UDC 519.217 Вестник СПбГУ. Прикладная математика. Информатика... 2024. Т. 20. Вып. 1
MSC 90C40

Combining dynamic and static host intrusion detection features using variational long short-term memory recurrent autoencoder

V. H. Nguyen, N. N. Tran

Le Quy Don Technical University, 236, ul. Hoang Quoc Viet, Hanoi, 140000, The Socialist Republic of Vietnam

For citation: Nguyen V. H., Tran N. N. Combining dynamic and static host intrusion detection features using variational long short-term memory recurrent autoencoder. *Vestnik of Saint Petersburg University. Applied Mathematics. Computer Science. Control Processes*, 2024, vol. 20, iss. 1, pp. 34–51. https://doi.org/10.21638/11701/spbu10.2024.104

Despite the many advantages offered by Host Intrusion Detection Systems (HIDS), they are rarely adopted in mainstream cybersecurity strategies. Unlike Network Intrusion Detection Systems, a HIDS is the last layer of defence between potential attacks and the underlying OSs. One of the main reasons behind this is its poor capabilities to adequately protect against zero-day attacks. With the rising number of zero-day exploits and related attacks, this is an increasingly imperative requirement for a modern HIDS. In this paper variational long short-term memory — recurrent autoencoder approach which improves zero-day attack detection is proposed. We have practically implemented our model using TensorFlow and evaluated its performance using benchmark ADFA-LD and UNM datasets. We have also compared the results against those from notable publications in the area.

Keywords: HIDS, anomaly detection, variational autoencoder, deep learning.

1. Introduction. With the increasingly complex and dynamic nature of modern networks, Intrusion Detection Systems (IDS) are playing a crucial role in cyber security. The spate of recent high profile and high impact cyber security incidents have highlighted the importance of such technology [1].

In recent years, the number of new types of attack has drastically increased. This creates difficulty for IDSs, as for reliable operation they need to extract information out of such large-scale data. Moreover, hackers have developed many automated toolkits that can make minor modification to attack behaviour, which are sufficient for it be treated as a zero-day attack by IDSs. As a result, zero-day attack samples have been rapidly gaining prevalence.

Host IDS (HIDS) is a variant of IDS that operates purely on the host operating system. They have gained popularity in recent years due to the increasing number of security threats. The heterogeneity of modern networks means that Network-based IDSs (NIDSs) often adopt a more abstract approach to detecting intrusions e.g. monitoring network packet content, structure and conformity. Whereas a HIDS is a much more device-centric approach that monitors resources within the host e.g. log files, file changes or system calls (methods used to request services from the OS kernel). Thus, a HIDS is able to offer a superior level of protection uniquely tailored to its host.

HIDS are arguably more complex; not only do they have to co-exist on the host OS but must maintain a minimal system footprint. Most importantly, users are far less tolerant of false alarms from a HIDS compared to a NIDS. Therefore, detection accuracy is a priority for any HIDS approach. Unfortunately, modern OSs facilitate high levels of interconnectivity with remote resources and services (e.g. Cortana, Siri and Dropbox) that can directly influence their behaviour. Considering the extra attack vectors this introduces into an OS, it is unsurprising that zero-day attacks are becoming increasingly difficult for HIDS to detect with satisfactory levels of accuracy.

In this paper, we propose our novel method for HIDS-based detection of zero-day threats based on system calls. We make the following novel contributions:

• **Our variational long short-term memory (LSTM) recurrent autoencoder (VLRAE) technique** to build a model for Intrusion Detection by leveraging the capabilities of LSTM cells and recurrent deep neural networks. Unlike traditional approaches, this technique is fully capable of analysing time-dependent data which maintains the ordered characteristic of system calls in the extracted features.

• **A holistic feature extraction approach** that extracts both static (by VLRAE) and dynamic variables. Unlike existing approaches, we do not rely on static feature alone and have developed a technique to combine the power of both variable types, to improve the performance of the model. That model we called **VLRAE4ID**.

• **An outlier detection algorithm** that uses an isolated forest technique to combine the high-dimension extracted feature vector to construct the RandomForest classifier.

The remainder of this paper is organised as follows. Section 2 presents background information for the key concepts featured in this paper. Section 3 provides a review of notable existing work in related fields. Section 4 details our proposed methodology and algorithms. Section 5 presents our evaluation results conducted using benchmark datasets. Section 6 discusses the results of our evaluation and their wider significance. Finally, the paper is concluded and future work outlined in Section 7.

2. Background.

2.1. Long short term memory. LSTM is a concept that was first introduced in [2], and was originally designed to resolve the vanishing gradient problem of a traditional Recurrent Neural Network (RNN).

A LSTM is a single block that is itself a RNN, but most commonly several blocks are conjoined to construct LSTM networks. They have risen to prominence within the deep learning paradigm, and are being applied to a wide variety of research problems. This is because they provide a rudimentary solution to learning the links between cause and effect, which is one of the major problems faced in AI.

Fundamentally, LSTMs help preserve the error that can be backpropagated through time and layers. By maintaining a more constant error, they allow for learning to continue over numerous iterations, thereby understanding links between cause and effect.

A LSTM block consists of four main components; a cell (responsible for remembering values) and three gates to replicate conventional artificial neurons. These gates control the data flow through the cell using sigmoid functions, which maintain the range of 0–1.

These gates include an **Input Gate** (controls the extent to which a new value flows into the memory), **Output Gate** (controls the extent to which the value in memory is used to compute the output activation of the LSTM block) and **Forget Gate** (controls the extent to which a value remains in memory); each of which has its own parameters (e.g. weights and biases). An example LSTM block is illustrated in Figure 1.

2.2. Variational LSTM — recurrent autoencoder. The architecture of RNNs has been used for many years to form a directed cycle by utilising the connections between units. This architecture allows RNNs to exhibit dynamic temporal behaviour, which makes them suitable for capturing time dependencies in temporal data. RNN models have been developed and successfully applied to many research problems, such as time series

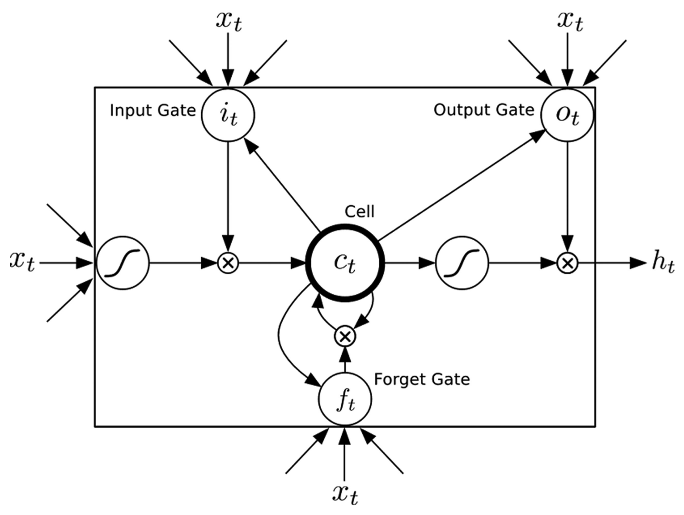

Figure 1. Example LSTM Cell [6]

prediction [3], machine translation [4], handwriting recognition [5] and human activity recognition [6]. Recently a new RNN model based on Variational Bayes (VB): the Variational Recurrent Autoencoder (VRAE) has been proposed in [7]. This model is similar to an autoencoder, in that it has two blocks: encoder and decoder. The encoder learns a mapping from data to a latent representation, and the decoder learns latent representation to data. This model is based on autoencoding VB [8], which maps the data to a distribution over latent variables. VRAE is a model that combines the architecture of RNN and autoencoding VB, so it allows the mapping of time sequences to a latent representation. Hence, it enables efficient, large scale unsupervised variational learning on time sequences.

According to [7], RNN can take an input sequence $x = (x_1, x_2, ..., x_T)$ and transfer to a time-dependent hidden state h that is a function not only of the input x_t, but also of the previous hidden state of the network. At each timestep t, the RNN reads input $x_t \in R^d$ and computes its hidden state $h_t \in R^p$ by

$$h_t = f(x \leqslant t) = f_\theta(x_t, h_{t-1}),$$

where f is a deterministic non-linear transition function, θ is the parameter set of f. This architecture was combined in the autoencoder model and makes the latent representation of the autoencoder be a function of time-dependent input. Variational autoencoder replaced the latent representation z of sequence input x with stochastic variables. The decoding distribution (or recognition model) $q_\varphi(z|x)$ approximates the true, unknown posterior $p(z|x)$. The encoding distribution $p_\theta(x|z)$ implements a graphical model. In [7] the “reparametrization trick” was used in which the random variable $z \sim p(z|x)$ was reparametrised as a deterministic variable $z = \mu + \sigma\varepsilon$ with $\varepsilon \sim N(0, 1)$. The distribution over Z is obtained from the last state of the RNN, h_{end}, such that

$$\begin{gathered} h_{t+1} = \tanh\left(W_{\text{enc}}^T h_t + W_{\text{in}}^T x_{t+1} + b_{\text{enc}}\right), \\ \mu_z = W_\mu^T h_{\text{enc}} + b_\mu, \\ \log(\sigma_z) = W_\sigma^T h_{\text{enc}} + b_\sigma. \end{gathered}$$

After that, z is sampled from this encoding and the initial state of the decoding RNN is computed with one set of weights:

$$h = \tanh\left(W_z^T + b_t\right),$$
$$h_{t+1} = \tanh\left(W_{\text{dec}}^T h_t + W_x^T x_t + b_{\text{dec}}\right),$$
$$x_t = \text{sigmoid}\left(W_{\text{out}}^T h_t + b_{\text{out}}\right).$$

As VAE, VRAE can be trained with Stochastic Gradient VB (SGVB) with the target of approximating the true posterior $p(z|x)$ by $q(z|x)$ and then optimising a lower bound on the log-likelihood:

$$L = D_{\text{KL}}(q_\varphi(z|x)|p(z)) + E_{q_\varphi(z|x)}[\log p_\theta(x|z)],$$

where D_{KL} is Kullback—Leibler divergence.

3. Related work.

3.1. System call-based HIDS. Host intrusion detection remains a persistent and difficult problem to solve. Despite the development of various novel methods and techniques, very few are in mainstream use. The main reason behind this is that due to the complex and diverse nature of modern operating systems and the threats they face. Reliable and accurate threat detection is a problematic and challenging research area.

C. Warrender et al. [9] compare four system call-based IDS methods (STIDE, t-STIDE, RIPPER and HMM) to determine their capabilities to reliably characterise normal behaviour and correctly identify intrusions. Each method was analysed against various open datasets. The authors state their belief that the choice of data stream is more important than analysis method. Ultimately, they conclude that no single method consistently gave the best results, which summarises the challenges faced in this area.

Modern research into HIDS tends to focus on system call analysis, some of the most current and notable works in this field are discussed in the remainder of this subsection.

F. Maggi et al. [10] propose an unsupervised HIDS based on system calls' arguments and sequences. The authors propose the use of a clustering phase alongside their Markov model-based approach. This allows correlations to be identified between the various arguments passed to the same single system call. Hence, the model can automatically infer different ways of using the same system call. Their evaluations were performed on the IDEVAL dataset and yielded results of 100 % sequence Detection Rate (DR), 1.6 % sequence False Positive Rate (FPR), which is also called as False Alarm Rate (FAR).

M. Xie et al. [11] investigate the potential application of frequency-based algorithms to reduce the computational overheads associated with analysing the ADFA-LD host-based IDS dataset. The authors' proposed approach involves transforming system call traces into lower dimension frequency vectors using principal component analysis. Then k-nearest neighbour (kNN) and k-means clustering (kMC) methods with various distance measures (Euclidean, Minkowski, Cosine and Correlation) were used to validate effectiveness. The authors' results demonstrated that kNN was unable to effectively detect the attacks using the low dimensional frequency vectors, regardless of the distance function used and its performance was worsened. However, the kMC algorithm is able to achieve an accuracy rate of over 60 % with a FPR of lower than 20 % for most types of attack.

The authors of [11] developed their previous work by investigating the application of one-class Support Vector Machine (SVM) for analysing the ADFA-LD dataset [12]. They aspire to improve the dictionary computation time required by the model proposed by G. Creech et al. [13]. The proposed method involves the use of a short-sequence model and one-class SVM. The short sequences are created by transforming the traces into a fixed size matrix and duplicate entries are eliminated. The SVM algorithm depends upon strong separation between normal and abnormal behaviour, which is enforced artificially by weighting each short sequence with its corresponding system call frequency, computed from training traces. Using the optimum matrix length of 5, the proposed approach was evaluated using the ADFA-LD dataset. The results showed an average accuracy rate of 70 %, with a FPR of 20 %. The authors [14] propose an algorithm to extract distinct short sequences from traces of system calls. Features after extracting are feed to the machine learning model such as SVM, kNN to detect anomalies. Y. Zhang et al. [15] also propose the algorithm to extract semantics features of system calls sequence and then input into the multi-channel Text-CNN. F. Osamor and B. Wellman [16] propose a method to enhance the accuracy of intrusion detection by combining CNN and LSTM models. They use the CNN model to reduce the dimension of the system call sequence first. The output vector of CNN is fed to LSTM network to determine the final result.

M. Anandapriya and B. Lakshmanan [17] propose a HIDS based upon semantic system call patterns. Their method involves extracting semantic features from the raw call sequence and decisions are made using the Extreme Learning Machine (ELM) algorithm (a new type of neural network). This is similar to the system call method proposed by G. Creech and J. Hu in [13]. Their approach involves the use of a semantic pattern analysis of system calls. It proposes that context-free grammar is applied to system call traces. Unlike existing methods, the authors propose a more semantically-sound approach of using contigious and non-contigious system calls. This semantic feature is also utilised with an ELM decision engine. Their solution was evaluated on KDD98, UNM and ADFA datasets, with the authors claiming a DR of 100 % with a FPR of 0.6 % and excellent portability characteristics. Nevertheless, this method suffer from the combinatorics explosion. With the n-grams models, the size of the vector space model is very high and in the case of Creech and Hu approach, the combinatorics is much more higher with an estimated feature space dimension of 1016 which makes this model intractable for common hardware.

Y. Lu and S. Teng [18] propose the sequence representation model for system calls. They convert system calls into embedding vectors by constructing embedding vectors for all system calls and build the sequences with system calls embedding and weighting. The experimental result with ADFA-LD dataset archives a low TPR rate. L. Ouarda et al. [19] focus on enhancing the performance of the HIDS. They propose using textual distance to classify the system call trace as attack or non-attack based on interclass decoupling and intra-class coupling. The experimental results on ADFA-LD dataset present a good detection performance in real-time.

3.2. LSTM-RNN in intrusion detection. As the advantages and capabilities of deep learning continue to improve, many researchers are investigating novel applications for this work. One such technique is LSTM-RNN, which we will utilise within our own methodology. There are several research works that utilise a derivative of this technique in various ways, the most notable are discussed in the remaining part of this subsection.

T.-T.-H. Le et al. [20] propose an IDS based upon a LTSM-RNN using an Adam optimiser. They evaluated their solution using the KDD'99 dataset and drew comparisons with related techniques. The authors claim their solution has an accuracy of 97.54 %, a DR of 98.95 % and a FPR of 9.98 %.

R. Staudemeyer [21, 22] proposes the use of LSTM-RNN in a NIDS with feature extraction being performed using a J4.8 decision tree. The author describes the optimum learning structure for the KDD'99 dataset (which is used for the evaluation) as being

40 input neurons connected to hidden layer of 2 memory blocks with two cells each and an output of 5 target neurons. The author claims the proposed solution achieves an accuracy of 93.72 % with 4 features, 93.69 % with 8 features and 93.82 % with all the features.

L. Bontemps et al. [23] propose a time-series anomaly detection model that is based on LSTM-RNN. Their model is trained using only normal time-series data (i.e. no malicious data) and the authors claim it is capable of predicting several time steps ahead of an input. This prediction is based upon a collective anomaly detection function that utilises a circular array. This array contains the prediction errors from a specific number of the latest time steps. If the prediction errors in the circular array are higher than a predetermined threshold and last for a certain time steps, it indicates an anomaly. The solution was evaluated using the KDD'99 dataset, using various threshold combinations. The authors were able to achieve true positive rate of between 86 % (with a 0 % FPR) and 100 % (with 63 % FPR).

J. Kim et al. [24] propose the use of a LSTM-RNN-based deep learning approach to intrusion detection, focusing entirely on the KDD'99 dataset. The paper focuses on establishing the optimum hyperparameters and measuring the achievable classification performance of the model. They stipulate that their optimum hyperparameters achieved through experimentation are a learning rate of 0.01 and a hidden layer size of 80. The results published indicate a 98.88 % DR, 10.04 % FPR and 96.93 % accuracy. When compared with similar classifier algorithms the authors claim it offered the best DR and accuracy, but FPR could be improved.

LSTM-RNN is a powerful model for time-dependent data, but in intrusion detection problem, only datasets with time-independent features (e.g. KDD'99 and NSL-KDD) were used in mentioned works above. In this paper, a variational LSTM-RNN autoencoder is proposed to work with time-dependent dataset of intrusion detection.

4. Proposed methodology. As mentioned previously, host-based intrusion data is often collected based on monitoring the system calls used by active, privileged processes. Each process is represented by a sequence of system calls used by that process from the beginning of its execution to the end. An abnormal process can be detected, intuitively through some specific system calls or specific combinations of system calls that it used. Consequently, ascertaining whether a trace of system calls is normal or abnormal should not only rely upon static features (e.g. meaning of a system call or frequency of system calls) but should also incorporate dynamic features (e.g. the order of system calls in the specific trace).

The proposed method has three key ideas. The first idea is a to use the variational LSTM — recurrent autoencoder to build a model for Intrusion Detection (VLRAE). The proposed model uses the ability of LSTM cell and recurrent deep neural network in working with time-dependent data that can keep the ordered characteristic of system calls in the extracted features. Moreover, VRAE is a generative model that can be used to generate new data similar to the input data. In anomaly detection, only normal data is used to build model, so we hope that the dynamic features extracted from VRAE can be used to reduce the false positive rate, which is often high in anomaly detection models due to the lack of normal training data. The second idea is the method of extracting dynamic and static features to enhance the performance of proposed model (VLRAE4ID). Dynamic features are extracted from the output of VLRAE, and static features of a system call trace are extracted based on the statistical characteristics of system calls. These features are hidden representative features from the trace of system calls, which can act as a good discriminator between normal or abnormal traces. Finally, an outlier detection algorithm based on isolation forest in high-dimensions combined features dataset is used to construct the classifier.

4.1. Dynamic feature extraction. Figure 2 shows the overview of VLRAE for training the normal data in intrusion detection. Figure 3 shows the data flow in VRAE through time.

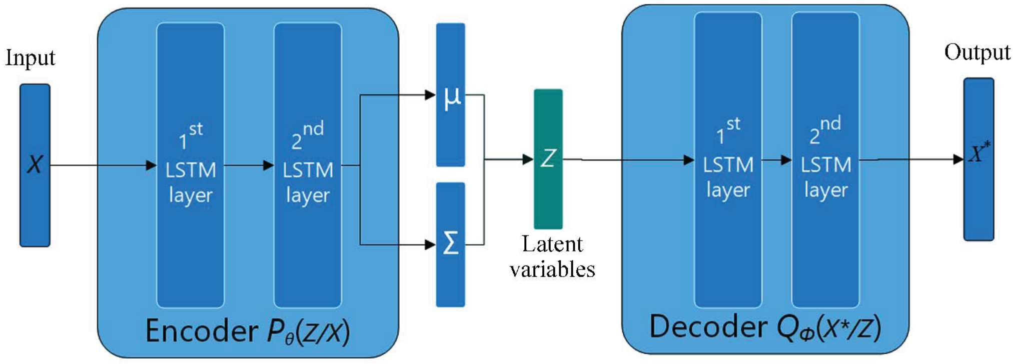

Figure 2. Overview of our proposed approach

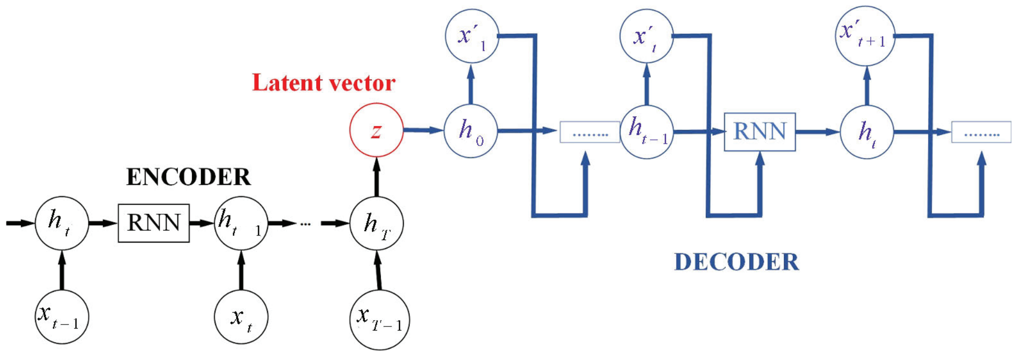

Figure 3. Dataflow in a VRAE

As mentioned in Section 2, LSTM has a special structure that can remember the relationship between inputs of cell in different times. The outputs of a LSTM recurrent network are computed using the remaining two characteristics of inputs: the value of a sample and its position at specific time. Therefore, LSTM recurrent networks can understand the context of inputs, and outputs of two same value input samples at different times may probably differ because of their relationship with previous samples. This way, when the input is a time-dependent data, the output of LSTM-RNN networks can be used as the dynamic features that stored the temporal characteristics of input sequence.

In variational autoencoders, the probabilistic encoder P_θ and decoder Q_φ both parameterise an isotropic normal distribution in the latent variable space and the original input variable space, respectively.

In our proposed VLRAE model, the encoder will implement a mapping from the input — a trace of system calls X — to a set of parameters that define an associated set of intermediate probability distributions $P_\theta(Z/X)$. For each sample from the input, the probabilistic encoder outputs the mean and variance parameter. These intermediate distributions are sampled and then the generated samples constitute a set of latent variables Z, and will be used as the input of the decoder. In the decoder block, the similar structure will map the latent variables to an output of the variational autoencoder X^*.

The final objective of VLRAE is to minimise the loss function that has two components: a log likelihood component, this means the difference between the input and output of the network X and X^*, and the KL-divergence between the true posterior $P_\theta(Z/X)$ and the approximate posterior $P(Z)$ [25].

4.2. Dynamic feature extraction. Variational autoencoder is a latent variable model, in which latent variables are underlying factors that affect data that we are observing. After the training process, latent vector z for every input trace X is vector in latent space that was generated from X and can map to $X^* \approx X$. In our methodology, we propose to use the difference between the input X and output X^* Z as a dynamic part of the features of the input trace X. VRAE is a generative model so intuitively after training with normal data, it can work effectively with the similar input data, i.e. make the difference between input X and output X^* small. With the input that differs from the training data, the difference between input and output will be larger. In the intrusion detection problem, the trace of system calls of attack process often has the “strange subsequence” that the normal processes do not have. Therefore, the difference between output and input is an important factor in determining between a normal and abnormal trace.

The details of the dynamic features extraction method using the VLRAE model are presented in Figure 4. In our proposed model, the encoder has 2 layers of LSTM cell with 128 LSTM cells in each layer. The decoder block has the similar structure with 2 layers of 128 LSTM cells. The latent vector Z has 64 dimensions. Following the VRAE training algorithm in Section 2, the process of the VLRAE dynamic feature extraction is described below.

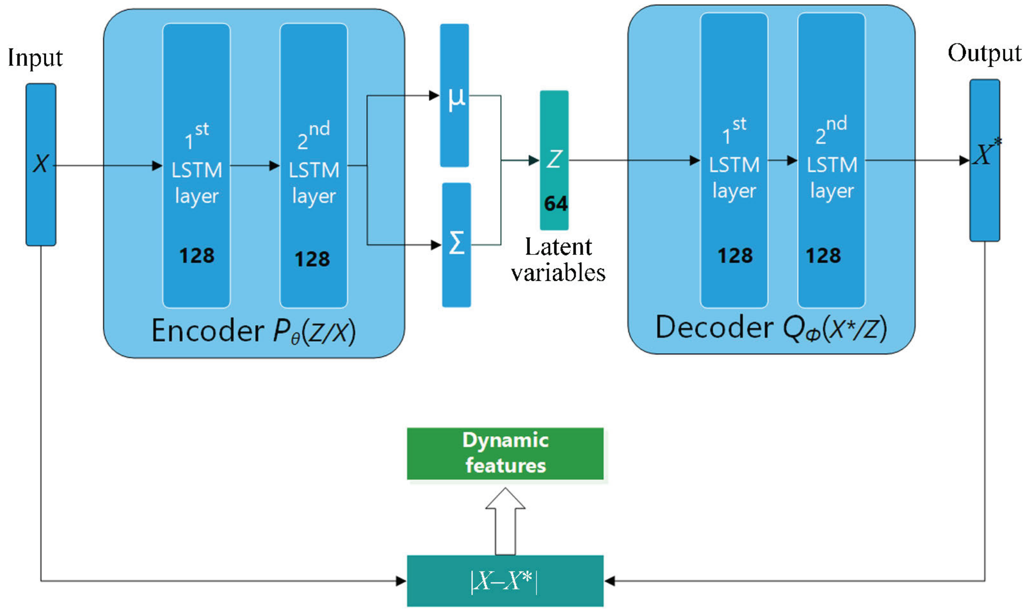

Figure 4. Dynamic feature extraction using VLRAE model

Input: Dataset of normal traces of system calls, trace X.
Output: Dynamic features of X.

1. Train input normal dataset with VRAE training algorithm in [7].
2. For each trace X, calculate the output X^* of VLRAE.
3. Create dynamic features of X from $|X - X^*|$.

4.3. Static feature extraction. We will describe the method to extract static features from a system call trace. Static features are the time independent features that acquire hidden representative characteristics from the trace of system calls, which have the capability to identify natural differences between attack and normal traces. For example, in Ubuntu OS, a brute force attack often exhibit multiple “sys_socketcall” calls to test passwords; or a buffer overflow attack may exhibit multiple “sys_brk” calls to increase the allocated memory. These observations have led us to investigate the use of statistical characteristics as a method of extracting static features.

Let S be a system call trace with length k and the elements ranging from 1 to N, where N is the maximum index of a system call:

$$S = \{s_1, s_2, \ldots, s_k \mid 1 \leqslant s_i \leqslant N,\ i = 1, ..., k\}.$$

Let f_j be the number of occurrence of the system call indexed by j in S, where $j = 1, 2, ..., N$. The frequency of a system call j in trace S can be defined as

$$\bar{f}_j = \frac{f_j}{k}.$$

We will use m max system calls that have m-largest frequency in the trace as a part of the static features vector for trace S.

The second part of the static features are selected from the operational function of system calls. In every OS, system calls are divided into specific groups depending upon their functionality. For example, in an OS running Linux Kernel of version 2.3.38, there are six groups of system calls: *arch*, *fs*, *ipc*, *kernel*, *mm* and *net*. System calls in each group have a similar functionality within the OS. The information of the frequency of each group in a trace of system calls is one of the static features that will be used to identify natural differences between attack and normal traces.

The algorithm for extracting these static features is described below.

Input: Traces of system calls $S = \{s_1, s_2, ..., s_k \mid 1 \leqslant s_i \leqslant N,\ i = 1, ..., k\}$.
Output: Static features of S.

1. Compute the frequency of each system call j in S

$$\bar{f}_j = \frac{f_j}{k}, \quad j = 1, ..., N,$$

where f_j is the number of occurrence of j in S.

2. Create SF_m, the vector of m system calls that have the largest frequency

$$SF_m = \{s_{j_1}, s_{j_2}, ..., s_{j_m} \mid \bar{f}_{j_u} \in \mathrm{MAX}_m(\bar{f}),\ u = 1, ..., m\}.$$

3. Transform S from system calls space to vector G in group space

$$G = \{g_1, g_2, ..., g_k \mid g_i = \mathrm{systemcall\_group}(s_i), 1 \leqslant g_i \leqslant M,\ i = 1, ..., k\},$$

where systemcall_group(s_i) is the group of the system call s_i, and M is the number of system call groups in the considered operating system.

4. Compute the frequency of each system call group for G

$$RF = \left\{ \bar{r}_1, \bar{r}_2, ..., \bar{r}_M \;\middle|\; \bar{r}_j = \frac{r_j}{k},\ j = 1, ..., M \right\},$$

where r_j is number of occurrence of j in G.

5. Create static features of X from SF_m and RF

$$\text{static\_feature} = \{s_{j_1}, s_{j_2}, ..., s_{j_m}, \bar{r}_1, \bar{r}_2, ..., \bar{r}_M\}. \tag{1}$$

4.4. Anomaly detection algorithm. In machine learning-based anomaly detection systems, only normal data is used to train the model [26]. In our approach, we will use the isolation Forest (iForest) model. iForest builds an ensemble of iTrees for a given data set; anomalies are those instances which have short average path lengths on the iTrees. There are two training parameters and one evaluation parameter in this method: the training parameters are the number of trees to build and subsampling size; the evaluation parameter is the tree height limit during evaluation. It has showed that iForest's detection accuracy converges quickly with a very small number of trees; it only requires a small subsampling size to achieve high detection accuracy with high efficiency; and the different height limits are used to cater for anomaly clusters of different density.

5. Evaluation and results. Our classification models were implemented using TensorFlow. All of our evaluations were performed using GPU-enabled TensorFlow running on a 64-bit Ubuntu 16.04 LTS PC with an Intel Xeon 3.60GHz processor, 16 GB RAM and an NVIDIA GTX 1080 GPU. To perform our evaluations, we have used the ADFA-LD, KDD98 and UNM datasets. These datasets are considered as benchmarks within HIDS research. Furthermore, using these datasets assists in drawing comparisons with existing methods and research:

As with the majority of other intrusion detection research, in this paper, we will be using the True Positive Rate (TPR) — attack data correctly classified as an attack, and False Positive Rate (FPR) — normal data incorrectly classified as an attack:

$$\text{TPR} = \frac{\text{number of detected attacks}}{\text{number of attacks}} \cdot 100\,\%, \tag{2}$$

$$\text{FPR} = \frac{\text{number of false alarms}}{\text{number of normal validation}} \cdot 100\,\%. \tag{3}$$

5.1. Dataset. For evaluating the performance of VLRAE (that extracts only dynamic feature) and VLRAE4ID (that extracts both dynamic and static features), we have used two well-known system call datasets: UNM developed by the University of New Mexico [27] and ADFA-LD by the Australian Defence Force Academy [28].

UNM dataset is an old dataset, that was collected in 1998 and has been using as a common benchmark for system call anomaly detection for several years [9]. The dataset contains many traces of system calls, that were collected at UNM on Sun SPARCstations running unpatched SunOS 4.1.1 and 4.1.4. Each trace of system calls was generated from the execution of one process. However, attack techniques in UNM are old, and can be accurately detected using numerous methods [9, 29, 30]. In this experiment, we have selected from UNM datasets a STIDE dataset, that has a large number of traces for training and testing. The attack on the STIDE is a denial of service attack that affects the memory request of any program in execution. The STIDE has 105 traces of intrusion and 13 726 traces of normal data.

ADFA-LD was developed by G. Creech and J. Hu [13]. It is a modern benchmark dataset used in the evaluation of HIDS. This data set uses a realistic and modern system configuration with small security flaws, which can be exploited incrementally to provide a full system compromise. The dataset was created by recording the system calls whilst the system was attacked by a certified penetration tester using current best-practice methods. Some of these attacks include web-based exploitation, simulated social engineering, poisoned executables, remote password brute force attacks and system manipulation using the C100 webshell. The ADFA-LD dataset consists of 833 normal system call traces for training the IDS, 4373 normal system call traces for evaluating FPR and 746 test traces with 60 different attack sets, each consisting of multiple system call traces.

For maintain the training : validation : intrusion testing ratio of 1:1:5 for ADFA-LD, we use only 100 traces of normal data to train the model, and 500 traces for validation.

5.2. Experiment. The procedure of our experiment for evaluating UNM and ADFA-LD datasets using VLRAE4ID is as follows:

- training VLRAE;
- evaluating VLRAE and VRAE4ID model with validation and testing attack dataset;
- evaluating VLRAE4ID in misuse intrusion detection problem.

5.2.1. Training VLRAE. In anomaly intrusion detection, the classifier model is built using only normal data. We also use only normal data to train our VLRAE (Figure 4). One process will call many system calls during its life. The average length of of a system call trace is around 500 but many sequences can be larger than 1000. LSTM-RNN networks can work with different lengths of input sequence, and can learn to remember the relationship through hundreds of steps. But with overly long input sequences, LSTM still has the same "vanishing gradient problem" as recurrent neural networks. We use the n-gram method to overcome this problem. Each sequence will be divided into subsequence with the length n. To remember as much as possible of the n-length subsequence, we use the n-gram overlap method. The n-gram length was set to 30, a value selected based on the empirical evidence from our experiments. We will summarise the 30 maximum largest error of n-gram subsequence to evaluate the output of one trace. This value is also the dynamic feature of this trace.

The dimension of the static feature was selected based upon the number of system call groups utilised by the host OS. For ADFA-LD, the host OS is Linux 2.3.38 which has 6 groups of system calls, therefore in (1) $M = 6$. In our preliminary experiments, we chose $m = 5$ system calls that have the largest frequency in each trace to create a static feature. In this way, the combined feature for every trace of system calls has 11 dimensions.

After the training process, we evaluate the convergence of the model using the training, validation and attack datasets. Figure 5 shows the absolute error of vector output X^* and the vector input X for the UNM STIDE dataset. The result shows the perfect separation of the normal and attack data through VLRAE. Therefore, all intrusion data was detected without any false alarm. The Receiver Operating Characteristic (ROC) (Figure 6) of Denial of Service attack detection has a perfect AUC = 1.0. Because of that, for our evaluation model, we will focus on ADFA-LD, a more complicated dataset with modern types of attack.

Figure 7 shows the absolute error of vector output X^* and the vector input X for the ADFA-LD dataset. The result shows that errors of traces in validation data are similar to errors of training data, and obviously, errors of attack data are bigger. Using a simple threshold selection, we can receive the sequence of FPR and TPR values (see (2) and (3)) and can build the ROC. Figure 8 presents the ROC of using VLRAE for intrusion

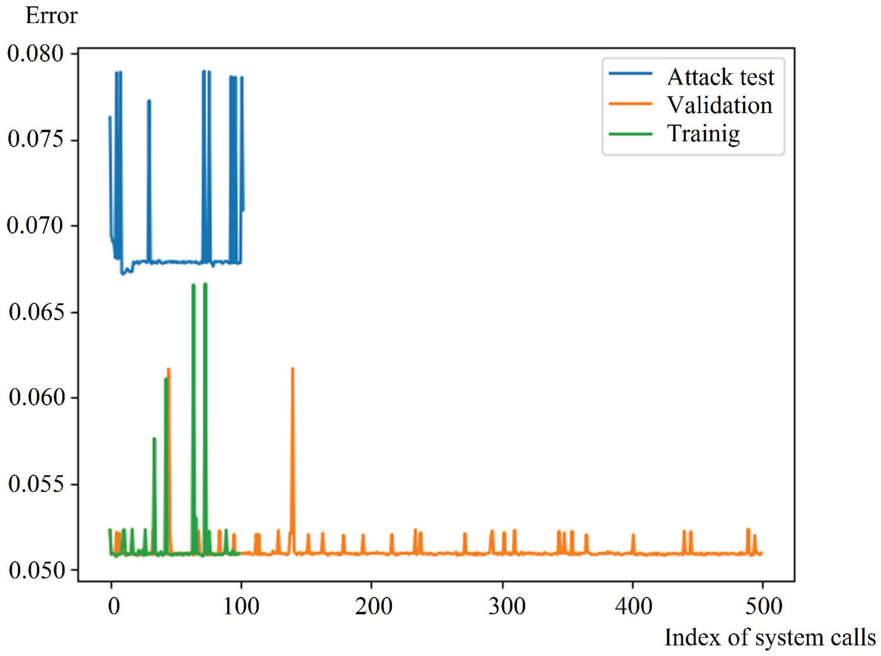

Figure 5. Difference between the input and output vectors with UNM STIDE dataset

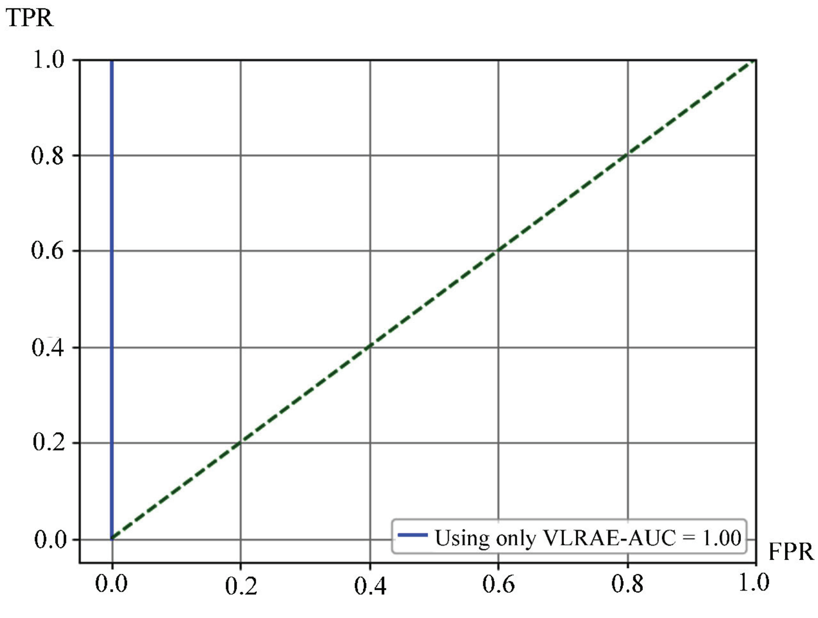

Figure 6. ROC curve of intrusion detection using VLRAE with UNM STIDE dataset

detection with ADFA-LD dataset. The AUC of this curve obtains 0.91, which is higher than in [11, 12], but slightly lower than in [13].

5.2.2. Evaluation of VLRAE4ID. The error between the output and input of VLRAE is a good factor to distinguish normal or attack traces. However, to enhance the perfor-

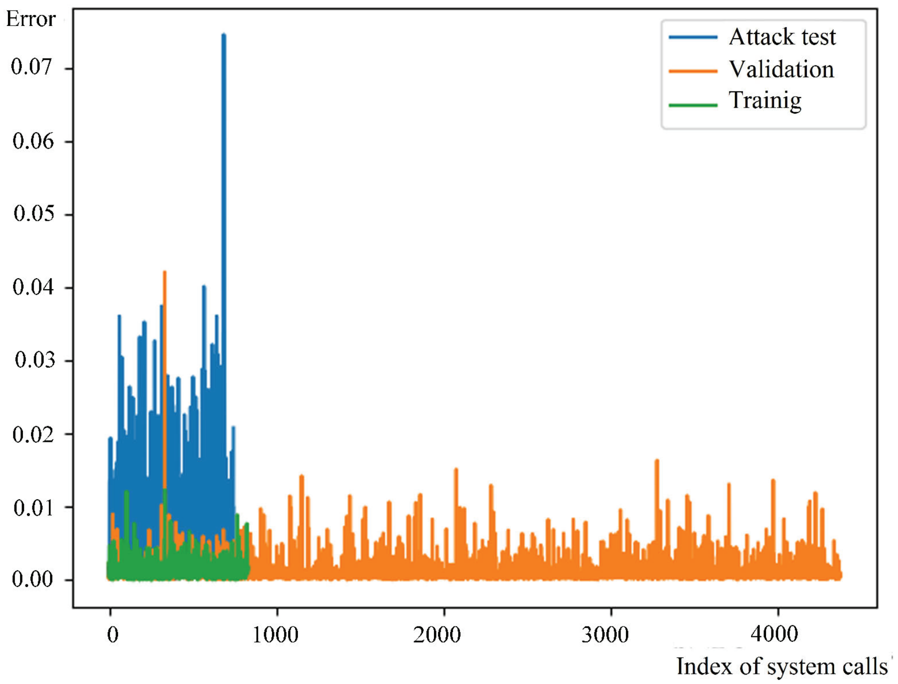

Figure 7. Difference between the input and output vectors with ADFA-LD dataset

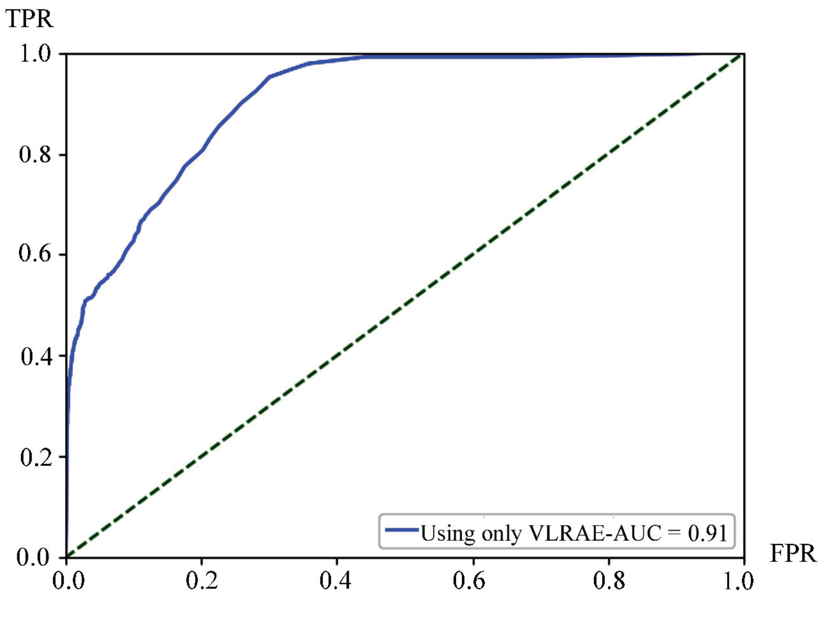

Figure 8. ROC curve of intrusion detection using VLRAE with ADFA-LD dataset

mance of the intrusion detection model, other features are required. We will use the static and dynamic features described in Section 4 to train the intrusion detection model.

Figure 9 shows the ROC of VLRAE4ID and VLRAE for ADFA-LD. As illustrated in this figure, the AUC of the proposed model is larger than VLRAE. In the middle of the FPR range (10–30 %), the TPR values are high and as good those published in [13], and better than other papers [11, 12]. VLRAE4ID achieves a TPR of 90.03 %, with a 12.97 % FPR.

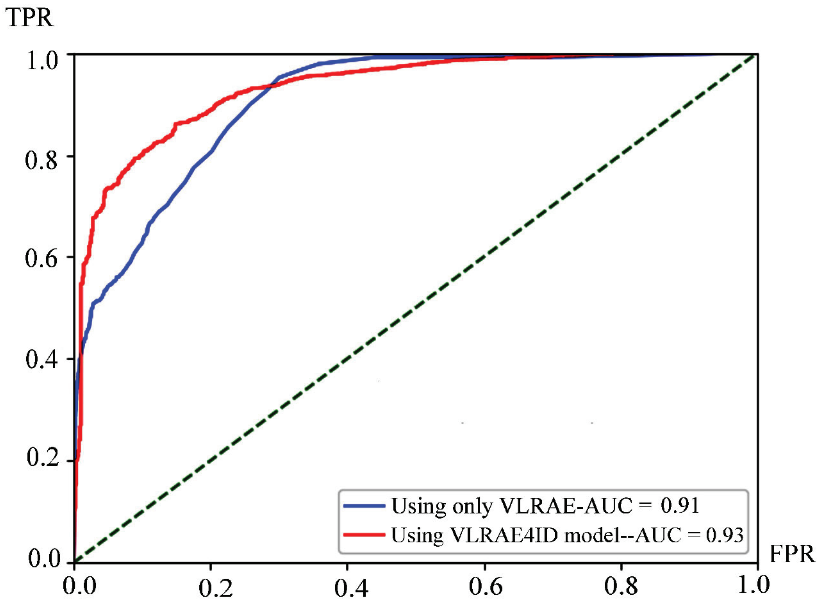

Figure 9. Compare ROC of VLRAE and VLRAE4ID model

5.2.3. VLRAE4ID with zero-day attack in misuse intrusion detection. Anomaly detection is a suitable model for outlier detection, when data is available for training can be taken as normal samples and the rest of the samples are not possible to define. Nowadays, attacks in information systems vary and can change instantly. Many of them are mimicry attacks, in which the exhibited behaviour is very similar to the normal process. For this type of attack, using only the features of normal behaviour to detect is hard. Therefore, intrusion detection systems are evaluated by the ability of known types of attacks and unknown (zero-day) attacks detection. Our model uses the features from unsupervised training data and static features, so perspectively may detect known and unknown attacks as well. In this experiment, we will evaluate VLRAE4ID misuse intrusion detection problem with the zero-day and known attacks in the ADFA-LD dataset.

For zero-day attacks, we create a new dataset for training and testing as follows. For each attack type data (AT) in attack dataset (AD)

$$\text{New\_Training\_Data} = \text{Training\_Data} + 0.7(\text{AD} - \text{AT}),$$
$$\text{ZeroDays\_Testing\_Data} = \text{Validation\_Data} + \text{AT}.$$

To evaluate the performance of known attack detection, the attack dataset was divided into 3 parts randomly, in which 2 parts for training and 1 part for testing. RandomForest was selected as the classifier in VLRAE4ID for misuse intrusion detection. Figure 10 shows the ROC of zero-day attack detection using VLRAE4ID for six types of attacks in ADFA-LD. The last ROC is the result of known attack detection. The AUC of all ROCs are larger than those of intrusion detection.

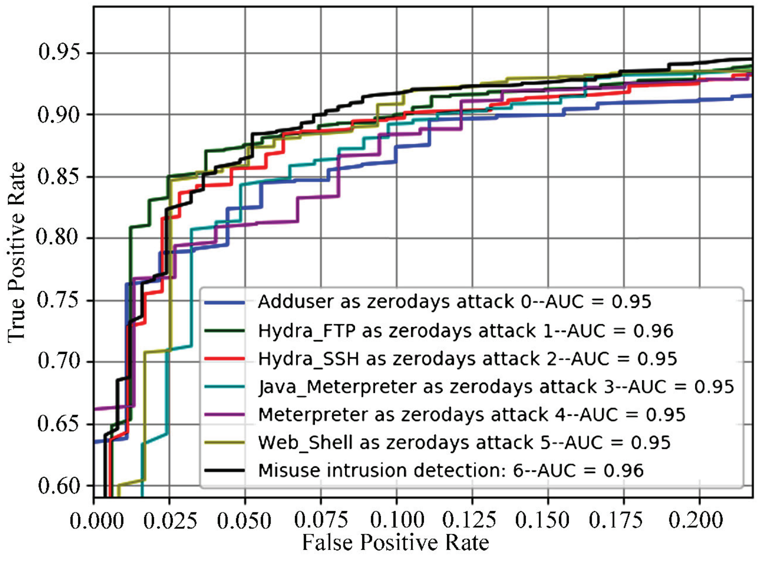

Figure 10. Zero-day attacks testing with VLRAE4ID

The FPR and TPR values in Table show that VLRAE4ID performs better in the detection of zero-day attacks if it has encountered data from other types of attacks.

Table. **Results of zero-day attack detection**

Zero-day attack	FPR	TPR
Adduser	11.0933	91.1111
Hydra-FTP	11.1619	93.7888
Hydra-SSH	10.9103	92.5714
Meterpreter	10.979	91.8699
Java-Meterpreter	11.1848	93.2432
Web-Shell	11.7566	92.3077

6. Discussion. Our evaluations show that our proposed method of using dynamic and static features based on variational LSTM-RNN networks has produced a promising set of results.

With regards to the anomaly intrusion detection, for the ADFA-LD dataset our VLRAE4ID method achieved an AUC value of 0.93 and a TPR of 90 %, with a 12.97 % FPR. This result is comparable with the ELM model [13] (TPR 90 % with FPR about 13 %), but as mentioned previously, our model doesn't suffer from the combinatorics explosion unlike the method in [13]. Moreover, our result is better than other methods such as sequence time-delay embedding (STIDE) (TPR 90 % with FPR 23 %), and the hidden Markov model [9] (90 % with FPR 43 % according to [13]), frequency-based technique [11] (78 % TPR with FPR 30 %) or one class SVM [12] (TPR 75 % with FPR 20 % in average). In [31], the authors used the trace abstraction techniques to reduce system call traces. After that the authors used STIDE with a window size of five and report the TPR equal to 100 % with FPR of 12.69 %; and used a HMM model to achieve the 100 % TPR with FPR of 60 % for ADFA-LD dataset.

Results of the zero-day attack detection experiment in Section 5 show that the features combined from dynamic and static features can be seen as hidden representation of system calls, which can be used to detect natural difference between attack and normal behaviour.

This experiment is more realistic, because real intrusion detection systems try to use as much as possible data to detect all type of attacks, including zero-day attacks. In the ADFA-LD dataset, VRLAE4ID can detect from 91 to 94 % attacks with a FPR of around 10 % for every type of zero-day attack. This is the first experiment of zero-day attack detection for ADFA-LD, so we can only compare with other anomaly detection methods. This result is significantly better than the zero-day attack detection performance of any models in [11–13]. Our results are better than misuse intrusion detection model in [32]. This model is 10-fold cross-validation supervised classification in which attack data was used to train the classifiers and without zero-day attacks in validation. The authors claim an AUC $= 0.93$ for the best tested method (kNN with $k = 3$), using an "enhanced" vector space model with n-grams ($n = 2, 3, 4, 5$).

7. Conclusion and future work. In this paper, we have discussed the problems faced by existing HIDS techniques. In response to this, we have proposed our novel VL-RAE4ID method for unsupervised holistic extraction and combination of dynamic and static features to enhance the IDS performance.

For the UNM STIDE dataset, our model achieved a perfect result: 100 % intrusion detection without any false alarm alerts.

For the ADFA-LD dataset, our results have demonstrated that our approach offers a high TPR with small FPR. Most notably, we have compared our VLRAE4ID model against the other notable anomaly intrusion detection methods. These comparisons demonstrate that our model offers a comparable result with the best model published so far [13], but our model does not require several weeks to build a semantic dictionary. Unlike most previous work, we have evaluated the capabilities of our model with misuse intrusion detection for zero-day attack detection, revealing a consistent level of zero-day attack detection accuracy.

Although our model has achieved the above promising results, we acknowledge that it is not perfect and there is further room for improvement.

In the future work, the first avenue of exploration will be to assess and extend the capability of our model to further improve zero-day attack detection accuracy and reduce the FPR. We will then look to expand upon our existing evaluations by utilising real-world backbone network traffic to demonstrate the merits of the extended model.

References

1. *The incident response analyst report.* Moscow, Kaspersky Publ., 2022, 20 p.

2. Hochreiter S., Schmidhuber J. Long short-term memory. *Neural Computation*, 1997, vol. 9, iss. 8, pp. 1735–1780. https://www.doi.org/10.1162/neco.1997.9.8.1735

3. Chandra R. Competition and collaboration in cooperative coevolution of Elman recurrent neural networks for time-series prediction. *IEEE Transactions on Neural Networks and Learning Systems*, 2015, vol. 26, no. 12, pp. 3123–3136. https://doi.org/10.1109/TNNLS.2015.2404823

4. Cho K., van Merrienboer B., Gulcehre C., Bougares F., Schwenk H., Bengio Y. Learning phrase representations using RNN encoder–decoder for statistical machine translation. *Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1724–1734. https://doi.org/10.3115/v1/D14-1179

5. Graves A., Liwicki M., Fernández S., Bertolami R., Bunke H., Schmidhuber J. A novel connectionist system for unconstrained handwriting recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2009, vol. 31, no. 5, pp. 855–868.

6. Deepika S., Erinc M., Ismini P., Johannes K., Sten H., Matthieu G., Andreas H. Human activity recognition using recurrent neural networks. *Proceedings of International Cross-Domain Conference for Machine Learning and Knowledge Extraction.* Reggio, Italy, 2017, pp. 267–274.

7. Fabius O., van Amersfoort J. R. Variational recurrent auto-encoders. *ArXiv preprint*, 2015, no. 1412.6581. https://arxiv.org/abs/1412.6581

8. Kingma D. P., Welling M. Auto-encoding variational Bayes. *Proceedings of 2nd International Conference on Learning Representations (ICLR)*, 2014, pp. 1–6.

9. Warrender C., Forrest S., Pearlmutter B. Detecting intrusions using system calls: alternative data models. *Proceedings of the 1999 IEEE Symposium on Security and Privacy.* Oakland, USA, 1999, pp. 133–145. https://doi.org/10.1109/SECPRI.1999.766910

10. Maggi F., Matteucci M., Zanero S. Detecting intrusions through system call sequence and argument analysis. *IEEE Transactions on Dependable and Secure Computing*, 2010, vol. 7, iss. 4, pp. 381–395. https://doi.org/10.1109/TDSC.2008.69

11. Xie M., Hu J., Yu X., Chang E. Evaluating host-based anomaly detection systems: application of the frequency-based algorithms to ADFA-LD. *Proceedings of 8th International Conference on Network and System Security.* Xian, China, 2014, pp. 542–549. https://doi.org/10.1007/978-3-319-11698-344

12. Xie M., Hu J., Slay J. Evaluating host-based anomaly detection systems: Application of the one-class SVM algorithm to ADFA-LD. *International Conference on Fuzzy Systems and Knowledge Discovery (FSKD).* Xiamen, China, 2014, pp. 978–982.

13. Creech G., Hu J. A semantic approach to host-based intrusion detection systems using contiguous and discontiguous system call patterns. *IEEE Transactions on Computers*, 2014, vol. 63, iss. 4, pp. 807–819. https://doi.org/10.1109/TC.2013.13

14. Ikram Y. S., Madkour M. A. I. Enhanced host-based intrusion detection using system call traces. *Journal of King Abdulaziz University (Computing and Information Technology Sciences)*, 2019, vol. 8, pp. 93–109.

15. Zhang Y., Luo S., Pan L., Zhang H. Syscall-BSEM: Behavioral semantics enhancement method of system call sequence for high accurate and robust host intrusion detection. *Future Generation Computer Systems*, 2021, vol. 125, pp. 112–126.

16. Osamor F., Wellman B. Deep learning-based hybrid model for efficient anomaly detection. *International Journal of Advanced Computer Science and Applications*, 2022, vol. 13, no. 4, pp. 975–979. https://doi.org/10.14569/IJACSA.2022.01304111

17. Anandapriya M., Lakshmanan B. Anomaly based host intrusion detection system using semantic based system call patterns. *IEEE 9th International Conference on Intelligent Systems and Control (ISCO).* Coimbatore, India, 2015, pp. 1–4. https://doi.org/10.1109/ISCO.2015.7282244

18. Lu Y., Teng S. Application of sequence embedding in host-based intrusion detection system. *IEEE 24th International Conference on Computer Supported Cooperative Work in Design (CSCWD).* Dalian, China, 2021, pp. 434–439.

19. Ouarda L., Bourenane M., Bouderah B. Towards a better similarity algorithm for host-based intrusion detection system. *Journal of Intelligent Systems*, 2023, vol. 32, no. 1, art. no. 20220259. https://doi.org/10.1515/jisys-2022-0259

20. Le T.-T.-H., Kim J., Kim H. An effective intrusion detection classifier using long short-term memory with gradient descent optimization. *Proceedings of the 2017 International Conference on Platform Technology and Service (PlatCon).* Busan, Korea, 2017, pp. 1–6. https://doi.org/10.1109/PlatCon.2017.7883684

21. Staudemeyer R. C., Omlin C. W. Evaluating performance of long short-term memory recurrent neural networks on intrusion detection data. *Proceedings of the South African Institute for Computer Scientists and Information Technologists Conference (SAICSIT 13).* New York, ACM, 2013, pp. 218–224. https://doi.org/10.1145/2513456.2513490

22. Staudemeyer R. C. Applying long short-term memory recurrent neural networks to intrusion detection. *South African Computer Journal*, 2015, vol. 56. https://doi.org/10.18489/sacj.v56i1.248

23. Bontemps L., Cao V. L., McDermott J., Le-Khac N. A. Collective anomaly detection based on long short-term memory recurrent neural networks. *Proceedings of 3rd International Conference on Future Data and Security Engineering*, Springer International Publishing, 2016, pp. 141–152. https://doi.org/10.1007/978-3-319-48057-29

24. Kim J., Kim J., Thu H. L. T., Kim H. Long short term memory recurrent neural network classifier for intrusion detection. *International Conference on Platform Technology and Service (PlatCon)*, 2016, pp. 1–5. https://doi.org/10.1109/PlatCon.2016.7456805

25. Blei D. M., Kucukelbir A., McAuliffe J. D. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 2017, vol. 112, iss. 518, pp. 859–877. https://doi.org/10.1080/01621459.2017.1285773

26. Liu T. F., Ting K. M., Zhou Z.-H. Isolation forest. *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining.* Pisa, Italy, 2008, pp. 413–422.

27. *University of New Mexico (UNM) dataset for intrusion detection.* Available at: https://www.cs.unm.edu/ immsec/data-sets.htm (accessed: August 15, 2022).

28. *The ADFA Intrusion Detection Datasets.* Available at: https://research.unsw.edu.au/projects/adfa-ids-datasets (accessed: September 10, 2023).

29. Yeung D. Y., Ding Y. Host-based intrusion detection using dynamic and static behavioral models. *Pattern Recognition*, 2003, vol. 36, no. 1, pp. 229–243.

30. Wang W., Guan X. H., Zhang X. L. Modeling program behaviors by hidden Markov models for intrusion detection. *Proceedings of 2004 International Conference on Machine Learning and Cybernetics (IEEE Cat. no. 04EX826)*. Shanghai, China, 2004, vol. 5, pp. 2830–2835.

31. Murtaza S. S., Khreich W., Hamou-Lhadj A., Gagnon S. A trace abstraction approach for host-based anomaly detection. *2015 IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA)*. Verona, 2015, pp. 1–8.

32. Borisaniya B., Patel D. Evaluation of modified vector space representation using ADFA-LD and ADFA-WD datasets. *Journal of Information Security*, 2015, vol. 6, no. 3, pp. 250–264.

Received: October 1, 2023.
Accepted: December 26, 2023.

Authors' information:

Viet Hung Nguyen — PhD in Computer Technologies; hungnv@lqdtu.edu.vn

Nguyen Ngoc Tran — PhD in Computer Technologies, Associate Professor; ngoctn@lqdtu.edu.vn

Объединение преимуществ динамического и статического обнаружения вторжений с использованием вариационного рекуррентного автокодировщика с долгой краткосрочной памятью

В. Х. Нгуен, Н. Н. Чан

Вьетнамский государственный технический университет имени Ле Куй Дона,
Вьетнам, 140000, Ханой, ул. Хоанг Куок Вьет, 236

Для цитирования: *Nguyen V. H., Tran N. N.* Combining dynamic and static host intrusion detection features using variational long short-term memory recurrent autoencoder // Вестник Санкт-Петербургского университета. Прикладная математика. Информатика. Процессы управления. 2024. Т. 20. Вып. 1. С. 34–51. https://doi.org/10.21638/11701/spbu10.2024.104

Несмотря на многие преимущества систем обнаружения вторжения на хосте (HIDS), они в основном не приняты в мейнстримных стратегиях кибербезопасности. В отличие от мониторящих сетевой трафик систем обнаружения вторжения HIDS являются последним рубежом обороны операционной системы от потенциальной атаки. Одна из основных причин такого состояния дел — низкая эффективность адекватной защиты от атак нулевого дня. Поскольку число атак уязвимостей нулевого дня и связанных с ними атак растет, для современной HIDS становится критически необходимым уметь им противостоять. В настоящей работе рассмотрен подход с использованием вариационного рекуррентного автокодировщика с долгой краткосрочной памятью (variational long short-term memory — recurrent autoencoder), который увеличивает эффективность обнаружения атак нулевого дня. Разработанная модель реализована с использованием TensorFlow, и проверена ее работа на известных наборах данных ADFA-LD и UNM. Также проведено сравнение с другими подходами, опубликованными в наиболее известных статьях по данной тематике.

Ключевые слова: HIDS, обнаружение аномалий, вариационный автокодировщик, глубокое обучение.

Контактная информация:

Нгуен Вьет Хунг — канд. компьют. наук; hungnv@lqdtu.edu.vn

Чан Нгуен Нгок — канд. компьют. наук, доц.; ngoctn@lqdtu.edu.vn