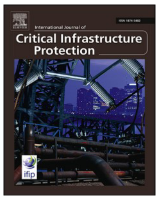
Contents lists available at ScienceDirect

International Journal of Critical Infrastructure Protection

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



A real-time network based anomaly detection in industrial control systems

Faeze Zare[a], Payam Mahmoudi-Nasr[a,*], Rohollah Yousefpour[b]

[a] *Computer engineering Department, University of Mazandaran, Babolsar, Iran*
[b] *Computer Science Department, University of Mazandaran, Babolsar, Iran*

A R T I C L E I N F O

Keywords:
Anomaly detection
Auto encoder
Manipulation attack
SCADA

A B S T R A C T

Data manipulation attacks targeting network traffic of SCADA systems may compromise the reliability of an Industrial Control system (ICS). This can mislead the control center about the real-time operating conditions of the ICS and can alter commands sent to the field equipment. Deep Learning techniques appear as a suitable solution for detecting such complicated attacks. This paper proposes a Network based Anomaly Detection System (NADS) to detect data manipulation attacks with a focus on Modbus/TCP-based SCADA systems. The proposed NADS is a sequence to sequence auto encoder which uses the long short term memory units with embedding layer, teacher forcing technique and attention mechanism. The model has been trained and tested using the SWaT dataset, which corresponds to a scaled-down water treatment plant. The model detected 23 of 36 attacks and outperformed two other existing NADS with an improvement of 0.22 for simple attacks and obtained a recall value of 0.86 on attack 36 compared to the other NADS which obtained 0.74.

1. Introduction

Supervisory control and data acquisition (SCADA) system is widely used in different industrial control systems (ICSs), such as power systems, water treatment, and oil purification [1]. As is shown in Table 1, according to the recent ICS-CERT report [2], the cyber-attacks in various ICSs have increased in H2 compared with H1 in 2020, and in 2020 they have increased compared with 2019.

The SCADA system relies heavily on accurate measurement values from field devices and issued commands from the control center. Meanwhile, the data sent over the network may be manipulated by the attacker. In a manipulation attack, which is a subset of Man-In-The-Middle (MITM) attacks, the attacker makes subtle modifications to the content of packets to mislead the controlled ICS to sabotage [3]. When the measurement values are manipulated, unwanted control commands may be issued, which may threaten the stability of the controlled grid [4]. Therefore, manipulation attack is one of the most dangerous threats and dealing with it is very important.

The most useful ICS protocol with low security is Modbus [5], so the constant request/response control commands loops in ICSs and typical lack of authentication make the manipulation attacks interesting in SCADA systems.

The complexity of manipulation attacks on networks, the increasing occurrence of them, and the increasing performance of deep neural networks (DNNs) [6] have motivated the development of intrusion detection systems for ICSs based on deep learning techniques. This paper proposes a real-time network-based anomaly detection system (ADS) based on DNNs for ICS. An anomaly is defined as a data manipulation attack in Modbus/TCP traffic on a SCADA system. The proposed model relies on sequence to sequence Auto Encoder (AE) using Long Short Term Memory (LSTM) units to memory and popularize the characteristics of the ICS network traffic. The sequence to sequence AE learns the pattern of network traffic/packet content and generates a normal sequence to detect anomaly behavior based on a predefined threshold. The threshold value is determined based on a precision-recall curve on the validation data set. The proposed ADS uses four techniques to reduce false alarm rates as follows: (i) separation of categorical/numerical input data into statics and dynamics (ii) the embedding layer to encode categorical features based on the relation between different values of features (iii) the teacher-forcing mechanism to prevent deviation and consequence faster convergence using original inputs from a prior time step as Decoder inputs (iv) the attention mechanism to make the model more efficient at each time step.

The proposed model has been trained and tested on the SWaT dataset, which is a scaled down water treatment plant, and among various SCADA datasets is the most popular one [7]. The SWaT dataset

* Corresponding author at: Computer Engineering Faculty of Engineering & Technology, University of Mazandaran (UMZ) Pasdaran Street, P.O. Box: 47415-416, Babolsar, Iran.
E-mail address: P.mahmoudi@umz.ac.ir (P. Mahmoudi-Nasr).

https://doi.org/10.1016/j.ijcip.2024.100676
Received 15 April 2023; Received in revised form 9 December 2023; Accepted 19 April 2024
Available online 26 April 2024

contains 12 days of data which has been transferred between PLCs and sensors. It includes a training set with normal packets, and a test set containing normal and anomalous records. The anomalies consist manipulated measured values targeting the integrity and the availability of the controlled system, which it is similar to the attack that happened in power system of Ukraine in 2016 [8] and led to power outages for more than 8 h.

Considering the above context, the key contributions to this paper are as follows:

1) Propose a real-time anomaly detection system in ICSs with Modbus/TCP network traffic.
2) Propose a daily network packet analyzer for the proposed ADS using sequence to sequence AE instead of evaluating attacks separately.
3) Improve zero-day attack detection using AE, which is not limited to several attacks, unlike classification methods.
4) Improving learning normal network patterns, convergence, and generalization using embedding layers, teacher forcing, attention, and separation of static and dynamic data.

The remaining paper is organized as follows: section 2 and 3 provide some related works and explanations of the SWaT dataset, respectively. Section 4 introduces the threat model and section 5 describes the proposed model. Section 6 shows the experimental results. Section 7 compares the proposed model with other available models.

2. Related works

Due to the growth of cyber attacks on ICSs and the impact of these systems on society and humans, many studies have been conducted on improving the security of ICSs. This section investigates recent studies on anomaly detection in ICSs. In particular, studies that have focused on water treatment plants and used the SWaT dataset for validation have been considered. Because their performance, as state-of-the-art, has been compared. Their comparision summary is available in Table 2.

Narayanan et al. [9] proposed a behavior-based anomaly detection method based on neural networks and validated it using the SWaT dataset. Although the model worked well, like other works by Alsaedi et al. [10] and Kravchik et al. [11], they have one main shortcoming: they do not consider the real-time issues of SCADA. Since they all worked with historical data, their models cannot be applied to inflight packets and function as a real-time model IDS.

According to [12] a design-driven invariants approach is a solution for industrial attack detection. They implement a series of invariants based on system design documents and use them to detect anomalies. On the other hand, Maiti et al. [13] have shown that the design-based invariants approach is not suitable for detecting processing anomalies. So, they have proposed a solution based on both design and data-driven. Among the most important shortcomings of such techniques are human intervention and avoidance of intelligent methods. Also, they cannot act as real-time IDS due to the use of historical data.

Somu et al. [14] proposed an ADS using a DNN from 1 to 3 layers and the cumulative sum technique. The model has been compared with 5 algorithms: Support Vector Machine (SVM), K Nearest Neighbors, Principle Component Analysis, AE, and Generative Adversarial Network (GAN), and has the lower false alarm rate and best performance. The limitations of the work are the use of historical data of the SWaT dataset which is logged per second and just has the sensor's values. Another overcome is that they have worked only on one stage of SWaT.

Shalyga et al. [15] have implemented 3 Neural Network (NN) models (Recurrent Neural Network (RNN), Convolutional Neural Network and Multi Layers Perceptron (MLP)) using Exponential Weighted Moving Average (EWMA) and mean p-powered error measure techniques to detect all 36 available attacks on historical data in the SWaT dataset . Among these 3 NNs, the MLP has the best performance with 0.69 recall with a sequence length of 200 s. The shortages of this work are (i) using a multi layer perception which produces more training parameters than other NNs and is the reason for a heavy and delayed model. (ii) the length of sequence, which takes 200 s and that's a long time for vital SCADA systems. (iii) using only the sensors values to train a model which prevents to detect other possible attack types. Despite these flaws, it couldn't get a satisfying recall.

AEs is an unsupervised NN which can models normal behaviour of the system in order to detect anomalies according to the difference between expected normal behaviour and live system's behaviour. Kwon et al. [16] presented a composite AE model to detect attacks in historical data on the SWaT dataset with a sequence length of 120 s and using EWMA and p-power techniques. In the first step, they use a signature based IDS to detect anomalies and in the second step, a behavior based IDS for more accuracy. The best recall value of this experiment is 0.82. The shortcomings of the work are also using only historical data and a long sequence length in a real-time SCADA system.

To better detect the anomalies in a system, some models give the praph of system as input and tries to learn it and detect the anomalies. Lin et al. [17] proposed a graphical-based anomaly detection model to detect SWaT attacks (history data), which divides the data into sub-models according to the system functionality and a Bayesian network will learn the relations and have got 0.78 recall. Deficiency of this model in addition to shortcomings of using the only history data, is that this model perfectly match the SWaT routine behaviors, so for a little change in system, it needs some ones to prepare a new graph and deviation between sensors and actuators, also since human mistakes always may happen so it's better to eliminate the human interventions.

Across RNNs, LSTM units with more hidden states can better remember some prior states and make decisions according to the past and present. Mieden et al. [18] implemented an ADS using two different classification models based on DNN and LSTM to detect network attacks in the SWaT dataset. The LSTM model has been tested on different types of attacks, including SSSP, MSMP, and MSSP with recall rates of 0.41, 0.58, and 0.44 respectively. The limitations of the work are: the model would only be trained for just the performed attacks, the model didn't test on SSMP attacks, and the obtained recall values are not satisfying.

Using privileged information (PI) in model training leads to faster convergence and obtaining more information. As this information doesn't exist in the test set, it may lead to overfitting. Pordelkhaki et al. [19] proposed a Network based IDS (NIDS) to detect attacks in SWaT using SVM with leverage using privileged information. It has used historical data as PI and network data as the main input. The proposed

Table 1
Infected ICS computers and ICS attack growth.

Increasing Percentage of Infected ICS computers				ICS attacks growth Percentage			
ICS title	2019	Hp1 2020	H2 2020	Attack type	2019	H1 2020	H2 2020
Building automation	38.0	39.9	46.7	Blacklist internet resources	12.5	11.0	13.5
Oil & Gas	36.3	37.8	44.0	Malicious scripts and redirects on web resources	7.8	6.5	8.1
ICS engineering & integration	32.7	31.5	39.3	Spy trojans, backdoors and keyloggers	5.8	5.6	7.0
Energy	36.6	31.7	36.9	Malicious documents	2.9	2.2	3.4
Automotive Manufacturing	37.6	30.2	35.0	Miners in the form of executable files for windows	1.1	0.9	1.6

Table 2
Summary of related works.

	year	Method	Dataset	Limitations	Advantages
[9]	2020	NNs	SWaT (history data)	Limited to the sensor values and value-based attacks, Inability to work in a real-time system	High detection rate Very Low false positive rate
[10]	2022	AE	SWaT (history data)	Limited to the sensor values and value-based attacks, Inability to work in a real-time system	Trying on various datasets F1 score of 0.96 on SWaT
[11]	2021	1D CNN, shallow UAEs, VAE and PCA	SWaT (history data)	Limited to the sensor values and value-based attacks, Inability to work in a real-time system	Testing different methods on various datasets
[12]	2018	Design-driven invariants	SWaT (history data)	a lot of human intervention, avoidance of smarting methods, cannot work as real-time IDS	Design-driven invariants start working at the beginning of operations
[14]	2020	DNN + CUSUM	SWaT (history data)	Limited to the sensor values and value-based attacks, evaluation is done on only 6 attacks (SSSP)	0 false alarm rate Suitable for SSSP attacks
[15]	2018	MLP + EWMA + mean p-powered error measure	SWaT (history data)	Limited to the sensor values and value-based attacks, sequence length is 200 s which is too long for SCADA and the MLP is a heavy NN model.	Recall of 0.69 and 24 attacks detected of 36 available attacks
[16]	2022	CAE + EWMA + mean p-powered error measure	SWaT (history data)	Limited to the sensor values and value-based attacks, sequence length of 120 s is too long for SCADA system	Use of signature base IDS makes some of detections fast, recall of 0.82
[17]	2018	graphical-based anomaly detection	SWaT (history data)	Limited to the sensor values and value-based attacks, based on the system details makes it special for SWaT system, need a graph of system work	perfectly matchs the SWaT routine behaviors and recall of 0.78
[18]	2020	LSTM network for classification problem	SWaT (network data)	Evaluation is done according to the types of attacks and the evaluation on SSMP isn't declared, the results of recalls are not good enough, since it's a classification problem it'll be fitted to the available attacks	Due to the use of network packets can be a real time IDS
[19]	2021	SVM + LUPI	SWaT (network data & history data)	The evaluation is done on only attack #36	Recall of 0.74 for attack #36, use of benefits of both network and history data
[20]	2019	AE	SWaT (history data)	Limited to the sensor values and value-based attacks, sequence length is 100 s which is too long for SCADA, training 6 models for each process stage	Detection of 29 attacks of 36 attacks
[21]	2020	CAE + EWMA	SWaT (history data)	Limited to the sensor values and value-based attacks, sequence length is 120 s which is too long for SCADA	Detection of 26 attacks of 36 attacks

model has been tested only on attack 36 and has got 0.74 as the average recall. The shortcomings of the model are: training the model with different data, and testing it only on one of 36 available attacks.

In order to increase the detection rate, according to the structure of the system, anomaly detection can be done in each stage separately. Kim et al. [20] have implemented a sequence to sequence ADS to detect attacks in historical data of the SWaT dataset. The model trained and tested for each of 6 processes separately with a sequence length of 100 s. The limitations of the model are: inability to detect 4, 13, 14,19, 24, 25 and 29 attacks, using only historical data, using 6 models as host base IDS, and a long sequence length in a real-time SCADA system.

Addition of some techniques, such as EWMA, can help to improve NNs. Wang et al. [21] have proposed a composite AE to detect anomalies in historical data of the SWaT. The model has two decoders, one to predict the next sequence and one to reconstruct the input. They have used EWMA techniques to calculate the loss value. The length of sequences was 120 s and the model couldn't detect the attacks 4, 10, 11, 12, 14, 16, 20, 22, 24 and 26. The limitations of the model are: using only historical data, and a long sequence length (2 min) in a real-time SCADA system.

According to our study, most of the research is limited to historical data in the SWaT dataset, and just a few of them focuse on real-time network packets.

3. Dataset

The used dataset in this work is a Secure Water Treatment (SWaT) system, which is a scaled down water treatment plant proposed by Singapore University of Technology and Design [22]. Fig. 1 shows the SWaT architecture, which is composed of different levels. Level 0 refers to physical processes and consists of 6 stages. Each stage works on a separate part of purification and is comprised of several sensors and actuators. These stages are related together with using Programmable Logic Controllers (PLC), making level 1. The PLCs send the controlling commands to the actuators according to the received data from sensors. They also transfer data using a central switch to the Human Machine

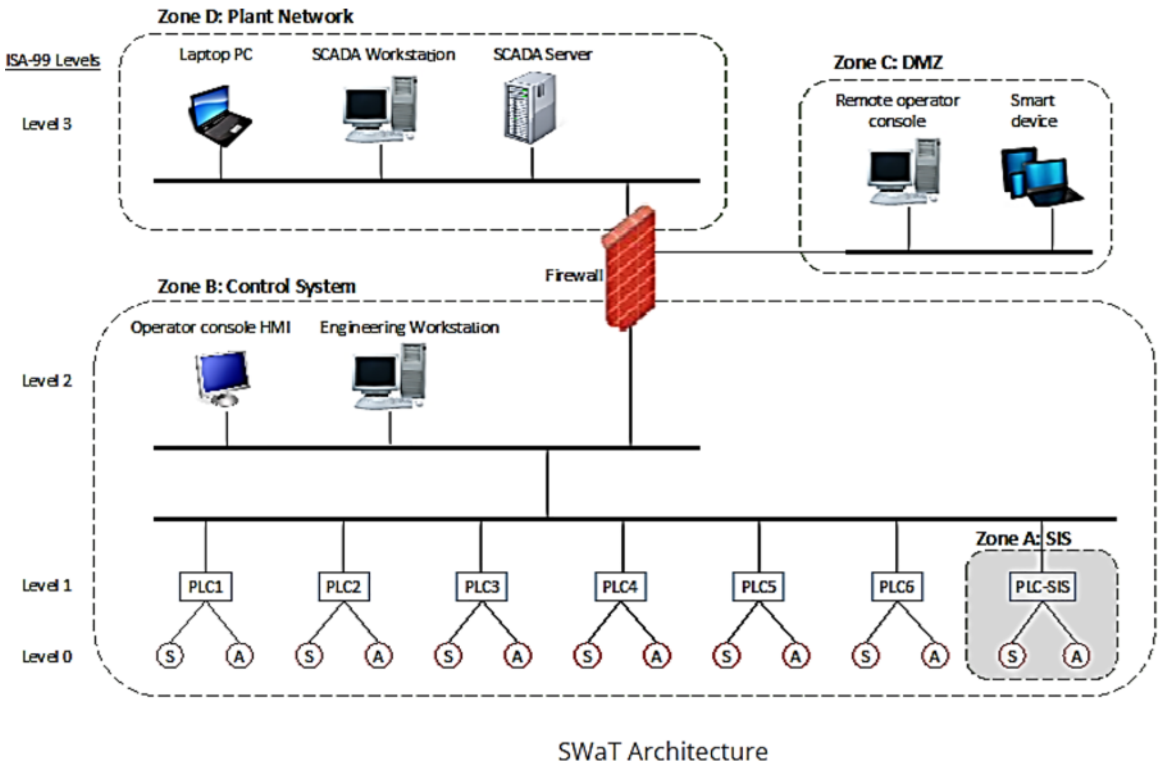


Fig. 1. SWaT levels (https://itrust.sutd.edu.sg/itrust-labs-home/itrust-labs_swat/).

Interface (HMI), Engineering workstation and SCADA server.

The most significant parts of transferring packets, which are Modbus TCP/IP, are the sensor's values, actuators states, PLCs' IP,and devices ID. The dataset is the collected packets for 12 days (22 Dec till 2 Jan), which is 7 days (22 dec till 28 Dec at 10:29:13) under normal operation and 6 days (28 Dec 10:29:14 till 2 Jan) under attack. It consists of historical data of logged values from the sensors and in-flight network data. The network data has 18 features. As Table 3 shows, some of them are numerical, which change during time, and some of them are categorical, which change among a few different predefined values. The available attacks in the SWaT dataset are all MITM and, as Table 4 shows, the attacks are comprised of 4 types, including SSSP, SSMP, MSSP and MSMP.

Table 3
Network packet features.

	Feature name	Type	Data Type	Description
1	Date	TS	Time	Day of packet logging
2	Time	TS	Time	Time of packet logging
3	Origin	CT	String	Server IP
4	Type	CT	String	Type of log
5	Interface Name	CT	String	Name of interface
6	Interface Direction	CT	String	Direction of interface
7	Source IP	CT	String	IP of source
8	Destination IP	CT	String	IP of destination
9	Protocol	CT	String	Network protocol
10	Proxy Source IP	CT	String	IP of proxy source
11	Application Name	CT	String	Name of application
12	Modbus Function Code	CT	Number	Modbus function code
13	Modbus Function Description	CT	String	Description of function code
14	Modbus Transaction ID	NUM	Number	Each packet Transaction ID
15	SCADA Tag	CT	String	Sensor or actuator ID
16	Modbus Value	NUM	Number	Transferring value
17	Service/Destination Port	CT	Number	Destination port number
18	Source Port	CT	Number	Source port number

(CT as categorical, TS as timestamp, NUM as numerical).

Table 4
SWaT attack types.

Attack type	Description
SSSP	The target is just a single point and the attack is also performing among single stage.
SSMP	The target is just a single point but the attack is performing among multi stages.
MSSP	This attack has multi targets and is performing among single stage.
MSMP	This attack has multi targets and is performing among multi stages.

4. Threat model

We assume the attacker's purpose is to damage or modify the SCADA operations to take the ICS out of service. We consider that the attacker, through a MITM attack, has access to the Modbus traffic packets using industrial network tools and intercepts them. As a result, the attacker can eavesdrop and manipulate all (i) measurement values between sensors and PLCs to make the PLC take wrong decisions, and (ii) control messages between PLCs and actuators to send wrong commands to actuators.

The sent data values are stored in the monitoring section at level 3, so detecting the manipulation attacks can be easily done by monitoring the stored measurements, and the changes outside the normal range are known as anomalies. But at the lower level which the raw data is being sent as packets, it is more complicated to detect these attacks. In this work, an attempt is made to provide a model based on deep learning so that attacks can be detected in real-time and at a low level.

5. Proposed ADS

Many efforts have been made to improve this model, the proposed model is an output of various reviews on the different ML methods and

SWaT dataset. In this regard, many structures and techniques, including CNN networks and data classification methods, were reviewed and rejected, and finally, the best-selected method to learn the behavior of a network, based on the flow of packets is to provide a sequence-to-sequence autoencoder. moreover, due to the large volume of SWaT network data (about 100 GB), any changes and checks require a long time.

The general structure of the proposed ADS is shown in Fig. 2. The proposed Model is trained using normal data after feature engineering. Then the trained model is tested with the rest of the data. The ADS is based on an AE which is constructed using LSTM units. The AE reconstructs the input sequence according to the preset parameters based on a normal pattern. The Mean Square Error (MSE) between input traffic and output shows the anomaly if it's upper than the predefined threshold. A threshold value is determined by the evaluation data and is updated using the precision-recall curve daily. The training and testing algorithms of ADS are explained in Figs. 3 and 4 respectively, the symbol "k" refers to the sequence length.

5.1. *Feature engineering*

The packet features are shown in Table 3 and according to Fig. 5 the following steps are carried out for feature engineering:

5.1.1. *Feature reduction*

Step1: Since the captured data in the dataset is time-dependent, it needs to create sequences in order by date and time features. The received network data consists of 18 features. The two first features (date and time) are used for making time series sequences, and after that, they could be removed. The sequence length is a hyperparameter and depends on the number of necessary packets to predict the system state.

Step2: the Modbus packets are a request/response type. Since the request packets don't have useful information for ADS, they can be removed from the dataset. In this way, the Transaction ID field, which is used to relate a request packet to the corresponding response packet, has no more informative data, and it can also be removed from the dataset.

Algorithm 1 Training algorithm

Static Input Sequence: $\left\{ X_{st}; t\,|_0^k \right\}$

Dynamic Input Sequence: $\left\{ X_{Dt}; t\,|_0^k \right\}$

Output: a list of MSEs

```
1.  EP ← number of epochs
2.  learning rate=0.001
3.  for epoch in EP do
4.     for each batch do
5.        for each input sequence do
6.           Encoder initial state c = FC2 (SEM ( $X_{st}$ ))
7.           Encoder initial state h = FC3 (SEM (( $X_{st}$ ))
8.           Numerical, categorical ← $X_{Dt}$
9.           DEM ← categorical, FC1 ← Numerical
10.          FC4 ← (FC1, DEM)
11.          Encoder output sequence ← Encoder (FC4)
12.          ATN (t) ←Attention layer (Encoder output sequence)
13.          TF (t) ←Teacher forcing (FC4)
14.          Decoder input ← concatenate (ATN, TF)
15.          Output sequence ← Distributed dense layer (Decoder output)
16.       Cost = MSE ( $FC4_t\,|_0^k$, output sequence $_t|_0^k$ )
17.       end for
18.       ADAM ← minimize the Cost
19.       New weight = weight – learning rate * gradient
20.    end for
21.    learning rate = learning rate * $e^{-0.1}$
22. end for
```

Fig. 3. Training proposed ADS.

5.1.2. *Preprocessing*

Step3: The string type features should be mapped to integer

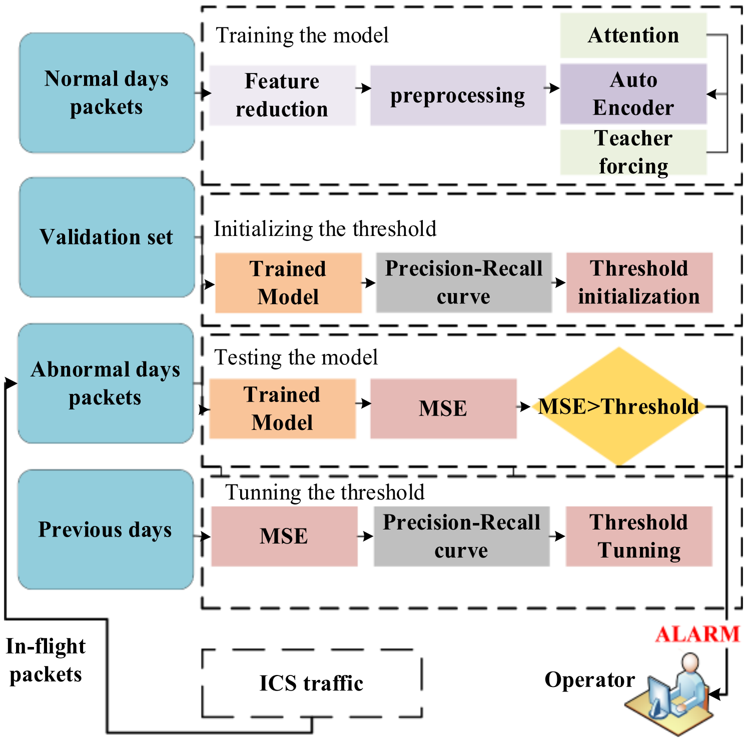


Fig. 2. The proposed ADS structure.

Algorithm 2 Testing algorithm

```
Static Input Sequence: ${X_{st}; t |_0^k}$
Dynamic Input Sequence: ${X_{Dt}; t |_0^k}$
Threshold ← predefined value of the previous days
1.  for each input sequence do
2.      Encoder initial state c = FC2 (SEM ( $X_{st}$ ))
3.      Encoder initial state h = FC3 (SEM (( $X_{st}$ ))
4.      Numerical, categorical ← $X_{Dt}$
5.      DEM ← categorical, FC1 ← Numerical
6.      FC4 ← (FC1, DEM)
7.      Encoder output sequence ← Encoder (FC4)
8.      ATN (t) ←Attention layer (Encoder output sequence)
9.      TF (t) ←Teacher forcing (FC4)
10.     Decoder input ← concatenate (ATN, TF)
11.     Output sequence ← Distributed dense layer (Decoder output)
12.     MSEs ← MSE ( $FC4_t |_0^k$, output sequence $_t |_0^k$ )
13.     if MSE > Threshold:
14.         Create an ALARM
15.     else:
16.         Continue
17. end for
```

Fig. 4. Testing proposed ADS.

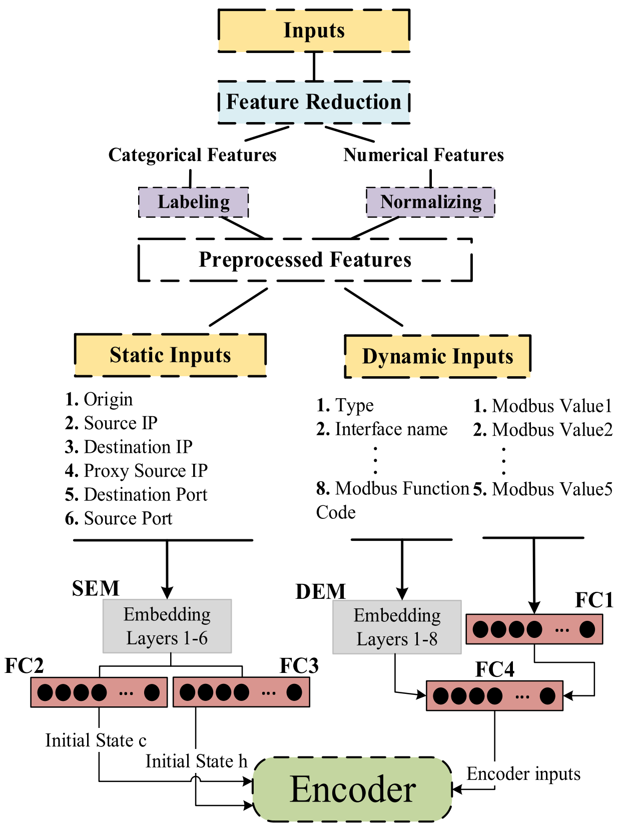


Fig. 5. Feature engineering process.

numbers, which is called labeling, in order to be used for calculating in deep learning processing.

Step4: By using one hot or embedding layer technique, a categorical data type should be mapped to a vector with n number of dimensions. In the proposed model, an embedding layer with a value between [−1, +1] is used for each categorical data.

Step5: The Modbus Value filed contains hexadecimal values of measured/state data of sensors/ actuators. Some packets contain 8 values and others contain more. Some researches e.g. [19] has used only one last value for training the anomaly detection model. But according to our experiment, the best performance will get using 5 last Modbus values.

After transforming the hex values to decimal values, the proposed model has been trained on different numbers of Modbus values and has been tested on all of the transferred packets on 30 Dec. December 30 is chosen because it contains three types of attacks (MSMP, MSSP, SSMP) and it is suitable for evaluation of a NIDS. The MSMP and MSSP attacks perform from different stages and SSMP targets different points. Figs. 6-9 show the AUC value for different numbers of Modbus values that have been used to train and test the proposed model. As it can be seen, the trained model with 5 last Modbus values has the best AUC value (The second and third Modbus values in the packet are all zeros, so they have not participated in the training.). Also, Fig. 10 shows the model overfitting for use of more than 5 Modbus values.

As it can be seen in Fig. 5, after normalizing the 5 Modbus values, FC1 has been used to map them to n dimensions and also find the best relationship among them.

Step 6: The last 5 Modbus values should be normalized$\in [0, +1]$. So, the minimum and maximum transferred values during 7 normal days have been selected and all of them are normalized according to the following formula.

$$x_{scaled} = (x - x_{\min})/(x_{\max} - x_{\min}) \tag{1}$$

Step 7: According to the existing attacks in the SWaT dataset, which are MITM, the attacker wouldn't change the IP address and port numbers. Therefore, to improve the model performance, origin, source/destination IP address, proxy source IP, and source/destination port number, have been considered static, and the other features as dynamic inputs.The comparison of Figs. 8 and 9 shows that this type of data separation could lead to a better AUC.

As the static input values are composed only of categorical features, in the proposed model, in order to limit them to a range of numbers and figure out the available relationship, they have passed through the Static Embedding Layer (SEM). To reduce the output dimensions of SEM and concatenate them, FC2 and FC3 have been used with the same number of neurons to prepare the LSTM initial states of AE.

The dynamic inputs, which include both numerical and categorical data types, require different preprocessing. To map the categorical data into a limited range of values, the Dynamic Embedding layer (DEM) has been used that provides an embedded matrix of float numbers which also finds out the amount of relation between them. For the numerical features to extract new and more informative data, the FC1 has been employed, which maps them to a float array. To concatenate the outputs of FC1 and the DEM, FC4 has been applied, which also reduces the dimensions according to the less number of neurons. It helps the AE to make the decision faster.

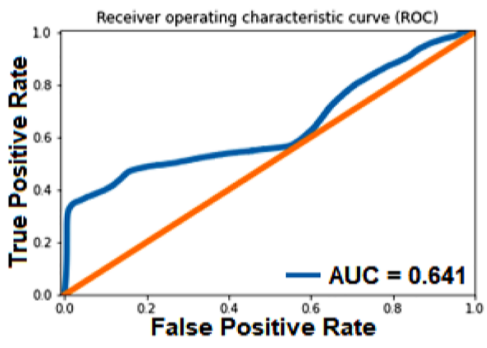


Fig. 6. ROC curve for one last value.

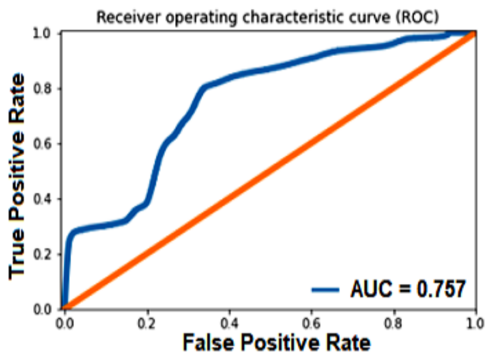


Fig. 7. ROC curve for 4 last values.

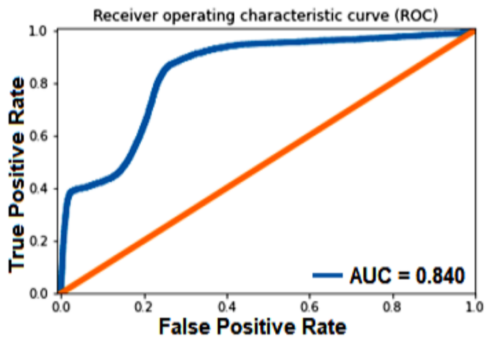


Fig. 8. ROC curve for 5 last values.

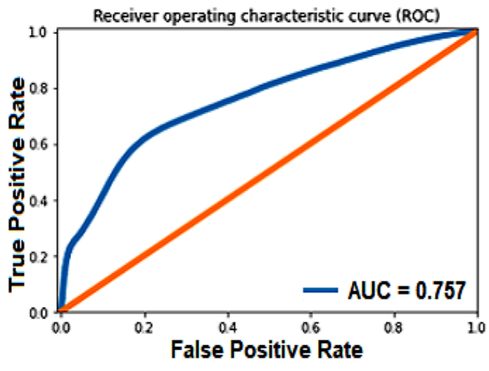


Fig. 9. ROC curve for 5 last values, without separation.

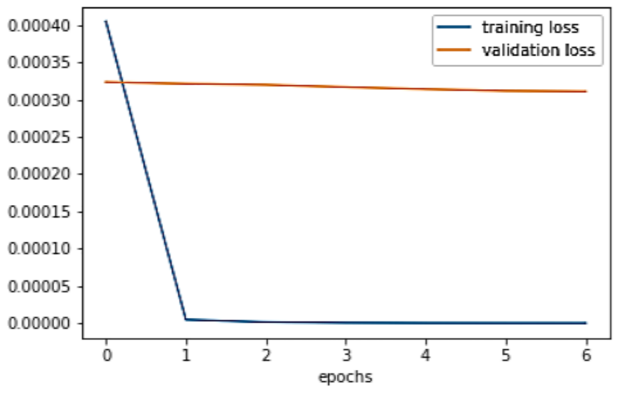


Fig. 10. Overfitting the model training using more than 5 last Modbus values.

5.2. Anomaly detection module

Fig. 2 shows the architecture of the proposed anomaly detection module (ADM). As is shown, the proposed ADM is based on the AE neural network. Since the SWaT is an extremely imbalanced dataset, a sequence to sequence AE is provided to detect abnormal traffic behaviour. Both the encoder and decoder are made of LSTM units where the number of them (time step) is a hyper parameter. That is because the SWaT dataset consists of time series data and each state depends on the (i) previous one, and (ii) the long term of sequences. According to formulas (2)-7, each LSTM unit contains three gates (update (u_t), forget (f_t) and output (o_t)), and two hidden states (h_t, C_t). The initial values of the hidden states (h_0, C_0) of the encoder module are determined by the dense layers' outputs (FC2 & FC3) for static data, and the input values of the LSTM units ($x_1..x_t$) of the encoder are obtained using the preproccesing dense layers's output (FC4) for dynamic data in feature engineering unit.

All encoded data is passed as initial values to the hidden states of the decoder (c_0,h_0). The input values of the decoder module are concatenated vectors from the output of the attention layer and teacher forcing technique. The more details about these two techniques are described in the next section. FC5 is used to match the output dimension of the decoder with the input of the encoder modules.

$$f_t = \sigma\left(W_f[h_{t-1}, x_t] + b_f\right) \tag{2}$$

$$u_t = \sigma(W_u.[h_{t-1}, x_t] + b_u) \tag{3}$$

$$o_t = \sigma(W_o.[h_{t-1}, x_t] + b_o) \tag{4}$$

$$C_t = f_t * C_{t-1} + u_t * \widetilde{C_t} \tag{5}$$

$$\widetilde{C} = \tanh(W_C.[h_{t-1}, x_t] + b_C) \tag{6}$$

$$h_t = o_t * \tanh(C_t) \tag{7}$$

5.3. Detection logic

The AE is trained using normal network traffic and sets its parameters to reconstruct normal data sequences in the output of a decoder. Whereas it worsens their reconstruction capability when abnormal sequence inputs are given. So, normal and abnormal sequence inputs are distinguished by selecting an appropriate threshold based on the reconstruction errors obtained on a validation set. The reconstruction error is calculated by the formula (8).

$$MSE = 1/N\left(\sum_{i=1}^{N}(Y_i - \widehat{Y}_i)^2\right) \tag{8}$$

In order for the precision and recall values to be adjusted to a balanced level, the threshold tunning is done using the intersection point of the precision-recall curve according to MSEs of previous days. The proposed model could detect an anomaly if the MSE value is more than the fixed threshold in real-time and an alarm will be sent to the administrator. As an example, threshold detection process for 31 Dec is shown in Fig. 12.

5.4. Techniques to get a better performance

In the proposed model to make a better AE performance, the Teacher Forcing technique [23] has been used. Using this technique, ground truth data is given to the decoder for fast convergence. Therefore, in Fig. 11, the main inputs are also applied as the decoder inputs, and prior time steps' output are not used.

The other mechanism which has been used is Bahdanau Attention [24]. The benefit of attention techniques, in addition to faster convergence, is to lead the model in a timestep to focus and pay more attention

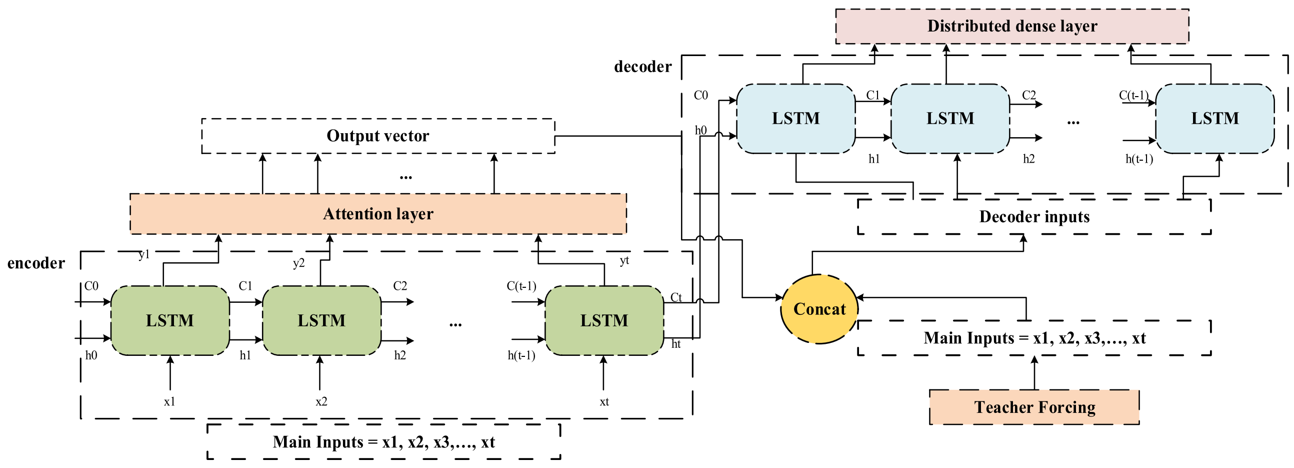


Fig. 11. AE of proposed ADS.

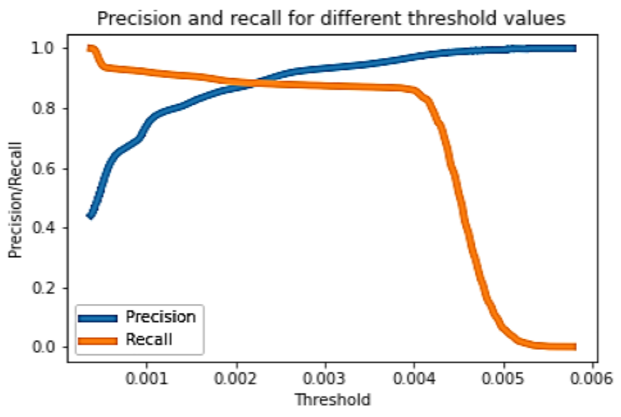


Fig. 12. Precision/Recall Curve on 31 Dec.

to related timesteps. As is shown in Fig. 11, the attention vector at each time step will be concatenated with the main inputs as decoder inputs.

6. Experimental results

6.1. System and model setup

The model has been trained on a machine with Intel(R) Core(TM) i7–10,510U CPU @ 1.80 GHz –2.30 GHz, 1 NVIDIA GP108 PCIe 2GB and 12GB RAM. The model has been implemented based on TensorFlow 2.9.1 using PyCharm IDE to train and the Jupyter notebook to test and fix a threshold.

The model fine-tuned hyper parameters is presented in Table 5. The loss is calculated between the input dense layer of the Encoder and the distributed dense on Decoder outputs. After the second epoch, the reduced learning rate callback with multiplication of eˆ(-(0.1)) is used at the end of each epoch.

Table 5
Finetuned parameters of proposed model.

Parameter	Value	Parameter	Value
Embedding output dimension	32	Number of LSTMs neurons	200
Number of neurons for Modbus values	32	Optimizer	Adam
Number of neurons for static inputs	100	LSTM activation functions	tanh
Number of neurons for concatenation layer	200	Dense layer activation functions	relu
Sequence length	150	Loss function	mse

All of the 7 normal days have been used for training and the 28 Dec from 10:29:14 onwards for the validation, and all of the under attack days have been used to test the model. Fig. 13 shows the training and validation losses are convergent.

6.2. Evaluation metrics

Since the SWaT dataset is extremely imbalanced, it is better to avoid evaluating using precision or f-measure metrics [25]. Instead, the two suitable metrics could be recall and Area Under Curve (AUC) of the ROC curve according to the following formulas.

$$\mathrm{Recall} = tp/tp + fn \tag{9}$$

$$FPR = fp/fp + tn \tag{10}$$

$$TPR = tp/tp + fn \tag{11}$$

6.3. Result analysis

6.3.1. Daily evaluation

In the first step, all the data of the dataset were evaluated on a daily basis. The results are shown in Table 6. As the results show, the proposed NIDS has the most recall values on 31 Dec, which includes more complicated network-based attacks (MSMP, MSSP, SSMP). Compared to the attacks of other days such as December 28 and 29, which include simpler attacks, it can be concluded that the proposed NIDS has the best result for detecting complex attacks that are applied in a distributed manner in several stages and through multiple points.

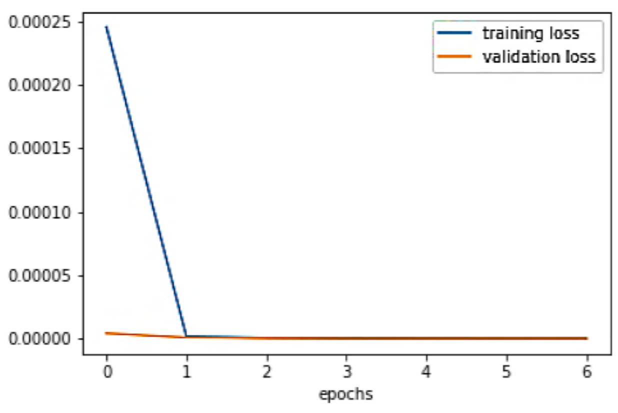


Fig. 13. Training and validation loss.

Table 6
Daily evaluation results.

Day	AUC	Attack *recall*	Normal *recall*	Weighted average *recall*
28 Dec	0.517	0.40	0.73	0.63
29 Dec	0.458	0.21	0.75	0.71
30 Dec	0.840	0.83	0.76	0.76
31 Dec	0.937	0.89	0.90	0.89
1 Jan	0.827	0.78	0.80	0.80
2 Jan	0.667	0.47	0.75	0.66

Table 8
Comparision with history data based models.

Day	Attack	[21]	[20]	[15]	[17]	Proposed model
28 Dec	1	+	+	−	+	+
	2	+	+	+	+	−
	3	+	+	−	−	−
	4	−	−	−	+	−
	5					
	6	+	+	+	+	+
	7	+	+	+	−	+
	8	+	+	+	+	+
	9					
	10	−	+	+	+	−
	11	−	+	+	+	−
29 Dec	12					
	13	+	−	−	−	−
	14	−	−	−	−	+
	15					
	16	−	+	+	−	−
	17	+	+	−	+	+
	18					
	19	+	−	+	+	−
	20	−	+	−	+	+
	21	+	−	+	+	+
	22	−	+	+	+	+
30 Dec	23	+	+	+	−	+
	24	−	−	−	−	+
	25	+	−	+	−	+
	26	−	+	+	+	+
31 Dec	27	+	+	−	+	+
	28	−	+	+	+	+
	29	+	−	+	−	−
	30	+	+	+	+	+
	31	+	+	+	−	−
1 Jan	32	+	+	−	−	+
	33	+	+	−	+	+
	34	+	+	+	+	−
	35	+	+	+	+	+
	36	+	+	+	+	+
2 Jan	37	+	+	+	+	−
	38	+	+	+	+	+
	39	+	+	+	+	+
	40	+	+	+	+	−
	41	+	+	−	−	+

6.3.2. *Network data evaluation*

Due to the complexities of real-time error detection based on network packets, most of founded related works working on SWaT dataset do not use network packets for anomaly detection. Not using network packets for detection and focus on measurements has two major disadvantages: 1. The location of this IDS model is not at the control level and is placed in the top level of the system, as a result, it receives and analyzes data with a longer delay. 2. The IDS models that are placed at the top level are trained based on data values and not based on network behavior. But the great advantage of these IDSs is that they can detect numerical changes well, despite the low efficiency and longer delay, and therefore definitely behave better compared to network-based IDSs that work based on network packets. The main goal of this work was to present an IDS based on network packets that can detect attacks faster and in real time by being placed at the lower level of the network. In this regard, unlike many researchers for finding related works in this field, we just found the works of [18] and [19], which, as shown in Table 7, our proposed model performed better.

In the second step, the proposed model was evaluated for detecting network attacks in real time. Table 7 shows the results compared to previous research. According to our research, only sources [18] and [19] have proposed network-based models for detecting SSSP attacks. The obtained recall values show that the proposed NIDS is better than the previous models, with a significant difference.

6.3.3. *Historical data evaluation*

In the third step, the proposed NIDS has also been compared with 4 other works [15,17,20], and [21] which are based on historical data and their evaluation is based on attack detection. As Table 8 shows, the proposed NIDS is able to detect attacks #24 and #14. According to [15], attack #24 is known as a difficult attack to detect, and attack #14 is impossible to detect.

Considering that the presented model is detecting anomalies based on all transferring network data, the system stages can also not be distinguished. To achieve this goal with further analysis, new features must be reached in the model to separate the stages based on the flow of packets in the network and can detect single-stage or single-point attacks better such as attacks #4 and #13.

Considering that the resources presented in Table 8 are not network-based and are detecting attacks based on the measurements data stored in the SCADA server, they can distinguish stages well and better detect single-stage and single-point attacks.

7. Discussion

The proposed model in this work was a sequence to sequence Auto Encoder. It was trained and tested using traffic packets of the SWaT dataset.

(i) The proposed NIDS locate between levels 1 and 2 beside the central switch and evaluate network packets to detect real-time attacks. While the four previous models were placed on workstations/servers to evaluate historical data.
(ii) The proposed NIDS works with a sequence length of less than a second (150 packets) while the previous models in [21,20,15] work with a sequence length of 120 s, 100 s and 200 s, respectively. Therefore, the detection speed of the proposed NIDS is higher than the previous methods. Reference [17] hasn't mentioned the length of sequence.
(iii) The proposed NIDS learns packet content pattern (header and payload) so it is able to detect zero day network attacks against all parts of the packet. While the previous models [21,20,15] and [17] only work on measuring sensor values in historical data. So they can only detect manipulation attacks against measurement values.
(iv) The results in Table 8 show that the proposed NIDS has found two difficult and indistinguishable attacks (#24,#14). Four comparison models have determined some attacks with uncertainty:

The model [21] which is a classification problem, has accepted even the too small recall value (0.04) as an attack.

The model [20] has accepted the 30 % probability as an attack and less than that as normal.

The model [15] has also declared the too small recalls (0.03) as an

Table 7
Comparision with two network based models.

Attack	*Recall*		
	Model [18]	Model [19]	Proposed NADS
SSSP	0.41	–	0.63
#36	–	0.74	0.86

attack.

The model [17] again has declared the too small recall as an attack (0.004).

The presented model, which is a network-based IDS, has performed better compared to other network-based models presented in Table 8.

The comparison of the presented model with the models that use prepared and stored historical data may not be good, but due to the lack of network-based work on the SWaT dataset, these comparisons were made and presented in Table 9. Although the proposed model is network-based but has not behaved badly.

8. Conclusion

In this paper, a real-time anomaly detection system was introduced in order to detect MITM attacks in a SCADA system. The proposed system is based on the sequence to sequence AE and detect manipulation attacks on Modbus TCP/IP packets. It was shown that the proposed ADS is very suitable for detecting complicated and network distributed attacks including MSMP, MSSP, SSMP. The SSSP attacks can be detected by preparing HIDS for each stage. One of the most important features of the proposed system is the separation of static and dynamic data, the use of attention mechanisms, and teacher forcing techniques to increase detection rates. A combination of precision and recall evaluation metrics were used for tuning the threshold value.

As a future work, it is possible to detect single-stage attacks on PLCs and hosts by further analyzing the dataset and generating new features to better distinguish the stages.

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.

References

[1] K.-D. Lu, G.-Q. Zeng, X. Luo, J. Weng, W. Luo, Y. Wu, Evolutionary deep belief network for cyber-attack detection in industrial automation and control system, IEEe Trans. Industr. Inform. 17 (11) (2021) 7618–7627.

[2] ICS-CERT, "Threat landscape for industrial automation systems," https://ics-cert.kaspersky.com/publications/reports/2021/03/25/threat-landscape-for-industrial-automation-systems-statistics-for-h2-2020/, 25.03.2021 2021.

[3] J. Giraldo, et al., A survey of physics-based attack detection in cyber-physical systems, ACM Comput. Surv. (CSUR) 51 (4) (2018) 1–36.

[4] M. Abdelaty, R. Doriguzzi-Corin, D. Siracusa, DAICS: a deep learning solution for anomaly detection in industrial control systems, IEEE Trans. Emerg. Top. Comput. 10 (2) (2021) 1117–1129.

[5] H. Zhang, Y. Min, S. Liu, H. Tong, Y. Li, MODLSTM: a Method to Recognize DoS Attacks on Modbus/TCP, in: 2022 IEEE International Performance, Computing, and Communications Conference (IPCCC), IEEE, 2022, pp. 319–324.

[6] R. Chalapathy, S. Chawla, Deep learning for anomaly detection: a survey, Comput. Sci. (2019), https://doi.org/10.48550/arXiv.1901.03407.

[7] Y. Luo, Y. Xiao, L. Cheng, G. Peng, D. Yao, Deep learning-based anomaly detection in cyber-physical systems: progress and opportunities, ACM Comput. Surv. (CSUR) 54 (5) (2021) 1–36.

[8] J. Wang, D. Shi, Y. Li, J. Chen, H. Ding, X. Duan, Distributed framework for detecting PMU data manipulation attacks with deep autoencoders, IEEE Trans. Smart Grid 10 (4) (2018) 4401–4410.

[9] S.N. Narayanan, A. Joshi, R. Bose, ABATe: automatic Behavioral Abstraction Technique to Detect Anomalies in Smart Cyber-Physical Systems, IEEE Trans. Depend. Secure Comput. 19 (3) (2020) 1673–1686.

[10] A. Alsaedi, Z. Tari, R. Mahmud, N. Moustafa, A. Mahmood, A. Anwar, USMD: unsupervised misbehaviour detection for multi-sensor data, IEEE Trans. Depend. Secure Comput. 20 (1) (2022) 724–739.

[11] M. Kravchik, A. Shabtai, Efficient cyber attack detection in industrial control systems using lightweight neural networks and pca, IEEE Trans. Depend. Secure Comput. (2021).

[12] S. Adepu, A. Mathur, Distributed attack detection in a water treatment plant: method and case study, IEEE Trans. Depend. Secure Comput. 18 (1) (2018) 86–99.

[13] R.R. Maiti, C.H. Yoong, V.R. Palleti, A. Silva, C.M. Poskitt, Mitigating adversarial attacks on data-driven invariant checkers for cyber-physical systems, IEEE Trans. Depend. Secure Comput. (2022).

[14] G.R. MR, N. Somu, A.P. Mathur, A multilayer perceptron model for anomaly detection in water treatment plants, Int. J. Critic. Infrastruct. Protect. 31 (2020) 100393.

[15] D. Shalyga, P. Filonov, A. Lavrentyev, Anomaly detection for water treatment system based on neural network with automatic architecture optimization, in: ICML Workshop for Deep Learning for Safety-Critical in Engineering Systems, 2018, https://doi.org/10.48550/arXiv.1807.07282.

[16] H.-Y. Kwon, T. Kim, M.-K. Lee, Advanced intrusion detection combining signature-based and behavior-based detection methods, Electronics (Basel) 11 (6) (2022) 867.

[17] Q. Lin, S. Adepu, S. Verwer, A. Mathur, TABOR: a graphical model-based approach for anomaly detection in industrial control systems, in: Proceedings of the 2018 on asia conference on computer and communications security, 2018, pp. 525–536.

[18] P. Mieden, R. Beltman, Network Anomaly Detection in Modbus TCP Industrial Control Systems, UNIVERSITY OF AMSTERDAM, 2020. Available online: https://dreadl0ck.net/papers/RP1_paper.pdf (accessed on 28 November 2022).

[19] M. Pordelkhaki, S. Fouad, M. Josephs, Intrusion Detection for Industrial Control Systems by Machine Learning using Privileged Information, in: 2021 IEEE International Conference on Intelligence and Security Informatics (ISI), IEEE, 2021, pp. 1–6.

[20] J. Kim, J.-H. Yun, H.C. Kim, Anomaly detection for industrial control systems using sequence-to-sequence neural networks. Computer Security, Springer, 2019, pp. 3–18.

[21] C. Wang, B. Wang, H. Liu, H. Qu, Anomaly detection for industrial control system based on autoencoder neural network, Wirel. Commun. Mobile Comput. (2020).

[22] J. Goh, S. Adepu, K.N. Junejo, A. Mathur, A dataset to support research in the design of secure water treatment systems, in: International conference on critical information infrastructures security, Springer, 2016, pp. 88–99.

[23] R.J. Williams, D. Zipser, A learning algorithm for continually running fully recurrent neural networks, Neural Comput. 1 (2) (1989) 270–280.

[24] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, Comput. Sci. (2014), https://doi.org/10.48550/arXiv.1409.0473.

[25] B. Siegel, Industrial anomaly detection: a comparison of unsupervised neural network architectures, IEEe Sens. Lett. 4 (8) (2020) 1–4.

Faeze Zare received the B.Eng. and M.Eng. degrees in computer engineering from the University of Mazandaran, Babolsar, Iran, in 2017 and 2022, respectively. Her research interests include Artificial Intelligence, industrial network security and information security.

Payam Mahmoudi-Nasr received the B.*Sc*. and M.Eng. degrees in computer engineering from the Amirkabir University of Technology, Tehran, Iran, in 1994 and 1996, respectively, and the Ph.D. degree in electrical engineering from the Tarbiat Modares University, Tehran, Iran, in 2016. Since 2008, he has been with the Computer Engineering Department, University of Mazandaran, Babolsar, Iran. His-research interests include industrial network security, information security, and smart grids.

Rohollah Yousefpour is an Associated Professor of Computer Scices Faculty at University of Mazandaran. Rohollah get his B. S. Degree in Applied Mathematics in Shahid bahonar University from Kerman and M.*Sc*. and Ph.D. Degree in Applied Mathematics from Sharif University of Thechnology, Tehran. His-intereset include Machine Lerning and Deep Lerning.