-attention mechanism process for ransomware identification in the inference phase. The proposed method addresses the zero-shot paradigm problem, enabling the inference of unseen ransomware samples and thereby contributing to the resolution of the growing problem of zero-day ransomware attacks.

In summary, the contribution of this paper is summarized as follows.

- A ZRS method for the early detection of zero-day ransomware was proposed. This approach diverges from traditional methods that depend on external expert knowledge for attribute learning, which often results in the inability to handle zero-day exploit attacks. AE-CAL is utilized to learn low-dimensional core attribute features from high-dimensional raw features in an unsupervised form. These extracted core attribute features are then employed to learn zero-day ransomware features for ransomware early detection.
- According to the correlation between input features, in the inference stage, this paper proposes a convolutional neural network classification model based on the self-attentive mechanism (SA-CNN-IS). This model can prompt the network to focus on more significant features with high weights and ignore irrelevant features or noise with low weights. Compared with traditional ML methods, empirical results demonstrate that SA-CNN-IS exhibits better detection performance.
- Current solutions to the zero-day attack problem typically rely on dynamic analysis or complex network approaches, which can be time-consuming and struggle to meet the requirements of early detection. To the best of our knowledge, this study is the first to explore the use of zero-shot learning for addressing the zero-day ransomware early detection problem.

The remainder of this paper is organized as follows: Section 2 presents related work and background. Section 3 introduces the proposed attribute learning and inference stages. Section 4 provides details on comparative experiments, ablation experiments, and experiments conducted in this paper. Finally, the conclusions and future work are presented.

2. Related works

Typically, ransomware detection methods include static analysis, dynamic analysis, and hybrid analysis. Static analysis extracts the static features of a ransomware attack before it occurs, enabling early detection without the need to execute the ransomware. Dynamic analysis, on the other hand, entails the extraction of features during a ransomware attack to identify anomalies while the code is running. Dynamic analysis generally provides better detection performance than static analysis but requires features to be extracted during the execution of the ransomware, which requires longer run times and does not guarantee the security of running ransomware in a virtual environment such as the Cuckoo sandbox (Oktavianto and Muhardianto, 2013). Hybrid analysis, which combines static and dynamic analysis, can enhance the accuracy and efficiency of ransomware detection. However, the resulting detection models may be overly complex, and detection times can be protracted. For zero-day exploit attacks, zero-day ransomware detection techniques can be employed, which can be based on either static or dynamic analysis. Hence, this section will present the latest research on ransomware detection using the methods described above.

2.1. *Dynamic analysis*

Many scholars have proposed various ransomware detection methods based on dynamic analysis. Feng et al. (2017) proposed a detection method that determines if a decoy file is encrypted to detect ransomware. However, this method does not have specific rules for different types of ransomware in terms of encryption order, making it challenging to apply to practical applications.

Vinayakumar et al. (2017) collected 131 APIs as features by running several samples of ransomware families in a virtual environment and used a multi-layer perceptron algorithm to detect the extracted features. Although the method has high accuracy, it requires a long runtime to extract API calls from day files, which makes early detection difficult.

Chen et al. (2017) proposed a real-time ransomware detection method for Android mobile devices called RansomProber. This method uses dynamic analysis to analyze user interface widgets and user finger movements on the touchscreen. However, before RansomProber successfully detects anomalous actions caused by ransomware, users may have already lost some important and sensitive files.

Chen et al. (2021) proposed an early detection method for ransomware based on API short sequences. They extracted short sequences of ransomware application programming interfaces (APIs) and used N-grams and TF-IDF algorithms to compute feature vectors from the collected API short sequences. Finally, a random forest algorithm was used for ransomware detection. However, their proposed early detection method still requires the software to run for 7.2 seconds before the ransomware is detected, during which the ransomware may have already been attacked. Kharaz et al. (2016) have proposed a dynamic ransomware analysis system. Their proposed method detects ransomware by monitoring the file system activity and the entropy transformation of I/O data buffers. Ramesh and Menen (2020) proposes a new ransomware detection technique that identifies ransomware attacks by evaluating the current state of a computer system.

2.2. *Static analysis*

The above research shows that dynamic analysis usually requires the extraction of features for detection during software operation, which is often time-consuming and difficult to meet the requirements of early detection, and there is a risk of ransomware leakage from the virtual environment during operation. Static analysis is a common analysis method for early detection of ransomware. Common static features include opcodes, entropy features, PE header, and binary raw images. Guo et al. (2020) proposed a visualization-based ransomware detection method. In their proposed method, the binary files of ransomware and benign software are converted into grayscale images, then the image features are extracted using a migration learning VGG 16 neural network, and finally, an SVM machine learning classification model is used for classification. Zhang et al. (2019) used disassembly techniques to extract the N-gram opcode sequences of each software, then used the TF-IDF algorithm to calculate the importance of the extracted features, and finally used the random forest algorithm to detect the filtered features. Zhu et al. (2022) proposed an entropy graph-based learning method with few samples for ransomware classification. In the proposed method, raw binary files are converted into entropy images and then the similarities between ransomware image pairs are analyzed using twin neural networks for multi-classification. Baldwin and Dehghantanha (2018) used nine feature selection algorithms to filter the extracted opcodes. The filtered features were then used for ransomware detection using a support vector machine (SVM) machine learning algorithm.

In addition to the above-mentioned opcodes, entropy features, and raw binary images being used for ransomware detection, the PE header contains key structural information that can be used to identify benign ransomware and its extraction process is easier and more convenient compared to other static features, therefore there are also many research works using PE header features for ransomware detection.

Moreira et al. (2023) apply static analysis to detect ransomware by converting PE header files into color images in a sequential vector pattern and classifying these via the Xception CNN model without transfer learning. Rezaei et al. (2021) proposed a ransomware approach using raw bytes of PE headers. Their proposed method uses a deep embedding technique to process PE headers obtained from ransomware and then uses the KNN clustering algorithm for malware detection. Manavi and Hamzeh (2022) proposed a ransomware detection method using a graph embedding technique by using PE header features. Vidyarthi et al. (2019) proposed a ransomware detection method based on PE headers of executable files. Feature values in the header field of each program's executable were extracted. Subsequently, these features were employed to train machine learning models, including algorithms like J48, Random Forest, and Naive Bayes.

2.3. *Hybrid analysis*

In addition to the static and dynamic analysis methods mentioned above, some researchers use hybrid analysis for ransomware detection. Hybrid analysis methods take a lot of time for feature extraction and are therefore difficult to meet the requirements of early detection (Ferrante et al., 2017; Netto et al., 2018; Ashraf et al., 2019; O'Shaughnessy and Sheridan, 2022). Ashraf et al. (2019) proposed a hybrid analysis-based ransomware detection method. They extract different parts of PE headers as static features. Then, each program is applied in the Cuckoo environment, and registry entries and API calls are extracted as dynamic features during the runtime. Finally, the ResNet-18 network is used for classification.

2.4. *Zero-day vulnerability detection technology*

None of the research work mentioned above is effective in zero-day ransomware detection. Zero-day ransomware detection means that the class of ransomware samples to be detected in the test set never appears in the training set. In response to the growing number of zero-day exploit attacks, some research works use the zero-shot learning method (Romera-Paredes and Torr, 2015) to perform ransomware detection.

Zahoora et al. (2022) propose a zero-shot learning framework for zero-day ransomware detection using a combination of deep shrinking self-encoder networks and integrated learning. Their proposed method learns the core features of the original features through a self-encoder network and then uses voting-based integrated learning for inference. However, it requires obtaining seven dynamic features by executing the ransomware in a virtual environment, such as Cuckoo Sandbox, which can be time-consuming and is not suitable for early detection. To effectively detect malware including zero-day attacks, Kim et al. (2018) proposed a model for detecting unknown malware, called transitive deep-convolutional generative adversarial network (tDCGAN). In their proposed approach, the authors use a generative adversarial network (GAN) to generate fake malware and distinguish it from real malware. Zero-day malware is modeled as known software with additional noise and distinguished using a trained detector. Further to Kim et al.'s work, a malware training framework called PlausMal-GAN is proposed by Won et al. (2022). PlausMal-GAN can be used to generate malware from The PlausMal-GAN can generate zero-day malware-like images with high quality and diversity from existing malware data appropriately, and can show stable and efficient performance in zero-day

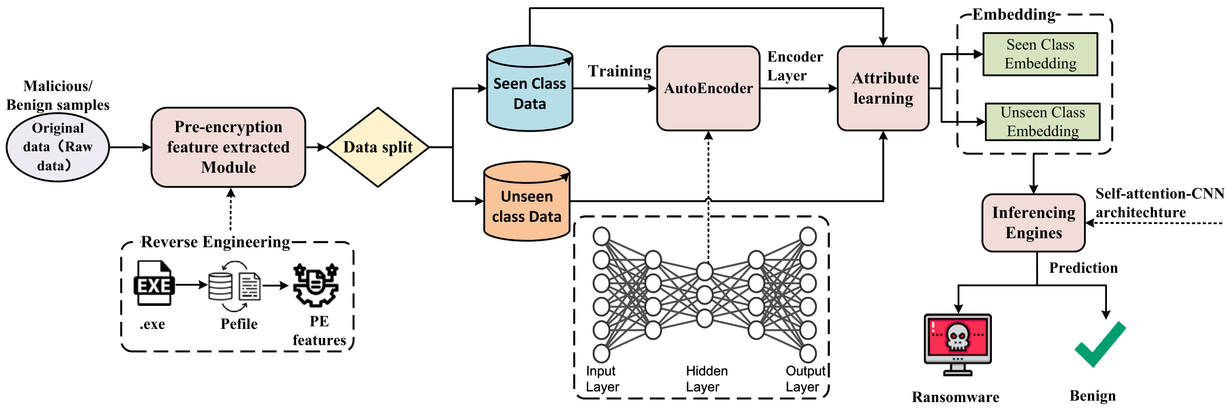


Fig. 2. The framework of the proposed Zero-Ran Sniff (ZRS).

Table 1
Key notations used in the proposed method.

Notation	Description
X	Feature space, which is D-dimensional
$\mathcal{T}$	Semantic space, which is M-dimensional
$\mathcal{S}, \mathcal{U}$	Set of seen classes and set of unseen classes, respectively
N_{tr}, N_{te}	Number of training instances and number of testing instances, respectively
N_s, N_u	Number of seen classes and number of unseen classes, respectively
D^{tr}	The set of labeled training data from seen classes
X^{te}	The set of testing instances from unseen classes
Y^{te}	Labels for testing instances
$\left(x_i^{tr}, y_i^{tr}\right)$	The ith labeled training instance: features $x_i^{tr} \in \mathcal{X}$ and label $y_i^{tr} \in \mathcal{S}$
x_i^{te}	The ith unlabeled testing instance: features $x_i^{te} \in \mathcal{X}$
T^s, T^u	The set of prototypes for seen classes and unseen classes, respectively
$\left(c_i^s, t_i^s\right)$	The ith seen class$c_i^s \in \mathcal{S}$ and its class prototype $t_i^s \in \mathcal{T}$
$\left(c_i^u, t_i^u\right)$	The ith unseen class$c_i^u \in \mathcal{U}$ and its class prototype $t_i^u \in \mathcal{T}$
$\pi(\cdot)$	A class prototyping function $\pi(\cdot) : \mathcal{S} \cup \mathcal{U} \rightarrow \mathcal{T}$
$f^u(\cdot)$	A zero-shot classifier $f^u(\cdot) : \mathcal{X} \rightarrow \mathcal{U}$

malware. Similarly, Barros et al. (2022) proposed a zero-shot learning strategy for detecting zero-day exploits named Malware-SMELL, which develops a new representation space to compute the similarity between malware image pairs and enhances the separability between classes, resulting in better results in zero-day ransomware detection.

The above proposed zero-day ransomware detection methods are all implemented relying on dynamic features or malware images, without considering the importance of early detection in zero-day ransomware detection. Therefore, this paper proposes a zero-day ransomware early detection method based on PE header static features combined with zero-shot learning. In contrast to alternative methods that require different amounts of preprocessing, time consumption, and resources to extract features, the proposed method can not only implement zero-day ransomware but also only use PE header features. Consequently, this results in significantly faster feature extraction, aligning with the imperative for early detection. The proposed detection method will be described in detail in the following sections.

3. The methodology

This section presents the details of the proposed method for zero-day ransomware detection. Fig. 2 depicts the proposed ransomware early detection architecture and its workflow, while Table 1 summarizes the primary notations used in this work. The architecture consists of three main parts: pre-processing, core attribute learning, and inference phase. The functions of each part are briefly described below.

Pre-processing: In the initial phase, the utilized dataset is described and the corresponding raw ransomware binaries are obtained from reputable sources such as Virustotal (VirusTotal, 2024) and VirusShare (VirusShare, 2024) using the provided hashes. These raw binaries are then statically analyzed using the pefile library in Python (Python, 2023) to extract essential PE header features for subsequent detection tasks.

Core attribute learning: To solve the zero-day ransomware detection problem, core attributes need to be extracted from seen classes and correlated between seen and unseen classes to identify ransomware in unseen classes. To this end, the self-coding technique is adopted to encode the original features and extract deep core semantic information for zero-day ransomware detection, based on the theory related to zero-shot learning.

Inference phase: In this phase, based on the correlation between PE header features and the advantages of self-attentive mechanism in feature extraction, this paper proposes a convolutional neural network inference engine combined with self-attentive mechanism (SA-CNN-IS) to better locate and extract key features for zero-day ransomware detection.

3.1. Pre-processing

3.1.1. Dataset

The proposed method was evaluated using the dataset provided by Sgandurra et al. (2016). This dataset includes 582 ransomware and 942 benign software, which is a highly unbalanced dataset. The ransomware binaries are obtained from the VirusShare (VirusShare, 2024) website and are presented in diverse executable file formats, such as .exe, .pdf, .zip, etc. These ransomware binaries can be further categorized into 11 different families, with their distribution illustrated in Table 2.

Table 3 shows the SHA-256 hashes of some of the ransomware in the dataset along with the corresponding ransomware family and time of appearance. The benign software used in this study was collected from reliable sources such as popular web browsers, file readers, games, and other commonly used applications on personal computers.

Table 2
Distribution of the different ransomware families.

ID	Family	Numbers	ID	Family	Numbers
0.	Citroni	50	6.	Matsnu	59
1.	CryptLocker	107	7.	Pgpcoder	4
2.	CryptoWall	46	8.	Reveton	90
3.	Kollah	25	9.	TeslaCrypt	6
4.	Kovter	64	10.	Trojan-Ransom	34
5.	Locker	97	11.	benign	942

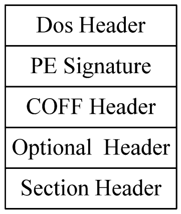


Fig. 3. The PE file format.

3.1.2. Dataset split

Zero-day ransomware refers to new variants of ransomware that have been developed in response to new vulnerabilities and are often very damaging and stealthy. Because they are developed in response to new vulnerabilities, traditional methods of using database signatures do not find the corresponding signatures and are often difficult to detect. Zero-shot learning has emerged as a promising technique for detecting zero-day ransomware. In zero-shot learning, the classes in the training set and the classes in the test set are different from each other. The ransomware instances in the training set can be considered as seen classes to simulate real-life ransomware that has already been discovered. In contrast, the classes in the test set can be considered as unseen classes to simulate newly developed ransomware variants, i.e. zero-day ransomware. Zero-shot learning can be divided into two parts: attribute learning and inference phase. The attribute learning process aims to connect the seen and unseen classes by extracting knowledge from labeled training samples or learning intermediate information. The goal is to learn a semantic attribute space that can establish the relationship between seen and unseen classes. It can be subdivided into Engineered Semantic Spaces and Learned Semantic Spaces (Wang et al., 2019). In Engineered Semantic Spaces, each dimension of the semantic space is designed by humans. In the learning semantic space, the dimensions in the space are not designed by humans but are obtained from the output of some machine learning models. In the output of these machine learning models, each dimension does not have an explicit semantic meaning. In the inference phase, the learned derived knowledge, or core attributes, are used to detect zero-day ransomware. Depending on the information used in the model learning process, zero-shot learning can be divided into inductive zero-shot learning and direct inference zero-shot learning. Inductive zero-shot learning uses only seen class features to learn the core attributes, while inductive zero-shot learning uses both seen and unseen class features to learn the core attributes. Depending on the test set used, zero-shot learning can be divided into two types, traditional zero-shot learning (CZSL), which uses unseen classes as the test set and evaluates model goodness on them, and generalized zero-shot learning (GZSL) (Sun et al., 2021), which uses both seen classes and unseen classes as the test set. The Zero-Ran sniff detection framework proposed in this paper is an inductive-based approach that uses only data from the seen class for model training and uses CZSL conventional zero-shot learning to evaluate the model.

Thus, according to the above description, the original ransomware dataset D is divided into two disjoint subsets of the seen and unseen classes, Seen-train-dataset$=\left\{X_i^{tr}, Y_i^{tr}\right\}$ and Unseen-test-dataset$=$ $\left\{X_i^{tsr} X_i^{ts}, Y_i^{ts}\right\}$, where X is the original feature, Y is its corresponding label and $Y_i^{tr} \cap Y_i^{ts} = \varnothing$. The ransomware categories for the seen and unseen classes are shown below, where G_i represents benign software, while datasets Y^{tr} and Y^{ts} are dedicated to model training and evaluation, respectively.

$$Y^{tr} = \left\{G_i^k,\ \text{Critroni, CryptLocker, CryptoWall, Kollah, Kovter, Locker, Matsnu}\right\} \tag{1}$$

$$Y^{ts} = \left\{G_{k+1}^n,\ \text{Pgpcoder, Reveton, TeslaCrypt, Trojan - Ransom}\right\} \tag{2}$$

3.1.3. Feature selection

The majority of individuals and organizations currently use Windows as their operating system because of its fluidity. As a result, ransomware creators have developed many ransomware programs for this operating system. The programs are usually installed in the standard binary format of .exe and .dell. Following Microsoft's comprehensive documentation (Microsoft, 2024), an overview of the PE file format is shown in Fig. 3. The header of a PE file is composed of five key components, namely the Dos header, PE signature, COFF header, Optional header, and Section header. Each PE file starts with an MS-Dos header that determines whether the executable runs on a Windows operating system and can output at least one message indicating that it needs to run on a Windows platform.

The PE Signature contains 4 bytes, typically with the value "PE\0\0", and is used to specify that the file is a PE file. The COFF header contains general information about the physical distribution of the PE file such as the Machine field, Time Date Stamp, Number Of Sections, Number Of Symbols, Machine, Size Of Code, and Pointer To Symbol Table, which can be used to identify ransomware. After the COFF header comes the Optional header, which contains information about the logical distribution of the PE file and the data directory. The operating system uses the PE file logical distribution information to load and execute the file. The data directory contains the location of the different tables required by the operating system such as import tables, relocation tables, and sizes, which are loaded into memory during runtime. Lastly, the section header contains the location of the program code and data. For example, the ".bss" section is used for data initialization, the ".text" section for storing code, and the ".data" section for storing static and global variables. Matt Pietrek calls it the "phone book containing information about each section in the image" (Pietrek, 1994). Since zero-day ransomware may use unknown or uncommonly imported dynamic-link libraries (DLLs) for attacks, analyzing the import table of the PE file can help identify import behaviors different from known ransomware, indicating potential zero-day attacks. Additionally, zero-day attacks may modify code or add hidden code segments. By examining the PE file's section table and code segments, inconsistencies with normal programs can be discovered. Zero-day attacks may also alter file attributes, such as file size and timestamps, to evade detection. Checking for abnormal changes in these file attributes can help detect potential zero-day attacks.

According to document (Microsoft, 2024) and the work of Kim (2018), this study utilizes the pefile library in Python to extract features from five different structures of executable file samples. Each sample, regardless of whether it belongs to the category of ransomware or benign software, undergoes the extraction of 87 features (as depicted in Table 4). A total of 50,634 PE original features from ransomware samples and 81,954 features from benign software samples have been amassed. These PE features encompass critical aspects, including physical distribution, logical distribution, and code characteristics, which effectively enable the differentiation of ransomware from normal software. As previously detailed in Formulas (1) and (2), the training dataset encompasses seven distinct categories of ransomware, while the testing dataset comprises four varying ransomware categories. The sample count ratio between the training and testing datasets is set at 7:3.

Table 3
Ransomware samples: SHA-256 hash values and the Last analysis dates at VirusTotal.

Family	SHA-256 hash	Date
Critroni variant #1	0f9d3cfa0af4bc4c79ae1df6d243df77386a2ca9be0f47fba266fd6a7d9ad73a	2022-02
Critroni variant #2	1a5984c35777add7ef677cf4a47b89507ff8af16065ee62684e3b11b3bc22057	2022-01
Cryptolocker variant #1	0a0ceed71e45ea8692b1584e89fc2173690436e42f9f8d0482e4c54dba900d33	2018-10
Cryptowall variant #1	0e8ad5c809bc8c67b7cd0aaf59fcd9e684d0e64bbc6d01b4fb40c86e6c695db3	2022-03
Locker variant #1	0ca8a831ab8b9c2b46b1998b659ee7bc15c20fa349db1885dc927237c175eba0	2018-10
Matsnu variant #1	00d786974e609adf93b29e6e86f7439074149e6cceabf36bdb56f708d47aee0e	2018-10
Kovter variant #1	00bf847c9a53922a2b36348456ee0f1afff0eec705f22162b9662a77f1440cd6	2018-10
Trojan-ransomware#1	2a48e06e0641ce0ffdcfa1f5633ce77d4669dab87c311e70cda5930f2d809456	2021-10

Table 4
87 PE header features extracted using pefile library.

Features	Features	Features
e_magic	e_res2_2	Signature
e_cblp	e_res2_3	Machine
e_cp	e_res2_4	NumberOfSections
e_crlc	e_res2_5	TimeDateStamp
e_cparhdr	e_res2_6	PointerToSymbolTable
e_minalloc	e_res2_7	NumberOfSymbols
e_maxalloc	e_res2_8	SizeOfOptionalHeader
e_ss	e_res2_9	Characteristics
e_sp	e_res2_10	Magic
e_csum	e_res2_11	MajorLinkerVersion
e_ip	e_res2_12	MinorLinkerVersion
e_cs	e_res2_13	SizeOfCode
e_lfarlc	e_res2_14	SizeOfInitializedData
e_ovno	e_res2_15	SizeOfUninitializedData
e_res	e_res2_16	AddressOfEntryPoint
e_oemid	e_res2_17	BaseOfCode
e_oeminfo	e_res2_18	ImageBase
e_res2_1	e_res2_19	SectionAlignment
e_lfanew	e_res2_20	FileAlignment
Misc	Name	MajorOperatingSystemVersion
Misc_PhsicalAddress	VirtualAddress	MinorOperatingSystemVersion
Misc_VirtualSize	SizeOfRawData	MajorImageVersion
SizeOfStackReserve	PointerToRawData	MinorImageVersion
SizeOfStackCommit	PointerToRelocations	MajorSubsystemVersion
SizeOfHeapReserve	PointerToLinenumbers	MinorSubsystemVersion
SizeOfHeapCommit	NumberOfRelocations	Win32VersionValue
LoaderFlags	NumberOfLinenumbers	SizeOfImage
NumberOfRvaAndSizes	Characteristics	SizeOfHeaders
Subsystem	DllCharacteristics	CheckSum

3.1.4. Autoencoder-core attribute learning

Since no ransomware samples of the unseen classes are used in the training process, to solve the zero-day ransomware detection problem, core attributes need to be extracted from the seen classes to help identify zero-day ransomware in the unseen classes. These core attributes should contain relevant information about all unseen classes and be associated with samples in the feature space to ensure their availability. The core attribute information involved in existing zero-shot learning methods is usually some semantic information. It forms a semantic space that describes both seen and unseen classes. The semantic space contains semantic information about the unseen classes and is a very important component of zero-shot learning.

There are two main ways to construct semantic spaces, one is engineering-based semantic space and the other is learning-based semantic space. Due to the complexity of ransomware attacks, which are difficult to represent in a concrete description and the process of constructing an engineering semantic space often requires a lot of time from domain experts, this paper uses a learning-based approach to construct the semantic space. In the learning semantic space, the core attributes in the space are not designed manually but are implicit representations of the original features (e.g. the output of the hidden layer) by machine learning or deep learning methods. An Autoencoder-based Core Attribute Learning network (AE-CAL) is proposed, utilizing seen samples as the training set for an autoencoder (Baldi, 2012) network. The extracted deep semantic information is used as core features for subsequent detection. In the training process, unseen samples are used as test samples applied to the trained autoencoder network, constantly monitoring changes in the loss function. If the loss function converges and the core features can reconstruct the original features, it indicates that the autoencoder network trained with seen samples is also effective for unseen samples. The self-encoder can learn useful potential representations, i.e. core attributes, by using the original attributes of the input as learning targets without labeling information.

Table 5
Number of neurons in encoding and decoding layers of Autoencoder.

Neurons	E1	E2	D1	D2
Input Neurons	87	35	20	35
Output Neurons	35	20	35	87
Activation Function	Relu	Relu	Relu	-

To build the AE-CAL model, a self-encoder with four fully connected layers was trained on the Seen-train-dataset. The first two layers are encoding layers, and the remaining two layers are decoding layers. After training the self-encoder, the trained coding layers are used on the Seen-train-dataset and Unseen-test-dataset to extract 20 core features. These are used as the training set and test set for subsequent model training and model testing, respectively. Fig. 4 shows the proposed AE-CAL network, and Table 5 provides the details of the self-encoder network design.

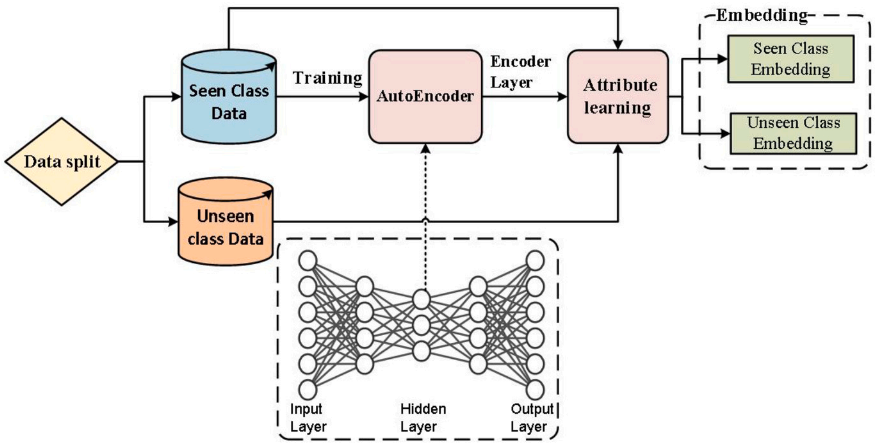


Fig. 4. Autoencoder-core attribute learning network.

3.1.5. *Autocoder*

Autoencoder is a type of artificial neural network commonly used in semi-supervised and unsupervised learning. It consists of an encoder that employs a back-propagation algorithm with an optimization method (e.g., gradient descent). The autoencoder uses the original input attribute x itself as supervision to guide the neural network to learn a potential representation for reconstructing the input data. The objective function for the autoencoder is shown below.

$$L = \|x - x'\|^2 \tag{3}$$

The role of the coding layer is to convert the original input high-dimensional raw data into a core attribute x' with lower dimensionality through the learning of the neural network, the coding process from the input layer to the hidden layer is shown as follows.

$$Encoding = x' = \partial\left(W_e x + b_e\right) \tag{4}$$

In the decoding layer, the role of the decoding layer decodes each encoding layer. The output of the decoder should be able to restore the original features x from the core attributes x' to the maximum extent. The decoding process from the hidden layer to the output layer can be expressed by the mathematical formula as

$$Decoding = x = \partial\left(W_d x' + b_d\right) \tag{5}$$

where W_e and W_d denote the corresponding neural network weights for encoding and decoding respectively, b_e and b_d are the corresponding biases, and ∂ is the activation function.

3.1.6. *Data visualization distribution*

In this section, the t-distributed stochastic neighbor embedding (t-SNE) algorithm (Van der Maaten and Hinton, 2008) is used to visualize the data distribution of ransomware and benign software. The t-SNE algorithm is capable of converting high-dimensional data into two-dimensional or three-dimensional data, which is useful for visualizing the clusters that occur in ransomware and benign software at different scales. Fig. 5 shows the distribution of the raw data using the t-SNE algorithm. Fig. 6 displays the distribution of the data using the t-SNE algorithm for extracting core features. It can be observed from the figures that the two categories overlap at some points, indicating some similarity in their data distribution. However, the differences between ransomware and benign software lead to significant differences in their overall distribution. The data distribution of the extracted core features shows that the core attribute features retain the relevant characteristics present in the original data, and their distribution behaves similarly to the original data.

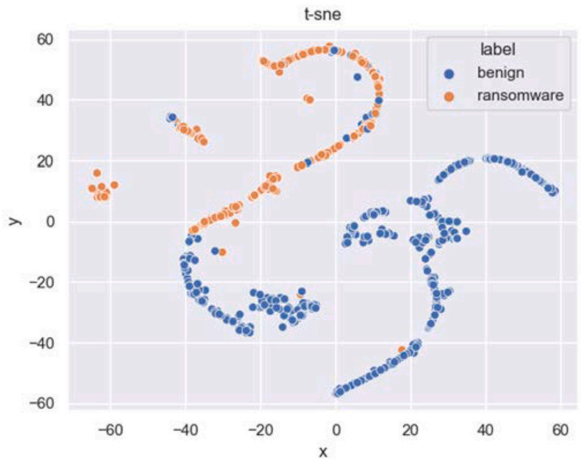


Fig. 5. Original features.

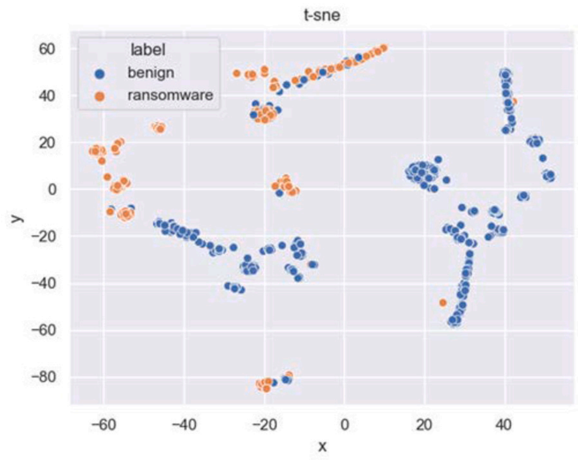


Fig. 6. Core attribute features.

3.2. *Inference stage (SA-CNN-IS)*

In the previous section, core attribute features of the PE header were extracted using the pefile and an AE-CAL network. In this subsection, the extracted core features will be employed for early detection of ransomware. Several well-known inference methods have been proposed in previous zero-shot learning work (Romera-Paredes and Torr, 2015), such as probabilistic frameworks (Lampert et al., 2013), energy function

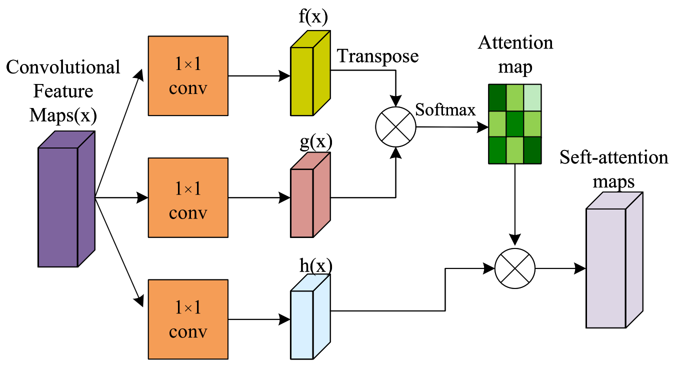


Fig. 7. Self-attention module adopted in SA-CNN-IS.

(Akata et al., 2013), KNN, integrated learning inference methods based on random forests, support vector machines, and bayesian regression (Zahoora et al., 2022). In this paper, a one-dimensional convolutional neural network inference method based on a self-attentive mechanism (SA-CNN-IS) is proposed. Its detection process can be divided into the following steps.

- Extract core features from the Seen-train dataset using the coding layer of the self-encoding network, and use the extracted core features to train a self-attentive-based ransomware early detection framework.
- Extract core features from the Unseen-train dataset using the coding layer of the self-encoding network, and use the extracted core features in the trained SA-CNN-IS detection model to measure the performance of the model.

With the introduction of the transformer model (Vaswani et al., 2017), the self-attention mechanism at its core has been widely used in natural language processing (Wu et al., 2018), statistical learning (Xie et al., 2020), target detection (Perreault et al., 2020), speech recognition (Yeh et al., 2019), and other fields. In the fields of malware analysis (Yakura et al., 2018; Athiwaratkun and Stokes, 2017) and ransomware analysis (Zhang et al., 2020), the self-attentiveness mechanism has also been introduced into proposed frameworks to improve the detection or classification performance of models. The attention mechanism is usually used after the output of a convolutional neural network (CNN) or recurrent neural network (RNN). In this paper, the self-attentiveness mechanism calculates the correlation between the PE header of the input features, allowing the network to focus on more significant features with high weights and ignore irrelevant features or noise with low weights. This prompts the network to concentrate on the relevant parts that have a significant impact on the detection results. The self-attention module adopted in the proposed SA-CNN-IS is shown in Fig. 7. The detailed configuration of SA-CNN-IS is shown in Table 6.

The self-attentive function can be described as mapping a query (represented by matrix Q) and a set of key-value pairs (represented by matrices K and V, respectively) to the output. The attention function is used to calculate the alignment fraction between the elements of K and V, which can be expressed by the following equation

$$\text{Attention}\ (Q,K,V) = \text{softmax}\left(s\left(QK^{\top}\right)\right)V \tag{6}$$

To calculate attention, the extracted core features are fed into a 1D convolutional neural network, and then the hidden layer features of the 1D convolutional neural network are transformed into two separate feature spaces, and attention is calculated using the following formula:

$$\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{W}_f\boldsymbol{x},\ \text{where}\ W_f \in \mathrm{R}^{\hat{C}\times C} \tag{7}$$

$$\boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{W}_g\boldsymbol{x},\ \text{where}\ \boldsymbol{W}_g \in \mathrm{R}^{\hat{C}\times C} \tag{8}$$

The attention is given by:

$$Att_{j,i} = \frac{\exp\left(s_{ij}\right)}{\sum_{i=1}^{N}\exp\left(s_{ij}\right)},\ \text{where}\ s_{ij} = \frac{\boldsymbol{f}\left(\boldsymbol{x}_i\right)^{\top}\boldsymbol{g}\left(\boldsymbol{x}_j\right)}{\sqrt{\hat{C}}} \tag{9}$$

4. Experiment results

4.1. Evaluation metrics

Several experiments were conducted to evaluate the effectiveness of the proposed early zero-day ransomware detection framework, utilizing evaluation metrics such as Accuracy (A), Precision (P), Recall (F1-score), ROC-AUC curve, and PR curve. The formulas for evaluation metrics are shown below.

$$\text{Accuracy} = \frac{\text{TP}+\text{TN}}{\text{TP}+\text{FP}+\text{TN}+\text{FN}} \tag{10}$$

$$F1-\text{score} = 2*\frac{\text{Precision}*\text{Recall}}{\text{Precision}+\text{Recall}} \tag{11}$$

4.2. Experimental environment and hyperparameter settings

4.2.1. Experimental setup

The experiments were conducted on a Windows 10 system equipped with an Intel(R) Core(TM) i7-10700KF CPU (3.8 GHz) and 16 GB RAM. The proposed detection framework was developed using the PyTorch library in Python 3.7. Additionally, Python's pefile library was utilized for feature extraction, while baseline and other machine learning methods, such as SVM, KNN, and RF, were implemented using scikit-learn.

4.2.2. Training setup

In the proposed detection architecture, an autoencoder is trained during the attribute learning phase to extract core attribute features from the original PE header features of ransomware. Throughout the training process of the autoencoder, the Adam optimizer is employed to optimize the small batch mean squared loss function, and ReLU is used as the activation function. To perform effective early zero-day ransomware detection with the extracted core attribute features, this paper constructs an inference engine based on the self-attentive mechanism and convolutional neural network. In the training process, the Adam optimizer is also used to optimize the loss function, and the cross-entropy loss function is used to optimize the neural network parameters. The maximum pooling layer is added to simplify the network complexity, reduce computational complexity, and prevent overfitting. Table 7 describes the relevant training parameter settings in the attribute learning and inference phases.

4.3. Comparison between original features and extracted core features

This subsection is dedicated to verifying that the core attribute features extracted by the self-encoding network can effectively represent the relevant features of the original data, thereby facilitating the association of features between seen and unseen classes for detecting zero-day ransomware attacks. To this end, we conduct comparative experiments between the original features and the core attribute features and present the comparison results of the detection performance based on the SA-CNN-IS inference method as shown in Fig. 8.

From the Fig. 8, it can be seen that compared to the detection model trained with the original features, the detection model trained with core attribute features shows significant improvement in accuracy (96.02%), precision (91.49%), recall (98.47%), and F1 score (96.31%). This indicates that an increase in feature dimensionality does not necessarily lead to an improvement in detection performance. On the contrary, high-dimensional redundant features increase the detection difficulty. Furthermore, experimental results demonstrate that the core attribute features extracted based on the self-encoding network are fully capable of replacing the original features for zero-day ransomware detection. In comparison to using the original features, the core features improve the

Table 6
Configuration of SA-CNN.

Layer	Parameters (Filter size, Stride)	Output (Batch_size,Dimension)
Input	-	16 × 87
Conv	16,1	16 × 80
Relu	-	16 × 80
Max-pool	2,2	16 × 40
Conv	16,1	16 × 40
Relu	-	16 × 40
Max-pool	2,2	16 × 20
Conv	16,1	16 × 20
Relu	-	16 × 20
Max-pool	2,2	16 × 8
Self-Attention block	-	16 × 8
Fully connected	-	16 × 2

Table 7
Training parameter settings.

Name	Attribute learning	Inference stage
Activation function	Relu	Relu
Loss function	MSE Loss	CrossEntropy Loss
Learning rate	0.001	0.01
Batch_size	64	16
Training epochs	50	300

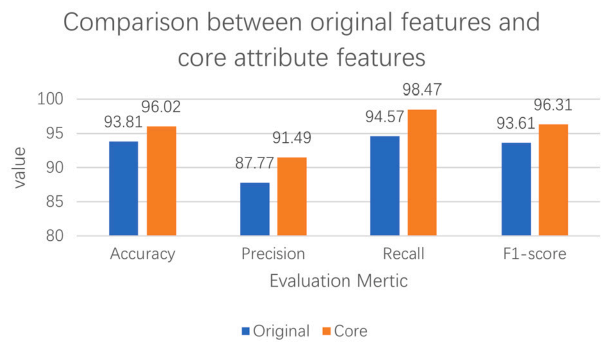


Fig. 8. Comparison between original features and extracted core features.

detection of zero-day ransomware on the one hand, and on the other hand, due to their lower data dimension, their training and detection time is shorter, which is more in line with the requirements of early and fast detection.

4.4. Comparison with other machine learning methods

The efficacy of the proposed self-attention mechanism-based inference engine was evaluated by conducting a comparative analysis with commonly used baseline machine learning methods. Experimental results using core features for traditional zero-shot learning with various machine learning algorithms are presented in Table 8. These algorithms include Support Vector Machines (SVM), Decision Trees (DT), Random Forests (RF), K-nearest Neighbor (KNN), and the Adaptive Augmentation Algorithm (Adaboost). It can be seen that the SVM and DT classifiers exhibit the poorest performance in detecting zero-day ransomware. Conversely, RF, KNN, and Adaboost exhibit more similar performances. The proposed SA-CNN-IS model, which is based on the self-attentive mechanism, outperforms all other baseline methods in all four evaluation metrics. In the context of zero-day ransomware detection, detecting ransomware is of utmost importance compared to benign software detection (Zahoora et al., 2022). Therefore, the recall rate is a more crucial performance metric in determining whether ransomware is truly detected or not. The proposed method achieves a recall rate of 98.47%, a significant improvement over other baseline methods. These results demonstrate the effectiveness of the proposed detection method for zero-day ransomware detection.

Table 8
Comparison with other machine learning methods.

Methods	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
DT	82.97	86.49	48.86	75.71
RF	94.02	91.26	87.86	92.66
SVM	78.32	58.05	90.84	76.79
KNN	92.95	83.97	90.91	91.20
AdaBoost	94.91	89.71	93.13	93.88
Proposed	96.02	91.49	98.47	96.31

Fig. 9 presents the loss function and accuracy curves on both the training and test sets. As the number of training iterations increases, the loss function value gradually decreases and converges to zero, while the accuracy value gradually converges to the maximum value.

The evaluation metrics of the proposed framework on the training and test sets are presented in Fig. 10. As the number of iterations increases, the values of precision (P), recall (R), and F1-score oscillate upwards and converge at the maximum value. This indicates that the proposed detection framework is effective in detecting zero-day ransomware.

Fig. 11 shows the PR plots of the baseline method and the proposed SA-CNN-IS method for zero-day ransomware detection. The proposed method achieves a maximum value of 0.90, which is significantly better than the other baseline methods.

ROC is a tool for measuring unevenness in classification and ROC curves can evaluate the merit of a binary classifier and can be used to assess the performance of zero-day ransomware detection. Fig. 12 depicts the ROC_AUC curve plots for the baseline approach and the proposed SA-CNN-IS method for zero-day detection. As can be observed from the figure, the proposed technique achieves the highest 98%, which is significantly better than other baseline methods, proving that the proposed method can be effective for early zero-day ransomware detection.

4.5. Compare with deep learning methods

In previous work on zero-day ransomware detection, several researchers have proposed a zero-day malware detection process in the form of images (Kim et al., 2018; Won et al., 2022), e.g., Venkatraman and Alazab (2018) proposed using a visualization-based approach for zero-day detection of malware. Barros et al. (2022) similarly transform malware into images and use metric learning and zero-shot learning to detect zero-day exploit attacks. To explore the effectiveness of ransomware visualization combined with deep learning for zero-day ransomware detection, this paper follows the same data partitioning in

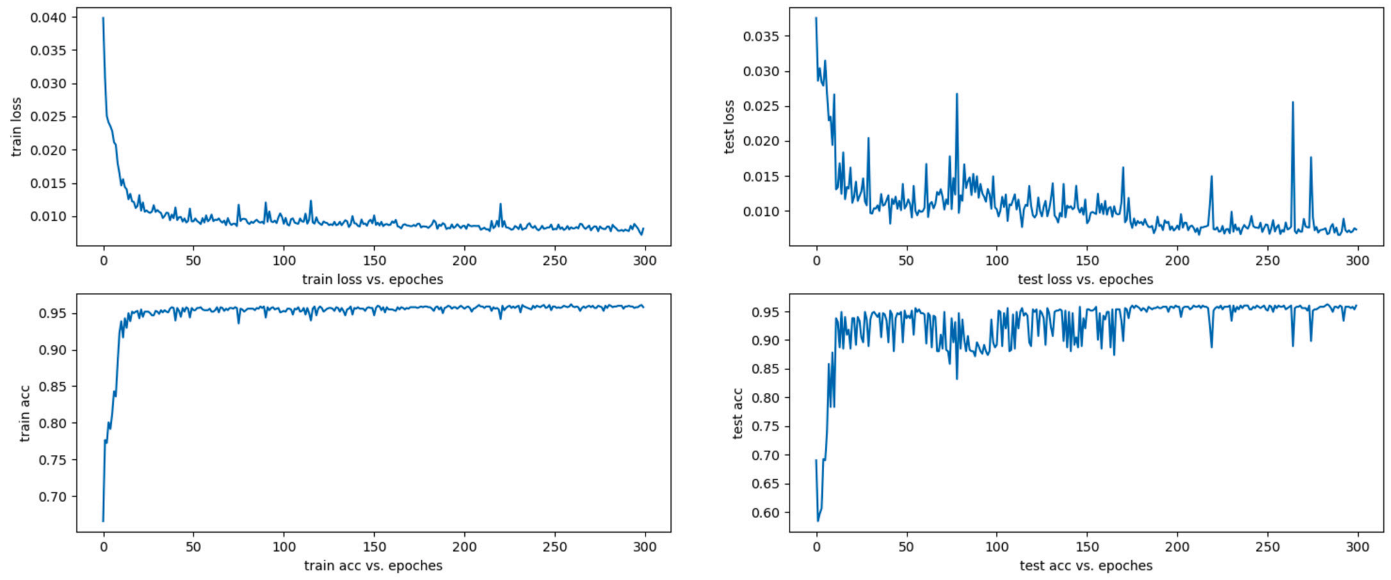


Fig. 9. The loss function and accuracy curves on the training and test sets.

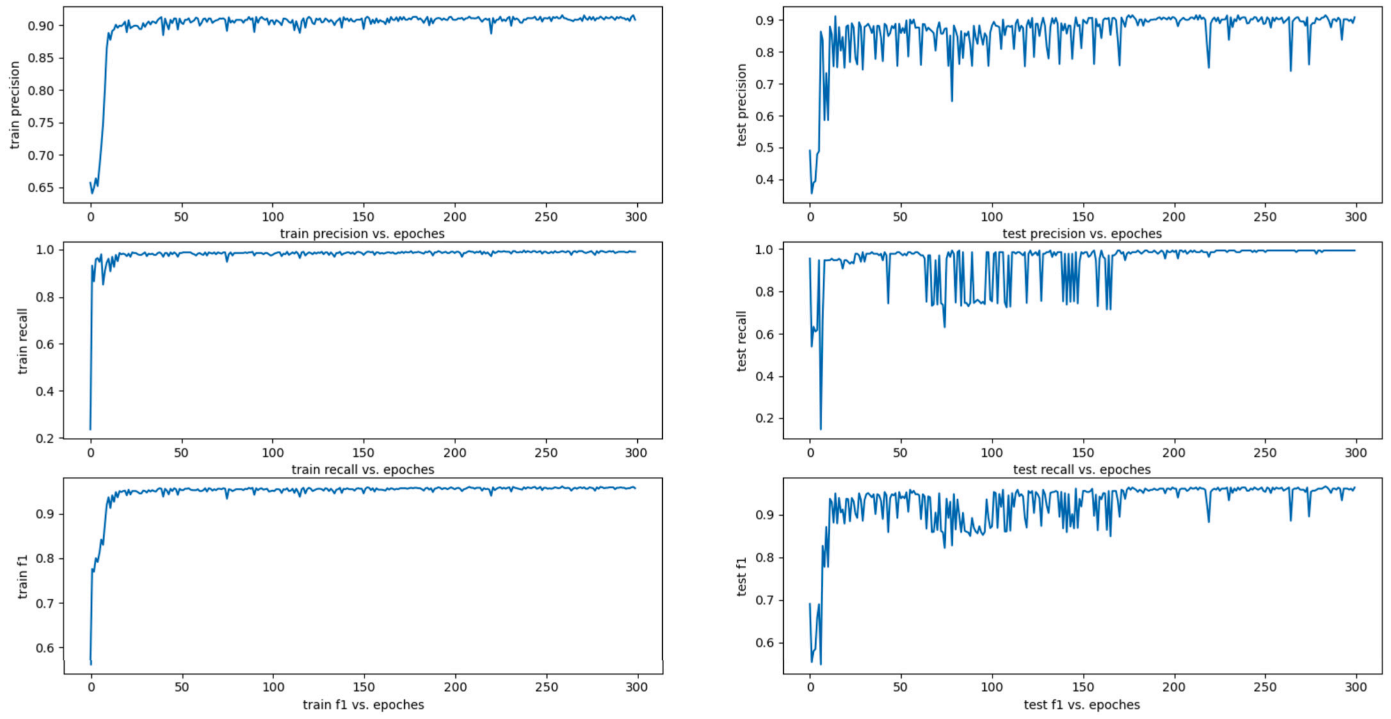


Fig. 10. The evaluation metrics of the proposed framework on the training and test sets.

Section 3 and uses deep learning methods to perform zero-day attack detection experiments on ransomware grayscale images and compare them with the detection methods proposed in this paper. These classification models now tend to solve complex problems such as intrusion detection. In this paper, VGG16 (Simonyan and Zisserman, 2014), Resnet (He et al., 2016), and Googlenet (Szegedy et al., 2015) are chosen as the baseline deep learning methods, and experiments are conducted. Table 9 depicts comparison of the proposed method with different deep-learning methods

As can be observed from Table 9, the proposed method has significant improvement in four evaluation metrics compared to zero-day ransomware detection using grayscale images, while it has shorter training time and faster detection speed due to the use of PE header as features and a lightweight detection model built based on zero-shot learning. Since ransomware is fast spreading and has many variants, the detection model must be constantly updated to understand the vulnerabilities and threats. Therefore, the execution time of the model is important, as fast model updates and rapid detection of ransomware are an important part of early detection, which can reduce the risk and loss to the enterprise. Table 10 gives the distribution of the time required to build the detection framework, where pre-processing is the phase of feature extraction for ransomware using the pefile library.

Table 11 compares the execution time of the proposed approach with three recently selected works. Among them, the method proposed by Zahora et al. has the smallest gap with the method proposed in this paper, where their proposed method requires 30 seconds to extract dynamic features of a ransomware sample in a secure environment (Sgandurra et al., 2016; Zahoora et al., 2022), while this paper takes an

Table 9
Comparison with deep learning methods.

Methods	Accuracy	Precision	Recall	F1-score
Vgg16 (Simonyan and Zisserman, 2014)	92.09	84.51	89.55	90.64
Resnet (He et al., 2016)	88.01	81.34	93.97	87.20
Googlenet (Szegedy et al., 2015)	85.39	75.37	94.39	83.81
Proposed	96.02	91.49	98.47	96.31

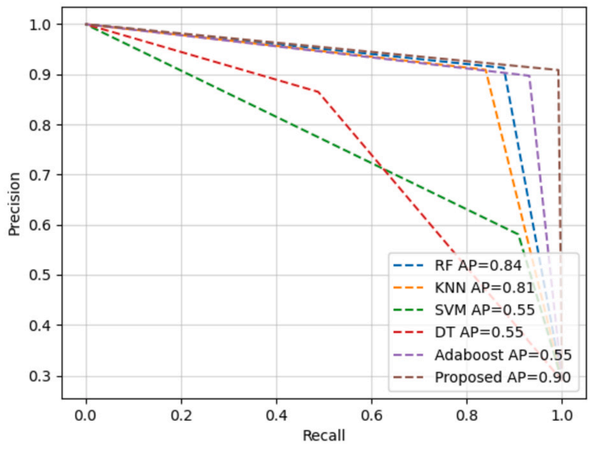


Fig. 11. PR curves.

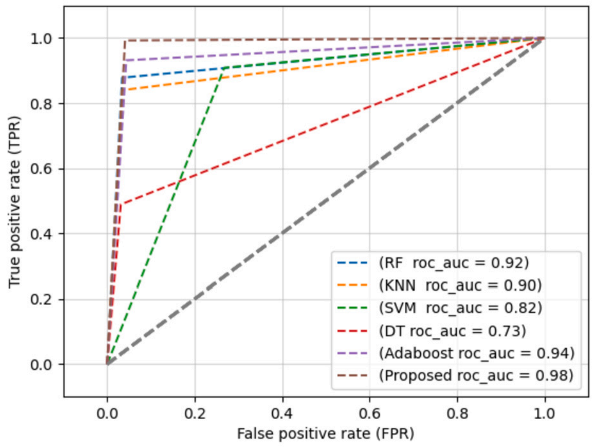


Fig. 12. ROC_AUC curve.

Table 10
Time taken to build the detection framework (unit: seconds).

Phase	Time (s)
Preprocessing	409.19
Attribute learning	1.16
Inference stage	31.14
Total	441.49

average of only 0.3 seconds to extract static features of ransomware using the pefile library.

4.6. Comparison with previous zero-day ransomware detection

The effectiveness of the proposed earlier zero-day ransomware detection methods is also compared with the most current ransomware detection methods, some of which apply to zero-day scenarios. These methods employ different features for detection. Table 12 presents a comparison of the proposed method with other detection methods across four evaluation metrics. It indicates that the proposed method not only detects zero-day ransomware but also outperforms other methods in all evaluation metrics, with a recall rate of 98.47%.

Table 11
Comparison of the proposed method with previous work in terms of execution time (unit: seconds).

Methods	Training time (s)	Test time (s)
Rezaei et al. (2021)	978.33	108.34
Zahoora et al. (2022)	100.84	15.53
Barros et al. (2022)	4448.12	2136.34
Proposed	40.82	4.32

Among the compared methods, the DCAE-ZSL-HVE proposed by Zahoora et al. (2022) uses the same ransomware dataset as this paper. Their approach employs a full set of dynamic features, including API sequences, file access, registry access, drop files, file extension search, and threatening text. The extracted core features are then trained and tested using integrated learning. In comparison, the proposed detection framework utilizes static features of the PE header, and its feature extraction process does not require the execution of ransomware, making it more secure than dynamic feature-based approaches. Moreover, the feature extraction time is shorter and faster, which is crucial for early detection. Additionally, the detection model based on the self-attentive mechanism proposed in this paper can fully exploit the correlation between the input features and improve the detection performance of the model.

4.7. Ablation experiment

In pursuit of a deeper comprehension of the proposed early zero-day detection framework, an ablation study was undertaken, focusing on two critical components: the dimensionality of the core attribute $\mathcal{T}$ and the self-attentive mechanism. The study initially explored the influence of the core attribute values' dimensionality on the detection results by adjusting $\mathcal{T}$'s dimensionality from 15 to 40. As depicted in Fig. 13, optimal detection performance was observed with the dimensionality set at 20. Both lower and higher dimensionalities adversely affected the detection model's performance. Consequently, this paper sets the dimensionality value of the core attribute $\mathcal{T}$ at 20.

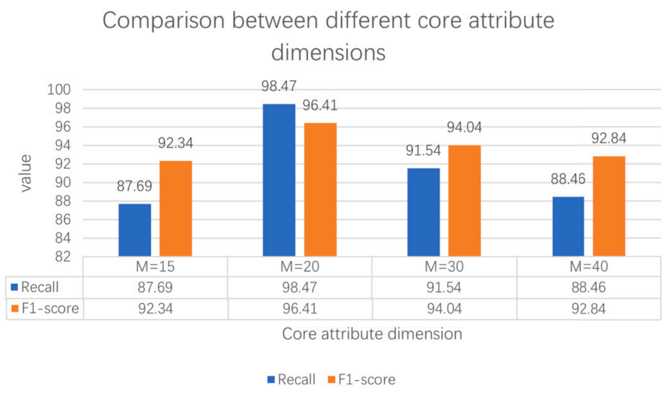
In addition, the self-attentiveness mechanism can improve the performance of the detection model by mining the correlation between the core PE header features. To investigate the impact of the self-attentiveness mechanism on detection performance, zero-day ransomware detection is conducted with and without the self-attentiveness mechanism. Fig. 14 depicts the results of the experiments with the four evaluation metrics. From the figure, it can be seen that the experimental results are significantly improved after the introduction of the self-attentiveness mechanism. This shows that the proposed SA-CNN-IS detection model can be well applied to zero-day ransomware detection.

5. Conclusion

In this paper, a new early ransomware detection method based on zero-shot learning using PE header features is proposed. The proposed method includes an attribute learning phase that utilizes a self-coding

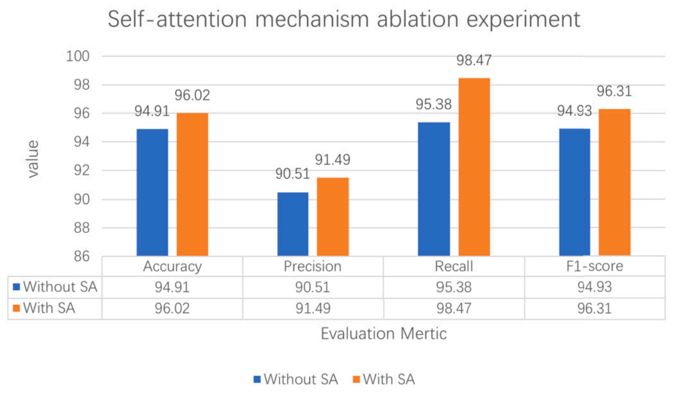
Table 12
Comparison with previous ransomware detection methods.

Methods	Feature Type	Zero-day detection	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
Rezaei et al. (2021)	PE header	No	91.13	90.76	92.06	91.40
DeepRan (Roy and Chen, 2021)	Host log	No	99.87	100	98.06	99.02
Manavi and Hamzeh (2022)	PE header	No	94.95	94.97	94.95	94.94
Xception ColSeq (Moreira et al., 2023)	PE header	No	93.73	92.95	94.64	93.75
DCAE-ZSL-HVE (Zahoora et al., 2022)	API	Yes	92.88	90.78	95.52	93.08
Malware-SMELL (Barros et al., 2022)	Image	Yes	84.00	85.00	80.00	81.00
Proposed	PE header	Yes	96.02	91.49	98.47	96.31



	M=15	M=20	M=30	M=40
■ Recall	87.69	98.47	91.54	88.46
■ F1-score	92.34	96.41	94.04	92.84

Fig. 13. The impact of core attribute dimensions on experimental performance.



	Accuracy	Precision	Recall	F1-score
■ Without SA	94.91	90.51	95.38	94.93
■ With SA	96.02	91.49	98.47	96.31

Fig. 14. Impact of the self-attention mechanism.

network-based core feature extraction technique (AE-CAL) to learn potential semantic information between unseen and seen classes. In the inference phase, a convolutional neural network detection framework with a self-attentive mechanism is trained using the potential core attribute features extracted in the attribute learning phase. With these two components, the proposed method can achieve early zero-day ransomware detection using the extracted core features. Various experiments were conducted on a dataset containing 11 ransomware families, and the proposed method outperformed the baseline machine learning approach and previously proposed detection methods in early zero-day ransomware detection. The ablation experiments further illustrated the effectiveness of the proposed framework. However, the current system is only applicable to binary classification tasks, and future work will explore extending it to early zero-day ransomware family classification tasks.

CRediT authorship contribution statement

Mingcan Cen: Conceptualization, Data curation, Methodology, Writing – original draft, Writing – review & editing. **Xizhen Deng:** Investigation, Writing – review & editing. **Frank Jiang:** Investigation, Supervision, Writing – review & editing. **Robin Doss:** Conceptualization, Project administration, Supervision, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgements

This work is partially supported by Guangxi key research and development funding (2019AB35004;AB20238030). The authors also sincerely thank Deakin University for financial support and the anonymous referees' valuable comments and suggestions for improvement of this paper.

References

Ahmadian, M.M., Shahriari, H.R., Ghaffarian, S.M., 2015. Connection-monitor &amp; connection-breaker: a novel approach for prevention and detection of high survivable ransomwares. In: 2015 12th International Iranian Society of Cryptology Conference on Information Security and Cryptology (ISCISC), pp. 79–84.
Ahmed, Yahye Abukar, Koçer, Barış, Huda, Shamsul, Al-rimy, Bander Ali Saleh, Hassan, Mohammad Mehedi, 2020. A system call refinement-based enhanced minimum redundancy maximum relevance method for ransomware early detection. J. Netw. Comput. Appl. 167.
Akata, Zeynep, Perronnin, Florent, Harchaoui, Zaid, Schmid, Cordelia, 2013. Label-embedding for attribute-based classification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 819–826.
Al-rimy, Bander Ali Saleh, Maarof, Mohd Aizaini, Alazab, Mamoun, Shaid, Syed Zainudeen Mohd, Ghaleb, Fuad A., Almalawi, Abdulmohsen, Ali, Abdullah Marish, Al-Hadhrami, Tawfik, 2021. Redundancy coefficient gradual up-weighting-based mutual information feature selection technique for crypto-ransomware early detection. Future Gener. Comput. Syst. 115, 641–658.
Ashraf, Arslan, Aziz, Abdul, Zahoora, Umme, Rajarajan, Muttukrishnan, Khan, Asifullah, 2019. Ransomware analysis using feature engineering and deep neural networks. arXiv preprint. arXiv:1910.00286.
Athiwaratkun, Ben, Stokes, Jack W., 2017. Malware classification with lstm and gru language models and a character-level cnn. In: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, pp. 2482–2486.
Baldi, Pierre, 2012. Autoencoders, unsupervised learning, and deep architectures. In: Proceedings of ICML Workshop on Unsupervised and Transfer Learning. In: JMLR Workshop and Conference Proceedings, pp. 37–49.
Baldwin, James, Dehghantanha, Ali, 2018. Leveraging support vector machine for opcode density based detection of crypto-ransomware. In: Cyber Threat Intelligence. Springer, pp. 107–136.
Barros, Pedro H., Chagas, Eduarda T.C., Oliveira, Leonardo B., Queiroz, Fabiane, Ramos, Heitor S., 2022. Malware-smell: a zero-shot learning strategy for detecting zero-day vulnerabilities. Comput. Secur., 102785.
Cen, Mingcan, Jiang, Frank, Qin, Xingsheng, Jiang, Qinghong, Doss, Robin, 2024. Ransomware early detection: a survey. Comput. Netw. 239, 110138.
Chen, Chang-qing, Cuo, Chun, Cui, Yun-he, Shen, Guo-wei, Jiang, Chao-hui, 2021. Ransomware early detection method based on short api sequence. Acta Electron. Sin. 49 (3), 586.
Chen, Jing, Wang, Chiheng, Zhao, Ziming, Chen, Kai, Du, Ruiying, Ahn, Gail-Joon, 2017. Uncovering the face of Android ransomware: characterization and real-time detection. IEEE Trans. Inf. Forensics Secur. 13 (5), 1286–1300.

Deng, XiZhen, Cen, MingCan, Jiang, M., Lu, Meiqu, 2023. Ransomware early detection using deep reinforcement learning on portable executable header. Clust. Comput.

Feng, Yun, Liu, Chaoge, Poster, Baoxu Liu, 2017. A new approach to detecting ransomware with deception. In: 38th IEEE Symposium on Security and Privacy.

Ferrante, Alberto, Malek, Miroslaw, Martinelli, Fabio, Mercaldo, Francesco, Milosevic, Jelena, 2017. Extinguishing ransomware-a hybrid approach to Android ransomware detection. In: International Symposium on Foundations and Practice of Security. Springer, pp. 242–258.

Ganta, Venkata Gopi, Venkata Harish, G., Prem Kumar, V., Rama Koteswar Rao, G., 2020. Ransomware detection in executable files using machine learning. In: 2020 International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT). IEEE, pp. 282–286.

Guo, C., Chen, C.-q., Shen, G.w., 2020. A ransomware classification method based on visualization. Netinfo Secur. 4, 31–39.

He, Kaiming, Zhang, Xiangyu, Ren, Shaoqing, Sun, Jian, 2016. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778.

Kharaz, Amin, Arshad, Sajjad, Mulliner, Collin, Robertson, William, Kirda, Engin, 2016. {UNVEIL}: a {Large-Scale}, automated approach to detecting ransomware. In: 25th USENIX Security Symposium (USENIX Security 16), pp. 757–772.

Kim, Jin-Young, Bu, Seok-Jun, Cho, Sung-Bae, 2018. Zero-day malware detection using transferred generative adversarial networks based on deep autoencoders. Inf. Sci. 460, 83–102.

Kim, Samuel, 2018. PE Header Analysis for Malware Detection. Master's Projects, 624.

Kok, S.H., Abdullah, Azween, Jhanjhi, N.Z., 2022. Early detection of crypto-ransomware using pre-encryption detection algorithm. J. King Saud Univ, Comput. Inf. Sci. 34 (5), 1984–1999.

Lampert, Christoph H., Nickisch, Hannes, Harmeling, Stefan, 2013. Attribute-based classification for zero-shot visual object categorization. IEEE Trans. Pattern Anal. Mach. Intell. 36 (3), 453–465.

Manavi, Farnoush, Hamzeh, Ali, 2022. A novel approach for ransomware detection based on pe header using graph embedding. J. Comput. Virol. Hacking Tech., 1–12.

Masdari, Mohammad, Khezri, Hemn, 2020. A survey and taxonomy of the fuzzy signature-based intrusion detection systems. Appl. Soft Comput. 92, 106301.

McIntosh, Timothy, Kayes, A.S.M., Chen, Yi-Ping Phoebe, Ng, Alex, Watters, Paul, 2022. Ransomware mitigation in the modern era: a comprehensive review, research challenges, and future directions. ACM Comput. Surv. 54 (9), 1–36.

Microsoft, 2024. PE-format. [OL]. https://docs.microsoft.com/zh-cn/windows/win32/debug/pe-format.

Moreira, Caio C., Moreira, Davi C., de S. de Sales Jr., Claudomiro, 2023. Improving ransomware detection based on portable executable header using xception convolutional neural network. Comput. Secur. 130, 103265.

Moussaileb, Routa, Bouget, Benjamin, Palisse, Aurélien, Le Bouder, Hélène, Cuppens, Nora, Lanet, Jean-Louis, 2018. Ransomware's early mitigation mechanisms. In: Proceedings of the 13th International Conference on Availability, Reliability and Security, ARES 2018. New York, NY, USA. Association for Computing Machinery.

Moussaileb, Routa, Cuppens, Nora, Lanet, Jean-Louis, Le Bouder, Hélène, 2021. A survey on windows-based ransomware taxonomy and detection mechanisms. ACM Comput. Surv. 54 (6), 1–36.

Netto, Daniel F., Shony, K.M., Lalson, Elizabeth Rose, 2018. An integrated approach for detecting ransomware using static and dynamic analysis. In: 2018 International CET Conference on Control, Communication, and Computing (IC4). IEEE, pp. 410–414.

Oktavianto, Digit, Muhardianto, Iqbal, 2013. Cuckoo Malware Analysis. Packt Publishing Ltd.

O'Shaughnessy, Stephen, Sheridan, Stephen, 2022. Image-based malware classification hybrid framework based on space-filling curves. Comput. Secur. 116, 102660.

Perreault, Hughes, Bilodeau, Guillaume-Alexandre, Saunier, Nicolas, Héritier, Maguelonne, 2020. Spotnet: Self-attention multi-task network for object detection. In: 2020 17th Conference on Computer and Robot Vision (CRV). IEEE, pp. 230–237.

Pietrek, Matt, 1994. Peering inside the pe: a tour of the win32 (r) portable executable file format. Microsoft Syst. J., US Ed. 9 (3), 15–38.

Python, 2023. Pefile project description. [OL]. https://pypi.org/project/pefile/.

Ramesh, Gowtham, Menen, Anjali, 2020. Automated dynamic approach for detecting ransomware using finite-state machine. Decis. Support Syst. 138, 113400.

Rezaei, Tina, Manavi, Farnoush, Hamzeh, Ali, 2021. A pe header-based method for malware detection using clustering and deep embedding techniques. J. Inf. Secur. Appl. 60, 102876.

Romera-Paredes, Bernardino, Torr, Philip, 2015. An embarrassingly simple approach to zero-shot learning. In: International Conference on Machine Learning. PMLR, pp. 2152–2161.

Roy, Krishna Chandra, Chen, Qian, 2021. Deepran: attention-based bilstm and crf for ransomware early detection and classification. Inf. Syst. Front. 23 (2), 299–315.

Sgandurra, Daniele, Muñoz-González, Luis, Mohsen, Rabih, Lupu, Emil C., 2016. Automated dynamic analysis of ransomware: benefits, limitations and use for detection. arXiv preprint. arXiv:1609.03020.

Simonyan, Karen, Zisserman, Andrew, 2014. Very deep convolutional networks for large-scale image recognition. arXiv preprint. arXiv:1409.1556.

Sun, Xiaohong, Gu, Jinan, Sun, Hongying, 2021. Research progress of zero-shot learning. Appl. Intell. 51 (6), 3600–3614.

Szegedy, Christian, Liu, Wei, Jia, Yangqing, Sermanet, Pierre, Reed, Scott, Anguelov, Dragomir, Erhan, Dumitru, Vanhoucke, Vincent, Rabinovich, Andrew, 2015. Going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–9.

Van der Maaten, Laurens, Hinton, Geoffrey, 2008. Visualizing data using t-sne. J. Mach. Learn. Res. 9 (11).

Vaswani, Ashish, Shazeer, Noam, Parmar, Niki, Uszkoreit, Jakob, Jones, Llion, Gomez, Aidan N., Kaiser, Łukasz, Polosukhin, Illia, 2017. Attention is all you need. Adv. Neural Inf. Process. Syst. 30.

Venkatraman, Sitalakshmi, Alazab, Mamoun, 2018. Use of data visualization for zero-day malware detection. Secur. Commun. Netw., 2018.

Vidyarthi, Deepti, Kumar, C.R.S., Rakshit, Subrata, Chansarkar, Shailesh, 2019. Static malware analysis to identify ransomware properties. Int. J. Comput. Sci. Issues 16 (3), 10–17.

Vinayakumar, R., Soman, K.P., Velan, KK Senthil, Ganorkar, Shaunak, 2017. Evaluating shallow and deep networks for ransomware detection and classification. In: 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI). IEEE, pp. 259–265.

VirusShare, 2024. [OL]. https://virusshare.com/.

VirusTotal, 2024. [OL]. https://www.virustotal.com/gui/home/search.

Wade, Megan, 2021. Digital hostages: leveraging ransomware attacks in cyberspace. Bus. Horiz.

Wang, Wei, Zheng, Vincent W., Yu, Han, Miao, Chunyan, 2019. A survey of zero-shot learning: settings, methods, and applications. ACM Trans. Intell. Syst. Technol. 10 (2), 1–37.

Won, Dong-Ok, Jang, Yong-Nam, Lee, Seong-Whan, 2022. Plausmal-gan: plausible malware training based on generative adversarial networks for analogous zero-day malware detection. IEEE Trans. Emerg. Top. Comput.

Wu, Wei, Wang, Houfeng, Liu, Tianyu, Ma, Shuming, 2018. Phrase-level self-attention networks for universal sentence encoding. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 3729–3738.

Xie, Yi, Xiong, Yun, Zhu, Yangyong, 2020. Sast-gnn: a self-attention based spatio-temporal graph neural network for traffic prediction. In: International Conference on Database Systems for Advanced Applications. Springer, pp. 707–714.

Yakura, Hiromu, Shinozaki, Shinnosuke, Nishimura, Reon, Oyama, Yoshihiro, Sakuma, Jun, 2018. Malware analysis of imaged binary samples by convolutional neural network with attention mechanism. In: Proceedings of the Eighth ACM Conference on Data and Application Security and Privacy, pp. 127–134.

Yeh, Ching-Feng, Mahadeokar, Jay, Kalgaonkar, Kaustubh, Wang, Yongqiang, Le, Duc, Jain, Mahaveer, Schubert, Kjell, Fuegen, Christian, Seltzer, Michael L., 2019. Transformer-transducer: end-to-end speech recognition with self-attention. arXiv preprint. arXiv:1910.12977.

Young, A., Moti, Yung, 1996. Cryptovirology: extortion-based security threats and countermeasures. In: Proceedings 1996 IEEE Symposium on Security and Privacy, pp. 129–140.

Zahoora, Umme, Rajarajan, Muttukrishnan, Pan, Zahoqing, Khan, Asifullah, 2022. Zero-day ransomware attack detection using deep contractive autoencoder and voting based ensemble classifier. Appl. Intell., 1–20.

Zhang, Bin, Xiao, Wentao, Xiao, Xi, Kumar Sangaiah, Arun, Zhang, Weizhe, Zhang, Jiajia, 2020. Ransomware classification using patch-based cnn and self-attention network on embedded n-grams of opcodes. Future Gener. Comput. Syst. 110, 708–720.

Zhang, Hanqi, Xiao, Xi, Mercaldo, Francesco, Ni, Shiguang, Martinelli, Fabio, Kumar Sangaiah, Arun, 2019. Classification of ransomware families with machine learning based on n-gram of opcodes. Future Gener. Comput. Syst. 90, 211–221.

Zhu, Jinting, Jang-Jaccard, Julian, Singh, Amardeep, Welch, Ian, Harith, AI-Sahaf, Camtepe, Seyit, 2022. A few-shot meta-learning based Siamese neural network using entropy features for ransomware classification. Comput. Secur. 117, 102691.

Mingcan Cen holds a B.S. degree from Guilin University of Electronic Technology and brings over a decade of experience from both academia and industry. He is currently pursuing his Ph.D. at the Faculty of Science, Engineering and Built Environment (SEBE) at Deakin University. His research primarily focuses on information security and data-driven cybersecurity.

Xizhen Deng is a senior engineer with many years of experience in the IT industry and is currently an engineer at Zibo Big Data Center. His research interests include data security as well as the Industrial Internet of Things.

Frank Jiang received the Ph.D. degree from the University of Technology Sydney (UTS). He gained the 4 years of post-doctoral research experience at UNSW. He has published 200+ highly reputed SCI/EI indexed journals and conferences articles. His main research interests include IoT security, biologically inspired learning schemes and their applications in the context-aware systems, data-driven cyber security, predictive analytics, and blockchain techniques.

Robin Doss (SM'16) received the B.E. degree in electronics and communication engineering from the University of Madras, India, and the master's and Ph.D. degrees from the Royal Melbourne Institute of Technology (RMIT), Australia. He was a part of the Technical Services Group, Ericsson, Australia, and a Research Engineer with RMIT University. He is currently a Professor of information technology and the Deputy Head of the School of Information Technology, Deakin University, Australia. He leads a team of researchers and Ph.D. students in the broad areas of communication systems and cybersecurity with a focus on emerging domains, such as the Internet of Things (IoT), pervasive computing, applied machine learning, and ambient intelligence. His research has been funded by the National Security Science and Technology Branch of the Office of National Security in collaboration with the Defense Signals Directorate, the Australian Research Council, and industry partners. He is the Founding Chair of the Future Network Systems and Security Conference series and an Associate Editor of the Cyber-Physical Systems Journal.