MS-ZeroWall: Detecting Zero-Day Multi-Step Attack in Smart Home using VAE and HMM

Taotao Li, Zhen Hong+, *Member, IEEE*, Wanglei Feng, Li Yu, *Member, IEEE*, and Zhenyu Wen, *Senior Member, IEEE*

***Abstract*—The development of technologies in the smart home provides convenience to our living life, however, the resource-constrained characteristics lead to its frequent exposure to various attacks. Previous techniques for attack detection in the Internet of Things (IoT) are primarily fitted to simple attacks (i.e., single-step attack) but are insufficient analysis for dynamic detection of so-called complicated attacks (i.e., multi-step attacks). Usually, there are three challenges for deploying a multi-step attack prediction system under IoT: 1) resource-constrained, 2) frequently unknown multi-step threats, and 3) high demand for real-time. To address these challenges, in this paper, we propose a multi-step attack prediction architecture (namely, *MS-ZeroWall*) applied in the smart home. Firstly, the *MS-ZeroWall* does not need expensive software, which can be easily deployed on resource-constrained IoT. Then, it captures the characteristics of known threats through the variational auto-encoder (VAE) and uses a VAE-based dual-domain defense strategy (DVAE) to achieve unknown multi-step threat identification. In addition, *MS-ZeroWall* automatically model any multi-step attack by combining the hidden Markov model (HMM) and VAE, and it uses an aggregated HMM (AHMM) approach to improve the multi-step attacks prediction under low time-delay windows so as to satisfy real-time. We evaluate the *MS-ZeroWall* on a publicly available multi-step attack dataset, with an $F1$-score of over 0.96 on unknown multi-step threat identification, and an average accuracy improvement of 12.3% on low-latency multi-step attack prediction.**

***Index Terms*—Internet of Things (IoT), multi-step attack, smart home, variational auto-encoder (VAE), hidden Markov model (HMM).**

I. Introduction

THE number of devices in smart homes is estimated to reach 50 billion by the end of 2021 [1]. It provides convenient services for customers [2], however, it also brings a series of security risks such as Mirai botnet, denial of service (DoS) attack, distributed denial of service (DDoS), jamming attack, spoofing, Man-in-the-Middle (MITM) attack, privacy leakage [3], [4], etc. The main reasons that make IoT devices vulnerable to attacks include their limited computing resources and insecure firmware (does not update frequently) [5].

To protect IoT devices, intrusion detection system (IDS) that uses traditional IT-based (information technology) defense schemes (e.g., firewalls) or modern artificial intelligence (AI)-based learning techniques [6]–[9] is commonly deployed to detect simple attacks (i.e., single-step attack) [4]. However, real-world network attacks are often premeditated and step-by-step [10], namely, **multi-step attacks**, which brings new challenges to existing IDS in IoT. This is difficult to further analyze the attack behaviors and purposes, such as analyzing the adversary's attack path, understanding the relationship between the different attack steps, and making a warning for the arrival of the next step. In this context, it is very significant for the security of IoT systems if we can previously predict the coming attack [11].

Currently, due to several reasons such as intuitive multi-step attack modeling, tracking capability, and low computational overhead, the hidden Markov model (HMM) is the leading approach to detect multi-step attacks [12]–[14] by establishing a probabilistic relationship between attack steps through IDS alert streams. However, there are still three interrelated problems to successfully deploy a multi-step attack prediction system in the IoT environment (e.g., smart home).

Resource-constrained IoT scenario. The HMM's prediction for multi-step attacks relies on rule-based filtering techniques for IDS, e.g., SNORT [15] and Bro, which are resource-consumed [9] and difficult to directly deploy in resource-constrained IoT environment.

Unknown multi-step threat challenge. The existing HMM-based prediction techniques for multi-step attacks mainly make modeling and predicting DDoS attacks [12], [13], without considering the identification of other different types of attacks. It cannot detect unknown multi-step threats, so-called zero-day multi-step threats, which are frequent and difficult to directly detect by machine learning (ML) algorithms in the cybersecurity domain.

Poor prediction under low time-delay. A longer time window for identifying multi-step attacks using HMM provides more information on the attack to ensure accuracy but also brings a longer time delay. Consequently, HMM-based approaches have limited predictive performance in low time-delay windows.

To address the above challenges, we propose *MS-ZeroWall* that uses the variational auto-encoder (VAE) [16] and HMM.

*This work was supported by the National Natural Science Foundation of China under Grants 62072408 and 62073292, the Zhejiang Provincial Science Fund for Distinguished Young Scholars under Grant LR24F020004, Zhejiang Provincial Natural Science Foundation of Major Program (Youth Original Project) under Grant LDQ24F020001 and China Postdoctoral Science Foundation under Grant 2023M743403.

T. Li, Z. Hong, W. Feng, L. Yu, and Z. Wen are with the Institute of Cyberspace Security, and College of Information Engineering, Zhejiang University of Technology, Hangzhou, Zhejiang 310023 China. Z. Wen is a postdoctoral at the University of Science and Technology of China, Hefei, China. Email: {2111903074, zhong1983, 211122030077, lyu, zhenyuwen }@zjut.edu.cn.

Corresponding author: Z. Hong.

The main contributions of this paper can be summarized as follows.

- The proposed *MS-ZeroWall* is appropriate for IoT environment that does not rely on IDS and its rich expert experience. This architecture enables automated modeling and prediction of multi-step attacks with a small computational overhead.
- A VAE-based dual-domain defense strategy (DVAE) is proposed to detect zero-day multi-step threats, which achieves average $F1$-scores over 0.96 and significantly outperforms other approaches.
- We propose an aggregated HMM (AHMM) approach with an average accuracy improvement of 12.3% accuracy at low time-delay window for multi-step attack prediction.

Organization. In Section II, we review the existing work on IDS for smart homes. An overview of the multi-step attack background and problem statement is given in Section III. We propose the details of *MS-ZeroWall* in Section IV. The setups and discussion of experiments are given in Section V. In Section VI, we conclude our paper and describe future work.

II. Related work

A. Single-step attack detection techniques

Aminanto et al. [17] used the sparse auto-encoder (SAE) technique to extract depth features and automatically determine traffic labels, and finally achieved 99.918% accuracy in detecting Evil Twin access point (AP) attacks. Wu et al. [18] based on distributed compression theory, utilized the sparsity and relativity of data to obtain classification features, which could reduce the computational expense and energy consumption in IoT. Zhou et al. [19] used a variation of long short-term memory (VLSTM) architecture to effectively solve the problem of network security data imbalance by introducing mutual information into the VAE. For zero-day threats, Mirsky et al. [20] used multiple auto-encoders (AE) to learn data characteristics and used reconstruction losses to identify zero-day threats. Tang et al. [21] transformed the weblogs into machine language through natural language processing (NLP) and trained them using LSTM-AE to effectively identify zero-day website threats. Unlike this, Anthi et al. [9] proposed the detection of multi-step attack types by three layers of machine learning, where the decision tree (DT) approach gained great results. Yoo et al. [22] propose a recurrent neural network (RNN)-based anomaly detection mechanism to solve the sample imbalance and long sequence detection problems. But their work is also limited to detecting a single stage in a multi-step attack. Li et al. [23] proposed a bidirectional LSTM method to address the long-term behavior detection of multi-stage attacks by combining the time series information of the attacks with the stage features. Although the above research has effective results, no further analysis of attacker behavior and attack paths has been made and no relationships between individual attacks can be inferred, i.e., no consideration of multi-step attacks.

B. Multi-step attack detection techniques

Existing works on multi-step attack prediction are mainly divided into correlation-based [24], [25] and machine learning-based techniques such as HMM, Bayesian networks, clustering [26]–[28]. The correlation-based techniques are used to construct attack graphs by correlation rules to understand the current attack step and warn the next step of the attack. These approaches rely on correlation rules, which require a combination of domain experts' knowledge and increased computational complexity [12]. Among the ML-based techniques, the HMM is considered to be the most suitable method for predicting multi-step attacks. The models using HMM [29] considered the hidden state as the status of the system's risk. These models then only used one HMM for any given attack type. In comparison, [30] argued that the hidden state was a phase characteristic similar to certain types of attacks. In [13], for DDoS attacks, the detection architecture for HMM was constructed by clustering the alert information from IDSs (e.g., SNORT) in traditional IT networks as observations. It modeled the steps of a DDoS attack as the hidden state of the HMM. This detection method has extensive applicability to other attacks because it can model each attack. Shawly et al. [12] argued that HMM-based detection architecture had a large false-positive rate for complex scenarios of cross-attacks. Therefore, it proposed a new detection architecture by training the data according to IP and other information. Their work depends on the alert flow of the IDS, which is generally a rule-based and signature-based scheme. These systems are difficult to deploy in a resource-limited and dynamic IoT environment.

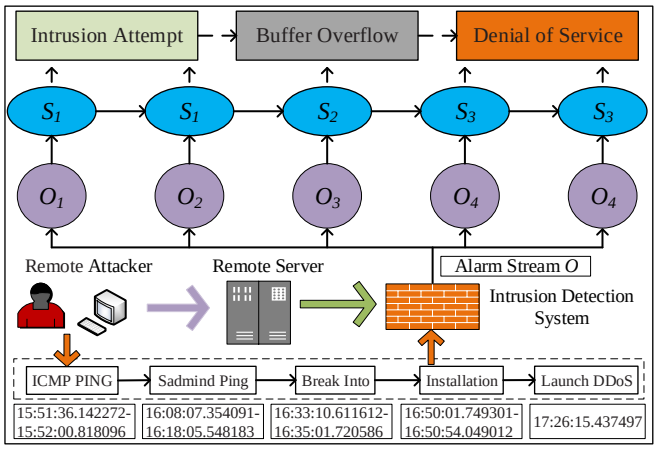


Fig. 1. A remote attacker attempting to incursion a remote server by performing time-dependent attack steps, i.e., the next attack must depend on the successful execution of the previous step. Such multi-step attacks can be modeled and the hidden state of the attacker can be predicted from the IDS alert stream using the HMM.

III. Preliminary Research Conntext

A. Problem Scope

Fig. 1 illustrates a typical multi-step attack (LLDDOS1.0) in the DARPA 2000 [31] dataset. We can see the remote attacker first sends an ICMP PING echo request to determine which service hosts are already up [13]. Then, the attacker probes with Sadmind Ping to identify which hosts are running the administrative tool and determine that the vulnerability can be exploited. Subsequently, the attacker attempts to break into the target host by performing a crack on the Sadmind vulnerability. Finally, the attacker installs Trojan horses or

viruses and launches a DDoS attack. In this case, the attack steps launched by the attacker follow a basic strategy, i.e., in ICMP PING and Sadmind Ping, the attacker makes an intrusion attempt, followed by a breach to exploit a buffer overflow caused by a vulnerability, and finally performs a denial of service. Here, IDS and other anomaly detection tools (e.g., firewall) of remote servers can issue alerts, which can only identify individual attack types but cannot effectively make judgments about the entire attack path and strategy. Therefore, our task is to model and predict a multi-step attack, and then further analyze the attacker's path, understand the relationship between the different attack steps, and make a warning for the arrival of the next step.

In Fig. 1, there is a correlation between different attack steps and events, which can be regarded as a stochastic Markov process. Therefore, HMM is considered the most suitable tool [13] that can depict the relationship between attack steps and anomalous observations through three matrices (i.e. hidden state transfer probability matrix A, observation probability matrix B, and initial states probability matrix π). A HMM is a Markov model λ with hidden states S, and the hidden state is invisible. This HMM (λ) can be defined as

$$\lambda = (A, B, \pi). \tag{1}$$

We define the IDS alert streams as observations $O = [o_1, o_2, \cdots, o_{\mathbb{C}}]$ of HMM, and the three main stages (i.e. intrusion attempt, buffer overflow and DoS launched) of the multi-step are defined as the hidden states $S = [s_1, s_2, \cdots, s_{\mathbb{C}}]$ of HMM. Subsequently, the set of observations (O) and the set of hidden states (S) can be obtained as $\mathsf{V} = \{v_1, v_2, \cdots v_{''}\}$ and $\mathsf{Q} = \{q_1, q_2, \cdots, q_{\#}\}$, respectively, where M is the number of possible observations and N is the number of hidden states. Through learning, HMM can model multi-step attacks as (A, B, π), where A here denotes the attack step transfer probability, B is the observed alert probability between the O and S, and π is the initial attack step probability. We can see that three probability matrices represent the relationship between the different steps and the anomaly alerts, thereby providing us with further analysis of the multi-step attack.

B. Problem Formulation

As mentioned above, the current state-of-the-art multi-step attack prediction systems [13] use HMM to depict the probabilistic relationship between hidden attack states S and alert observations O. We consider that HMM performs the multi-step attack prediction phase. Here, we have a database of K HMMs files to detect multi-step attacks. For g-th alert observation sub-sequence o_6 divided by a time window T, the observation probability is computed to match the most likely HMM by

$$\lambda(o_6)^* = \underset{1 \le : \le}{\arg} \max P(o_6|\lambda_:), 1 \le g \le G, \tag{2}$$

where G is the number of observation sub-sequences, $\lambda(o_6)^*$ denotes the HMM with the highest probability for o_6 and the probability $P(o_6|\lambda_:)$ can be calculated by forward algorithm:

$$\begin{aligned} P(o_6|\lambda_:) &= \sum_{\mathbb{C}=1}^{\mathbb{J}} \alpha_{\mathbb{C}}, \\ \alpha_{\mathbb{C}} &= \Big(\sum_{\mathbb{g}=1}^{\mathbb{f}} \alpha_{\mathbb{C}-1}(i) a^{:}_{\mathbb{g}\mathbb{g}}\Big) b^{:}_{\mathbb{g}}(o^{6}_{\mathbb{C}}), \end{aligned} \tag{3}$$

where $b^{:}_{\mathbb{g}}(o^{6}_{\mathbb{C}})$ denotes the probability that the observation corresponds to state j at time t, $a^{:}_{\mathbb{g}\mathbb{g}}$ denotes the transfer probability of the i-th to j-th attack state, $\alpha_{\mathbb{C}}$ is the forward probability at time t.

There are three difficulties in this process. Firstly, the above HMM-based multi-step modeling approach **relies on the IDS alert flow** to get O and lacks a certain level of automated modeling. For example, when a new multi-step threat is found, we first need to investigate the relevant information and subsequently mark the type of multi-step attack that is necessary. Then, to characterize and model the attack state, we need to further subdivide the anomaly types to form an anomaly alert flow observation by defining IDS rules. This segmentation and labeling process is tedious and requires some expert experience. At the same time, common IDSs are difficult to run directly [9], which requires a large resource footprint. Secondly, cyber attacks in the display world are often carefully planned step by step by attackers. These threats are out-of-sample (i.e. **zero-day attacks**), which pose a greater challenge for either expert rule-defined IDS or advanced ML-based anomaly detection systems. In other words, we can't match the unknown multi-step threats by the maximum probability ($\max P(o_6|\lambda_:)$) calculated by Eq. (2). Thirdly, the individual HMM selected by Eq. (2) depends on the forward probability calculation of Eq. (3). The shorter time window brings limited information about the sequence, and the selected HMM ($\lambda(o_6)^*$) can only prove the maximum probability under the local observation o_6 but not under the overall sequence $O = \{o_1, o_2, ..., o\ \}$. The local optimum does not imply the global optimum, which leads to the **limited and unstable prediction performance** of the matched individual HMMs under the low time-delay window.

IV. Proposed Architecture of *MS-ZeroWall*

A. Overview

To address the above challenges, we present *MS-ZeroWall*, a VAE and HMM-based multi-step attack prediction architecture. Fig. 2 depicts the detailed pipelines of the architecture, which include feature extraction, automatic modeling, threat identification, and aggregate prediction. Initially, we perform feature extraction of network traffic in Section IV-B. To solve the IDS dependency problem, we can employ any of the ML methods instead to perform the anomaly detection task like IDS. However, this does not allow us to make judgments about unknown multi-step threats. Therefore, in the anomaly identification phase, we propose the VAE-based defensive strategies, which aim to identify both known and unknown multi-step threats in Section IV-C. Finally, entering the multi-step attack analysis phase, which is mainly divided into automated modeling and prediction phases. In the automated modeling phase, we use VAE latent vectors Z to automate the process of obtaining anomalous observations O

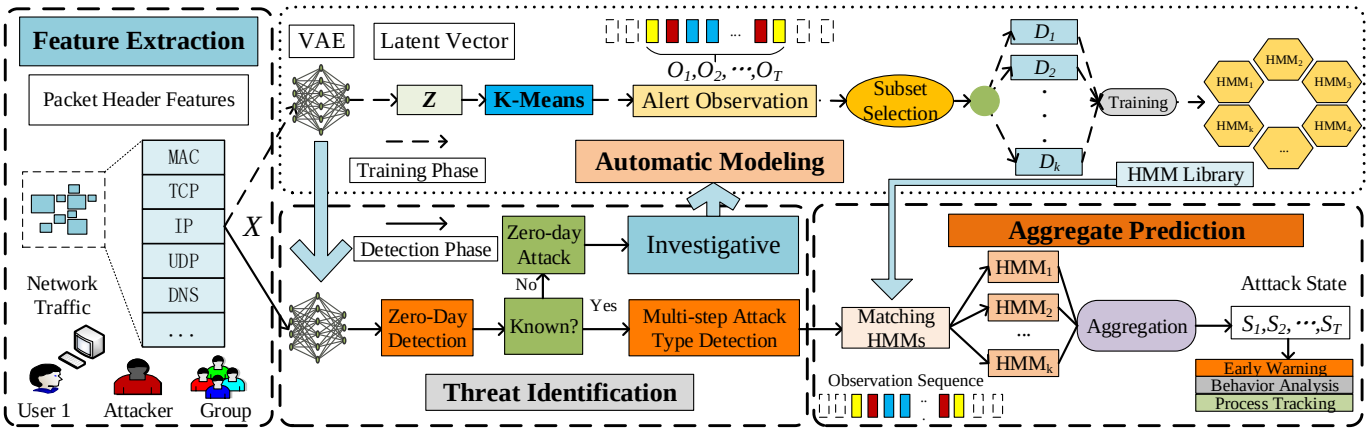


Fig. 2. The workflow of multi-step attack detection framework. 1) Feature extraction: extracting network traffic packet header features. 2) Automatic Modeling: mining feature semantics via VAE and modeling multi-step attacks using HMMs. 3) Threat Identification: identify whether the attack is known or not using VAE's feature comparison, known attacks will proceed by multi-step attack detection and unknown attacks will be investigated. 4) Aggregate Prediction: predicting multi-step attacks using AHMM.

TABLE I
NOTATIONS

Notation	Description
-	Traffic features
-′	Reconstruction traffic features
$	Alert observation
(	Hidden attack state
&	Encoder
%	Decoder
V	Set of observation
Q	Set of state
)	Time window
	Classifier
	State transfer probability matrix
	Observation probability matrix
c	Initial probability matrix
/	Latent vector

TABLE II
PACKAGE FEATURES

Index	Feature Name	Index	Feature Name
1	tcp.window_size_scalefactor	22	udp.dstport
2	tcp.checksum.status	23	udp.length
3	tcp.urgent_pointer	24	udp.checksum.status
4	tcp.options.timestamp.tsval	25	dns.flags.response
5	ntp.precision	26	dns.flags.opcode
6	tcp.analysis.bytes_in_flight	27	dns.flags.truncated
7	tcp.analysis.push_bytes_sent	28	dns.flags.recdesired
8	tcp.time_relative	29	dns.flags.z
9	tcp.time_relative	30	dns.flags.checkdisable
10	tcp.time_delta	31	dns.count.queries
11	tcp.payload	32	dns.count.answers
12	icmp.type	33	dns.count.auth_rr
13	icmp.code	34	igmp.type
14	icmp.ident	35	igmp.max_resp
15	icmp.seq	36	ntp.flags.li
16	icmp.seq_le	37	ntp.flags.vn
17	icmp.resp_in	38	ntp.flags.mode
18	data.len	39	ntp.stratum
19	udp.srcport	40	ntp.ppoll
20	ip.dst	41	ntp.rootdelay
21	ntp.rootdispersion	42	udp.stream

by K-means. Subsequently, subsets are selected and trained separately to obtain low correlation HMMs in Section IV-D. In the prediction phase, HMMs are weighted by aggregation to achieve state inference for multi-step attacks in Section IV-E to improve performance at low time delay. Table I gives the symbolic definitions of the important variables and related parameters of *MS-ZeroWall*.

B. Feature Selection Method

Without loss of generality, we let $X_\beta = \{x_1, x_2, ...x_\#\}$ represents a feature vector extracted from the traffic. The features are mainly information about each packet header, disregarding the encrypted transport data, which can greatly increase the computational speed. It is very important for real-time network intrusion detection in resource-constrained edge devices. Based on packet information, we extract 83 general network protocol features. Note that the source IP address, as well as the MAC address of the device, can't be used as feature information, since it is constantly changing in different smart home network individuals. The correlation analysis gives the top 42 traffic features that we finally extracted in Table II.

C. Threat Identification

Motivation. After obtaining the traffic features X, we need to perform threat identification to match multi-step attack types and unknown zero-day threats. Current advanced zero-day threat detectors use the auto-encoder (AE) [20], [21] architecture, due to its properties of small reconstruction errors for known samples and large errors for unknown samples. However, there are similar steps in a multi-step attack that are invisible anomalies and difficult to detect using only the reconstruction errors of AE. For this reason, we try to further exploit another major feature of the AE, the latent domain feature (i.e., the encoding feature Z of the AE). However, the latent domain properties of the AE are unknown because there is no given prior distribution. Therefore, to control the latent vector data distribution, the VAE [16] provides us with convenience. In VAE, the latent domain of the known samples is close to the normal Gaussian distribution $\mathcal{N}(0,1)$. The unseen sample distribution is some distance away from the $\mathcal{N}(0,1)$ due to being untrained. Combined with the latent domain and reconstruction domain (i.e. dual domain) property, it provides us with the possibility to identify zero-day multi-step attacks.

Fig. 3 illustrates the basic flow of threat identification. VAE

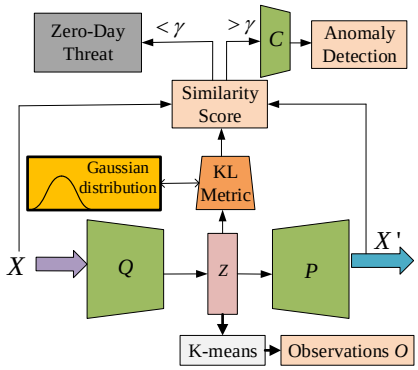


Fig. 3. We utilize the reconstruction error and the latent vector distribution of the VAE as the zero-day threat detection threshold. In addition, the latent vector features will be used as observations for subsequent multi-step attack modeling.

consists of an encoder Q and a decoder P. The traffic features X can be transformed into latent vectors Z by the encoder Q. The traffic reconstruction is then realized by the decoder P. VAE is first trained to identify whether the threat is unknown or not, and then the known threat pounds are identified by classifier C. In particular, the latent vector Z of VAE is further utilized to automate the generation of anomalous observations O via K-means to facilitate the next stage of the multi-step attack analysis.

Training objective. We consider such a generative model of VAE,

$$p(x,z) = p(x|z)p(z), \tag{4}$$

where $p(\bullet)$ denotes the probability density. Then the loss function of a VAE can be formalized as

$$\max_{\backslash} \mathsf{E}_{G\sim ?_{3OC0}}[\log p(x)], \tag{5}$$

where p_{3OC0} denotes the distribution of real data, and θ is the network parameters. The variation process of the loss function is formulated as

$$\begin{aligned} &\mathsf{E}_{G\sim ?_{3OC0}}[\log(p(x))] \\ &\quad = \mathsf{E}_{G\sim ?_{3OC0}}[KL(q(z|x)||p(z|x)) \\ &\quad +\mathsf{E}_{I\sim ?(I|G)}[\log p(x|z)] - KL(q(z|x)||p(z))] \\ &\quad \geq \mathsf{E}_{G\sim ?_{3OC0}}[\mathsf{E}_{I\sim ?(I|G)}[\log p(x|z)] \\ &\quad -KL(q(z|x)||p(z))] = \mathcal{L}_{;>F4A}, \end{aligned} \tag{6}$$

where $p(z)$ denotes the prior distribution of z, and $KL(\bullet)$ represents the Kullback-Leibler (KL) divergence, $q(z|x)$ denotes auxiliary distribution which is used to approximate the real posterior $p(z|x)$. The $q(z|x)$ and $p(x|z)$ can be calculated by encoder Q and decoder P, respectively. Due to $KL(q(z|x)||p(z|x))$ is non-negative, we can get the lower bound loss $\mathcal{L}_{;>F4A}$ of $\log(p(x))$. With VAE, we can get a prior distribution of $p(z)$ by sampling from a distribution $\mathcal{N}(0,1)$.

Zero-day threat detection. In Eq. 6, we refer to $\log p(x|z)$ as the reconfiguration domain and $KL(q(z|x)||p(z))$ as the latent domain. The VAE is trained so that the reconstruction domain loss of known samples converges and the latent domain distribution approximates the prior Gaussian distribution $\mathcal{N}(0,1)$. Therefore, for the zero-day threat, we propose a dual-domain VAE-based (DVAE) defense strategy compared to the general AE approach. We first introduce a similarity ($Sim.$) measure to distinguish known samples from unknown samples, which is defined as follows

$$Sim.(x_{=4F}) = \lambda \sum_{9=1}^{\tilde{O}} P'(x^9_{=4F}) + (1-\lambda) \sum_{9=1}^{\tilde{O}} P\ (z^9_{=4F}), \tag{7}$$

where λ denotes the weights, $P'(x^9_{=4F})$ denotes the reconfiguration domain anomaly probability, $P\ (z^9_{=4F})$ denotes the latent domain anomaly probability, J_1, J_2 denotes the dimension of features and latent vectors, respectively. Then the $P'(x_{=4F})$ can be calculated as follows

$$\begin{aligned} &P'(x^9_{=4F}) = P\{ED(x^9_{=4F},\xi^9) \leq ED(x^9_{=4F},\xi^9)\}, \\ &ED(x^9_{=4F},\xi^9) = ||x^9_{=4F} - x'^9_{=4F}||_2 - \mathsf{E}(\xi^9), \end{aligned} \tag{8}$$

where $\mathsf{E}(\xi^9)$ denotes the average reconstruction error of the known sample, $ED(x^9_{=4F},\xi^9)$ denotes the difference between the reconstruction error of the new sample $x_{=4F}$ and the known sample. After that, $P\ (z^9_{=4F})$ can be calculated as follows

$$\begin{aligned} &P\ (z^9_{=4F}) = P\{KD(z^9_{=4F},\mu_9) \leq KD(z^9_{=4F},\mu_9)\} \\ &KD(z^9_{=4F},\mu_9) = KL(q(z^9_{=4F}|x^9_{=4F})||p(z)) - \mathsf{E}(\zeta_9) \end{aligned} \tag{9}$$

where $KD(z^9_{=4F},\mu_9)$ denotes the difference between the KL measure of the unknown sample and the known sample, $\mathsf{E}(\zeta_9)$ denotes the average KL measure with positively Gaussian distribution $\mathcal{N}(0,1)$ of the known sample, which can be calculated by

$$\zeta_9 = KL(\mathcal{N}(\mu_9,\sigma_9^2))||\mathcal{N}(0,1)) = \frac{1}{2}(-\log\sigma_9^2 + \mu_9^2 + \sigma_9^2 - 1) \tag{10}$$

where $\mathcal{N}(\mu_9,\sigma_9^2)$ denotes the output distribution of the encoder Q, μ and σ^2 denote the mean and variance, respectively.

By calculating the average similarity score $\mathsf{E}(Sim.)$ and comparing it with threshold γ, we can know whether the current sample is a zero-day threat or not. For zero-day threats, investigators need to do further investigation, or else proceed to the multi-step threat detection.

Known threat detection. In this session, we identify known threats to match the corresponding HMM library to facilitate multi-step attack analysis. We use a new linear layer classifier

Algorithm 1 VAE network training and threat identification

Input: $- = \{G_8\}_{8=1}^{}$: Traffic feature vector
$4?>2\ \ B$: The number of training iterations
$\&-\%-$: Encoder $\&$, decoder $\%$, and classifier
$/$: Latent vectors $/$
W: Threshold
Output: Threat type $H_{=4F}$
Randomly initialize network parameters
for $4?>2\ \ = 1$ **to** $4?>2\ \ B$ **do**
  Sample:$G \sim ?_{3OC0}$, $I \sim ?(I)$
  Calculate $\mathcal{L}_{;>F4A}$ and $\mathcal{L}_2$ and update $\&-\%-$ network parameters
end for
Calculate reconstruction loss b and KL metric Z for known samples $-$
Calculate $(8<\bullet$ for unknown samples $G_{=4F}$ by Eq. 7
if $\mathsf{E}((8<\bullet)\ i\ W$ **then**
  $H_{=4F}$ is zero-day threat //Unknown threat
else
  $H_{=4F}$ = argmax$((>5\,C<OG(\ \ (--/)))$ //Known threat
end if
return $H_{=4F}$

C to classify multi-step attacks. Note that our classification model here is optimized separately. Because we need to input the features x into C along with the z to provide more features for performance improvement. So we use multi-classification cross-entropy loss for optimization as follows

$$\mathcal{L}_2 = -\sum_{\beta=1}^{\tilde{O}} y_\beta \log(p_\beta), \tag{11}$$

where y_β denotes the true label, and p_β denotes a predicted probability which can be obtained by $Softmax$. Alg. 1 depicts the process of network training by Adam optimizer [32] and threat identification.

D. Automated Modeling and Subset Selection

Automated modeling. After threat identification, if we detect a new multi-step attack, we need to re-model it, which is a tedious process. The goal of this section is to automate attack modeling and use it to improve efficiency. Unsupervised learning is a better choice because it does not require extensive labeling of exception types. However, the traditional clustering methods like K-means [33] are distance-based and difficult to apply to high-dimensional data. In this paper, high-dimensional traffic data are encoded into the latent domain by VAE. The encoded vectors Z of these latent domains carry important information about the original data X and are low dimensional and can be further exploited to form our alert observations O by K-means. This process improves efficiency by eliminating the need for detailed labeling of anomaly types which is very tedious. For example, in our scenario, the traffic data is close to a million. First we cluster each multi-step attack by the latent vector $Z = \{z_1, z_2, ..., z_=\}$ generated by VAE to get anomaly observation $O = \{o_1, o_2, ..., o_=\}$. We then train the observations for each multi-step attack separately by HMM to get the attack state transfer matrix $A = [a_{\beta\gamma}] \in \mathsf{R}^{\#\times\#}$ and the alarm observation probability matrix $B = [b_\gamma(m)] \in \mathsf{R}^{\#\times"}$, where

$$a_{\beta\gamma} = P(s_{\mathcal{C}+1} = q_\gamma | s_\mathcal{C} = q_\beta), i, j = 1, 2, \cdots N, \tag{12}$$

$$b_\gamma(m) = P(o_\mathcal{C} = v_< | s_\mathcal{C} = q_\gamma), m = 1, 2, \cdots, M. \tag{13}$$

Here Eq. (12) denotes the probability of attack state transferring between q_γ and q_β. Eq. (13) denotes the probability of generating an alert observation $v_<$ in state q_γ. The HMM training is usually done using a Baum-Welch unsupervised algorithm that is consistent with the EM algorithm [34]. For complete data $(O, S) = (o_1, o_2, \cdots, o_\mathcal{C}, s_1, s_2, \cdots, s_\mathcal{C})$, we can obtain the parameters of the HMM $\lambda = (A, B, \pi)$ by training. Different multi-step attacks can extend to get the corresponding set of HMMs.

Subset selection. For the next stage of aggregated HMM (AHMM) for multi-step attack prediction, we need to pick relatively independent subsets of data to train the HMM separately. Because our AHMM is based on ensemble learning, achieving effective prediction presupposes the need for relatively independent classifiers by training independent subsets. For L subsets of a multi-step attack data $\mathcal{D}$, we calculate the frequency $f(O_{;}) \in \mathsf{R}^{1\times"}$ of alert observation in each subset $\mathcal{D}_{;}$. The $f(O_{;})$ is calculated from the number of occurrences of the observation in the subset. For L sub-datasets, the observation frequency similarity matrix is represented as $F(\mathcal{D}) \in \mathsf{R}^{!\times"}$. The $F(\mathcal{D})$ can be calculated by Euclidean distance between each $f(O_{;})$. Finally, the K subsets of low correlations ranked in the top p^* are selected in the L dataset. The training algorithm is used to train these K samples to obtain K HMMs. Alg. 2 describes automated modeling and subset selection of multi-step attacks.

Algorithm 2 Subset selection and Automated Modeling of Multi-Step Attack

Input: Dataset: $\mathcal{D} = \{\mathcal{D}_1\text{-}\mathcal{D}_2\text{-}\bullet\bullet\bullet\text{-}\mathcal{D}_!\}$
Encoder of VAE: $\&$
Output: HMM parameters: $\_ = \{\__1\text{-}\__2\text{-}\bullet\bullet\bullet\text{-}\_\ \}$
1: **for** $; = 1$ **to** $!$ **do**
2: $/_{;} = \&(\mathcal{D}_{;})$
3: Obtaining anomalous observations $\$_{;}$ by K-means($/_{;}$)
4: Calculate frequency array $5(\$_{;})_{1\times"}$
5: **end for**
6: Construct a similarity matrix $(\mathcal{D})_{!\times"}$ by calculating Euclidean distances between $5(\$_{;})_{1\times"}$
7: Sort and select the top $?^*$ subsets:
$\mathcal{D} = \{\mathcal{D}_1\text{-}\mathcal{D}_2\text{-}\bullet\bullet\bullet\text{-}\mathcal{D}\ \}$
8: Train HMMs $(\_{:})$ using the datasets
9: **return** $\_ = \{\__1\text{-}\__2\text{-}\bullet\bullet\bullet\text{-}\_\ \}$

E. Aggregated HMM (AHMM) Prediction Algorithm

Motivation. To be able to improve the prediction performance of HMM for attack sequences, we present the AHMM method. It aggregates the classifiers by exploring the correlation between different classifiers and target sequences. In the training phase, we select and train subsets to obtain HMMs which is described in the IV-D. In the testing phase, we compute the observation probability of HMMs on target sequences. Moreover, we propose a weighted strategy on HMMs to emphasize more HMMs and suppress irrelevant HMMs. As a result, our scheme not only discriminates between different HMMs but also makes a single decision about the prediction task indiscriminately. Here is our explanation of the prediction stage.

Aggregated prediction. In the prediction phase, we use Eq. (2) to calculate the observation probability $P(O_6|\lambda_{:})$ of a sub-anomaly observation sequence O_6. Then we select the K HMM template (λ) with an observation probability greater than 0. The Viterbi algorithm [35] is used to calculate the optimal attack state path S. The Viterbi algorithm $(\mathcal{V})$ uses dynamic planning to solve the maximum probability path, i.e., optimal path, $S^* = \{s_1^*, s_2^*, \cdots, s_)^*\}$, which corresponds to an observed sequence $O = \{o_1, o_2, \cdots, o_)\}$. This path is

$$s_\mathcal{C}^* = \max_{1\le\beta\le\#} [\mathcal{V}_\mathcal{C}(i)], 1 \le t \le T, \tag{14}$$

$$\mathcal{V}_\mathcal{C}(i) = \frac{\alpha_\mathcal{C}(i) \cdot \beta_\mathcal{C}(i)}{P(O|\lambda)}, \tag{15}$$

where $\alpha_\mathcal{C}(i)$ and $\beta_\mathcal{C}(i)$ respectively denote the probability of the forward observable $[o_1, o_2, \cdots, o_\mathcal{C}]$ and the probability of the backward observable $[o_{\mathcal{C}+1}, o_{\mathcal{C}+2}, \cdots, o_)]$. These two variables are defined by the forward and backward algorithm as

$$\alpha_\mathcal{C}(i) = P(o_1, o_2, \cdots, o_\mathcal{C}, s_\mathcal{C} = q_\beta|\lambda), \tag{16}$$

$$\beta_\mathcal{C}(i) = P(o_{\mathcal{C}+1}, o_{\mathcal{C}+;0<1302}, \cdots, o_)\ , s_\mathcal{C} = q_\beta|\lambda). \tag{17}$$

Algorithm 3 AHMM prediction algorithm

Input: Observation sequence $O = \{o_1, o_2, \ldots, o_T\}$
HMM: $\lambda_k, k = 1, 2, \ldots,$
Output: Attack state $S = \{s_1, s_2, \ldots, s_T\}$
1: Split sequence O into sub-sequences $[O_1, O_2, \ldots, O]$
2: **for** $g = 1$ to do
3: **for** $k = 1$ to do
4: Calculate $P(O_g | \lambda_k)$
5: **if** $P(O_g | \lambda_k) > 0$ **then**
6: Calculate $\delta_t(i) = P(s_t = q_i | O_g, \lambda_k)$
7: Calculate $s_t = \max_{1 \le i \le N}[\psi_t(i)], 1 \le t$
8: Calculate $w_k = e^{-\frac{(P(O_g|\lambda_k)-1)^2}{2}}$
9: **else**
10: Drop the k-th HMM template (λ_k)
11: **end if**
12: **end for**
13: Result$(s_t) = \arg\max_{q_i \in Q} \sum_{k=1} w_k \lambda_k(o_t) = q_i$
14: **end for**
15: **return** $S = \{s_1, s_2, \ldots, s_T\}$

Then the preserved HMMs are used to obtain the respectively optimal state paths $S = [s_1, s_2, \ldots, s_t]$ by the Viterbi algorithm for each incoming sub-sequence of observations $O = [o_1, o_2, o_3, \ldots, o_t]$. Finally, for each moment of the observations o_t, we combine different predictive HMMs $\lambda_k(o_t)$ to obtain the final prediction of the target sequence,

$$Result(s_t) = \arg\max_{q_i \in Q} \sum_{k=1} w_k \lambda_k(o_t) = q_i, \quad (18)$$

where q_i is the attack state, $\lambda_k(o_t)$ is the predicted result of the k-th classifier, and w_k indicates the predictor weights. The key issue here is how to choose w_k that represents different source classifiers. Each classifier and target should emphasize more relevant classifiers for the predictive integration of state sequences. We design a weighted strategy based on observation probabilities. We assume that the observation probability $P(O_g|\lambda_k)$ estimated by each classifier for the target sub-sequence O_g after the previous training phase obeys a Gaussian distribution $N(0,1)$. Thus, the weights of each classifier can be calculated as

$$w_k = e^{-\frac{(P(O_g|\lambda_k)-1)^2}{2}}. \quad (19)$$

Alg. 3 gives a detailed description of the AHHM prediction algorithm.

V. Evaluation

A. Experiment Setup

Environment. In our experiments, the *MS-ZeroWall* is built on Python 3.8 in Pytorch. We use a laptop with configures of Intel(R) Core(TM) i5-5200 CPU @ 2.20GHz and 4 GB of RAM embedded to evaluate the architecture. The detailed configurations of the architecture are listed in Table III, and the detailed structure of the network are given in Table IV.

Datasets. Our data comes from two parts. The first is a publicly available dataset of multi-step attacks DARPA2000 [31] LLDDOS 1.0 and LLDDOS 2.0.2. They all consist of 5 steps: 1) IP sweeping, 2) Sadmind probing, 3) Sadmind exploitation, 4) DDoS software installation, and 5) Launching the DDoS attack. We have divided it into three main hidden states: We let steps 1 and 2 as intrusion attempts, steps 3 and 4 as buffer overflow, and step 5 as denial of service. Second, to be able to detect zero-day attacks, we extend the dataset and want to validate it in a real-world smart home scenario as shown in Table V. Fig. 4 shows the smart home experimental platform, which uses Wi-Fi to connect devices including smartphones, smart sockets, smart light bulbs, and T-mall Genie. We use a computer with a Kali system as a wireless attacker to perform a series of multi-step attacks (e.g., DoS and MITM). In particular, we use the Raspberry Pi 4B as a wireless AP to capture the network traffic data through the tcpdump [36]. A DoS scenario has an intrusion attempt, deep system scan, and denial of service, and an MITM scenario includes an intrusion attempt, ARP spoof, and packet injection. During the DoS attack scenario, the intrusion attempt phase uses the most popular Nmap scan to discover the IP address of the target network device. The deep system scan phase performs a deep scan to find open ports and other vulnerabilities in the devices, while the denial of service phase causes the target network devices to temporarily miss communications. Differently, in an MITM scenario, an ARP

TABLE III
Configurations

Parameters	Value
Learning rate (r)	0.0001
Weight parameter (λ)	0.5
Number of states (N)	3
Subset ranking selection points ( )	50%
Similarity threshold (δ)	2%

TABLE IV
Network Structure

Encoder Q	Decoder P	Classifier C
Input feature x	Input latent vector z	Input x and z
MLP 400, Relu	MLP 100, Relu	MLP 100, Relu
MLP 100, Relu	MLP 400, Sigmoid	MLP 20, sigmoid
MLP 20, Sigmoid		Softmax

TABLE V
Dataset Description

Category	Samples	Ratio(%)
Normal	70000	68.21
DoS step 1: Intrusion Attempt	521	0.50
DoS step 2: Deep System Scan	10187	9.9
DoS step 3: Denial of Service	14253	13.80
MITM step 1: Intrusion Attempt	6820	6.64
MITM step 2: ARP Spoof	516	0.50
MITM step 3: Packet Injection	324	0.31

Fig. 4. The smart home network architecture.

(a) (b) (c)

Fig. 5. CDF curves for malicious score distribution of known samples and zero-day attacks with different methods: (a)DDoS1.0 (b)DDoS2.0 (c)MITM(WiFi). Solid lines represent known class attack (KNOWN) samples and dashed lines represent zero-day attack (ZERODAY) samples.

Fig. 6. Accuracy of known class recognition under different similarity thresholdsW At about 0.98, the known classes are tested with the highest accuracy.

spoof places its host between the target device and the gateway to intercept traf c between the devices. Then, we perform active attacks which are mainly based on ICMP protocol packet injection, but other kinds of data injection can also be performed. Table V gives the types of traf c for datasets and the corresponding distributions.

Comparative Baselines.We rst compare our MS-ZeroWall with several recent deep-learning anomaly detection methods:

D-FES [17]: A method for deep feature extraction and anomaly detection using an AE. We adopt its AE architecture and perform anomaly detection.

VLSTM [19]: A VAE-based architecture employs LSTM as the network layer. We directly adopt its architecture for anomaly detection.

AE [20]: A classical approach that exploits the recon guration error of the AE architecture to address zero-day threat detection.

ZeroWall [21]: An approach that exploits the recon guration error of the AE architecture to address web zero-day threat detection, although its network layer uses LSTM.

Multi-step HMM [13]: A recent multi-step attack detection approach through HMM. This paper serves as the infrastructure of the multi-step attack.

Metrics. To quantitatively evaluate the zero-day threat identi cation performance, we use %A428B8>=420;;8 – 1 8 score as metrics, which as follows

$$\%A428B8>=\frac{)\%_8}{)\%_8,\ \%_8}-\ '420;;_8=\frac{)\%_8}{)\%_8,\ \#_8}- \quad (20)$$

$$1_8=\frac{2\ \ \%A428B8>=420;;_8}{\%A428B8>=420;;_8}- \quad (21)$$

where $)\%_8, \%_8, \#_8$ denote the true-positives, false-positives, and false-negatives for the 8th label in label set.

Then for multi-step attacks, we use accuracy (22), error rate (''), and attack probability ($\%_{CC02:}$) as metrics. The '' is given by

$$''\ =\frac{\text{Number of error states detected}}{\text{Total number of attack states}}\ 100\%- \quad (22)$$

and the 22 can be denoted as 1 '' . The attack risk is the probability that an attack will reach an ultimate state of endangering the target. Our attack probability is de ned as

$$\%_{CC02:}{}^{1}C^{o}=\frac{\acute{C}_{C=0}^{F}\ \max_{8}W_{C}{}^{18^{o}}}{}- \quad (23)$$

where C denotes the time point and is calculated using

$$=\ \tilde{O}_{9=0}^{7}\ 4_9-4_9=\begin{cases}{}^{9} & \text{if } 9\ \ \# \text{ and } {}_9\ i\ 2_9\\ 2_9 & \text{else}\end{cases}- \quad (24)$$

where $_9$ is the average number of occurrences of state 9 in the historical data, and 2_9 denotes the cumulative number of occurrences of each state throughout the prediction.

Methodology. We evaluate MS-ZeroWall in the following aspects: i) zero-day multi-step threat detection performance comparison in V-B; ii) known multi-step attack detection performance in V-C; iii) multi-step attack on automated modeling results in V-D; iv) multi-step attack analysis, including attack probability estimation in V-E, attack status prediction in V-F, state tracking and alerting in V-G; v) overhead of MS-ZeroWall in V-H.

B. Zero-day multi-step threat detection performance

First, we perform a zero-day multi-step threat assessment. Our evaluation process is performed by training the VAE with multi-step DoS attacks we collected in a smart home environment as well as normal traf c as known threats and

TABLE VI
ZERO-DAY THREAT ASSESSMENT RESULTS

Model	DDoS 1.0			DDoS 2.0			MITM(WiFi)		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
SVM	0.0001	0.0111	0.0002	0.1939	0.4198	0.2652	0.0043	0.0294	0.0076
DT	0.0013	0.0044	0.0021	0.0870	0.1339	0.1055	0.1336	0.1283	0.1309
D-FES [17]	0.3596	0.0099	0.0191	0.2787	0.0439	0.0743	0.0040	0.0130	0.0060
VLSTM [19]	0.3630	0.0102	0.0198	0.3454	0.2481	0.2864	0.0900	0.0170	0.0290
AE [20]	0.9630	0.0165	0.0306	0.9128	0.4260	0.5764	0.9988	0.0573	0.1063
ZeroWall [21]	0.5074	0.6567	0.5312	0.6200	0.6002	0.6064	0.8696	0.8820	0.8756
VAE [16]	0.9930	0.9822	0.9880	0.9861	0.5438	0.7005	0.3127	0.2404	0.1306
DVAE (Ours)	1.0000	0.9885	0.9942	1.0000	0.9253	0.9612	1.0000	0.9438	0.9710

TABLE VII
HMM STATE TRANSFER MATRIX OF DIFFERENT MULTI-STEP ATTACK

Attack State	DDoS 1.0			DDoS 2.0			Dos(WiFi)			MITM(WiFi)		
	S1	S2	S3	S1	S2	S3	S1	S2	S3	S1	S2	S3
S1	0.700	0.199	0.099	0.699	0.200	0.100	0.600	0.299	0.001	0.994	0.005	0.001
S2	0.100	0.599	0.300	0.099	0.599	0.300	0.102	0.599	0.299	0.001	0.987	0.012
S3	0.099	0.099	0.800	0.099	0.099	0.800	0.099	0.101	0.800	0.001	0.001	0.998
c	0.424	0.297	0.278	0.433	0.298	0.268	0.430	0.29	0.273	0.613	0.260	0.126

TABLE VIII
HMM OBSERVATION PROBABILITY MATRIX OF DIFFERENT MULTI-STEP ATTACK

Observations	DDoS 1.0			DDoS 2.0			Dos(WiFi)			MITM(WiFi)		
	S1	S2	S3	S1	S2	S3	S1	S2	S3	S1	S2	S3
$ 1	0.169	0.156	0.136	0.205	0.208	0.213	0.132	0.135	0.139	0.000	0.000	0.054
$ 2	0.301	0.307	0.315	0.088	0.089	0.091	0.158	0.161	0.166	0.239	0.000	0.000
$ 3	0.177	0.179	0.181	0.264	0.266	0.267	0.106	0.108	0.112	0.000	0.000	0.135
$ 4	0.191	0.194	0.200	0.101	0.102	0.104	0.090	0.092	0.094	0.315	0.000	0.000
$ 5	0.160	0.162	0.165	0.058	0.057	0.055	0.113	0.115	0.118	0.369	0.000	0.000
$ 6	-	-	-	0.059	0.053	0.043	0.159	0.143	0.117	0.070	0.987	0.000
$ 7	-	-	-	0.193	0.193	0.193	0.126	0.128	0.132	0.005	0.000	0.648
$ 8	-	-	-	0.029	0.029	0.030	0.113	0.115	0.118	0.000	0.013	0162

samples. The remaining three multi-step threats (DDoS1.0, DDoS2.0, and MITM (WiFi)) are then used as zero-day threats, which include both attack traf c and normal traf c. We calculate the malicious score by 1 ¹ (8<•º. As can be seen from Fig. 5, the malicious fraction (black solid line) of the known samples calculated using DVAE has a small value, which always stays below 0.08 (red bar line) across the data. And the zero-day threat malicious fraction of DVAE is largely greater than that value. Then other methods such as AE and Zero-Wall appear to be less stable, with the malicious fraction of known samples and the malicious fraction of zero-day threats showing different variations across data sets. In particular, the VAE method using only reconstruction loss (green solid line) also shows a lower malicious score on the known samples, but the malicious score of zero-day threat (green dashed line) shows a larger overlap with the known samples such as in DDOS2.0 and MITM.

Further, we need to manually set thresholds as judgment targets for known and unknown attacks. However, discriminating against unknown attacks presupposes that we must ensure that the known classes have optimal recognition accuracy. Using the training set as a criterion, we nd that the tested test classes can all be classi ed correctly when the similarity is 0.98 as shown in Fig. 6. Therefore, to simultaneously guarantee that the known classes are classi ed correctly and also to avoid a threshold that is too large and leads to a performance decrease of the unknown classes, we choose a critical similarity of

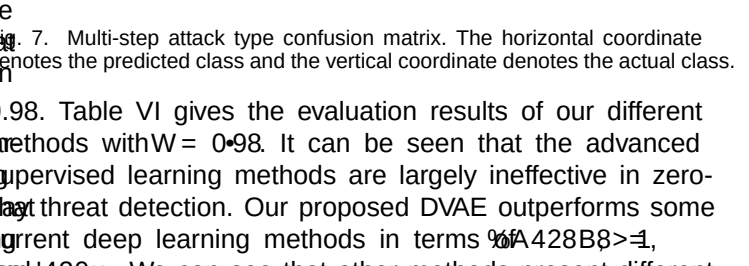


Fig. 7. Multi-step attack type confusion matrix. The horizontal coordinate denotes the predicted class and the vertical coordinate denotes the actual class.

0.98. Table VI gives the evaluation results of our different methods with W = 0•98. It can be seen that the advanced supervised learning methods are largely ineffective in zero-day threat detection. Our proposed DVAE outperforms some current deep learning methods in terms of %A428B8>=1, and '420;; . We can see that other methods present different results when spanning multiple multi-step attack datasets in the domain, which also re ects instability. Our DVAE has a high precision rate, a low value, and a high level of recall. In particular, compared to the traditional AE and VAE, our method exploiting the dual-domain information of latent vector

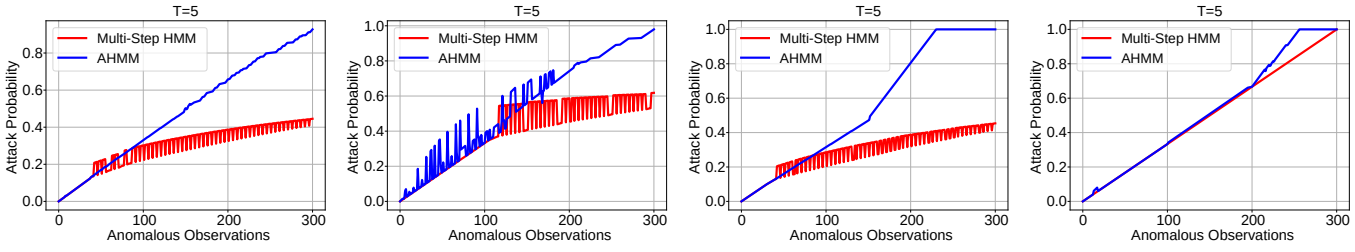


Fig. 8. The evaluation of attack probability predictions for the different multi-step attacks in T =5: (a)DDoS1.0 (b)DDoS2.0 (c)DoS(WiFi) (d)MITM(WiFi).

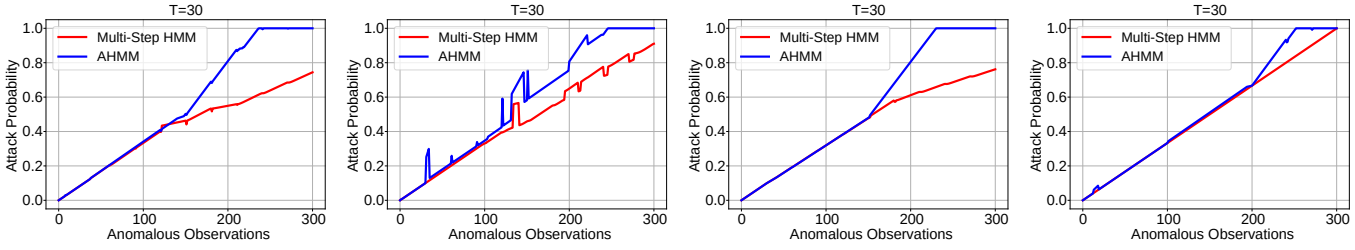


Fig. 9. The evaluation of attack probability predictions for different multi-step attacks in T =30: (a)DDoS1.0 (b)DDoS2.0 (c)DoS(WiFi) (d)MITM(WiFi).

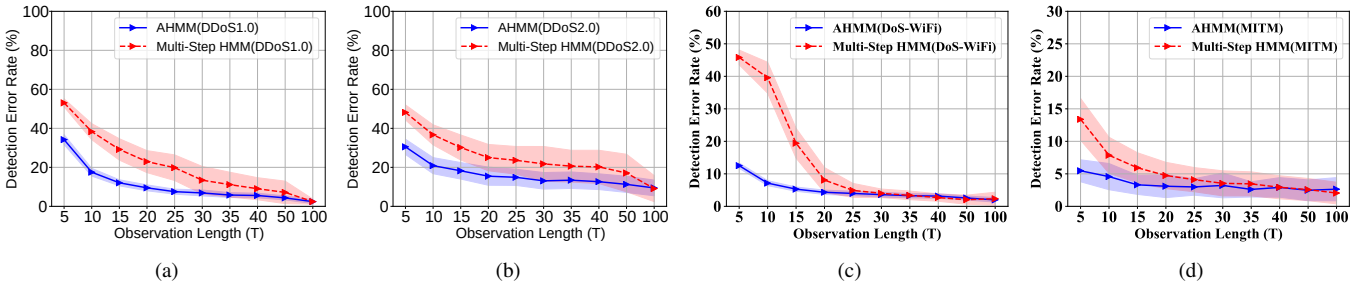


Fig. 10. Prediction error rate evaluation of multi-step attack states under different time windows. (a) DDoS1.0 (b) DDoS2.0 (c) DoS-WiFi (d)MITM-WiFi.

space and reconstruction error will be able to provide richer features and enhance the identification of zero-day threats.

C. Known Multi-step attack Identification

Fig. 7 shows the results of identifying our different multi-step attack types. Using the confusion matrix, we can see that the prediction accuracy of the red category is above 0.98, which is a good classification of the multi-step attack types. The success of the multi-step attack identification relies on the large number of traffic features extracted and reflects the differences in the behavioral traffic of the different attacks.

D. Parameters of the multi-step attack

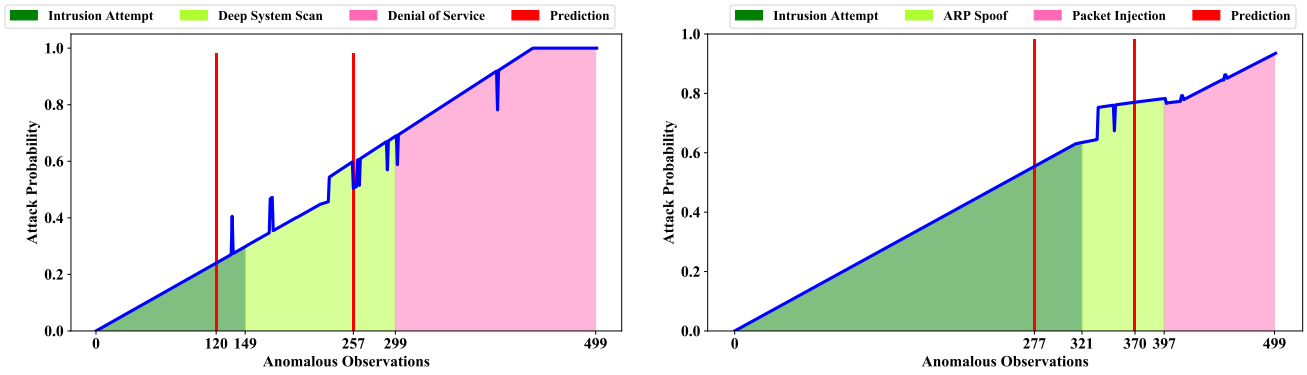
The HMM modeling results for our different multi-step attacks are given in Table VII and Table VIII. In Table VII, we see the probability of maintaining the current state in each attack state S and the probability of transferring to the next step. Also, the initial state probabilities π are given, and the probability of state $S1$ occurring in each multi-step attack is the largest, which indicates that the intrusion attempt phase of the multi-step attack will come in advance. The relationship between different alert observations relative to the hidden states is given in Table VIII. The alert observations O are automatically clustered and we can adjust the number of alerts for different types of attacks. For example, in DDoS 1.0 we only generated 5 types of alerts automatically, because we found that too many alert types do not reflect well the data characteristics, which can be observed from the state transfer matrix A in the modeling results, i.e., a normal attack step will have a larger probability of state transfer in the current step and the next step. The automated modeling process improves efficiency by eliminating the need to give labels and rules for the detailed attack types of multi-step attacks.

E. Attack Risk Probability

After identifying the multi-step attack threat, we perform further analysis of the multi-step attack to obtain more information about the attack behavior. The observation of the attack probability can help us track the progress of each attack. Fig. 8 and Fig. 9 show the attack risk probabilities for different attack scenarios (DDoS1.0, DDoS2.0, DoS-WiFi and MITM-WiFi) in two observation lengths (T=5, T=30). Specifically, the horizontal coordinate represents the entire process of the attack. From Fig. 8, the proposed AHMM reaches the attack confidence probability faster, but the general multi-step HMM does not reach it even after the attack is over. At T=5, the curve is not quite as smooth as it should be because of the shorter length. Similarly, Fig. 9 also shows the same trend when T=30. Our method has a more accurate probability of attack estimation. In addition, our AHMM is less

TABLE IX
MULTI-STEP ATTACK HIDDEN STATE PREDICTION ACCURACY IN LOW LATENCY WINDOWS

T	DDoS1.0(%)		DDoS2.0(%)		Dos-WiFi(%)		MITM(%)	
	AHMM	Multi-Step HMM	AHMM	Multi-Step HMM	AHMM	Multi-Step HMM	AHMM	Multi-Step HMM
5	65.8±2.5	47.0±2.1	69.5±3.9	51.8±3.8	87.4±0.8	54.1±2.2	94.5±1.6	87.6±2.2
10	82.5±2.0	61.7±4.1	79.1±4.1	63.3±5.1	92.8±0.7	60.4±4.7	95.4±2.0	93.1±2.7
15	88.0±1.5	70.9±5.4	81.7±4.2	69.8±6.4	94.7±0.6	80.5±4.5	96.6±1.4	95.0±2.3
20	90.5±1.5	77.1±5.6	84.5±4.6	75.0±6.7	95.6±0.6	91.9±3.8	96.9±1.7	96.2±2.0
25	92.5±1.2	80.2±6.5	85.1±4.1	76.3±7.0	96.0±0.6	95.0±2.1	97.0±1.2	96.8±1.8
30	93.2±1.5	86.4±7.0	87.0±4.2	78.2±8.9	96.3±0.7	95.9±1.3	97.7±1.9	97.4±1.9



(a) (b)

Fig. 11. Pre-alert for multi-step attacks. The red bars denote the point at which the alarm is raised, the three shades denote the different attack steps, the horizontal coordinate is the observation point, and the blue line is the attack probability. (a) DoS-WiFi (b) MITM-WiFi.

influenced by the length of the time window relatively multi-step HMM. Longer observation window lengths T provide more useful information, but it also needs longer acquisition wait times, i.e., increased latency. In summary, individual HMMs perform poorly and are unstable due to the lack of model parameters. Our proposed AHMM has better attack probability tracking performance at low latency time windows when T = 5 and T = 30. The aggregation of multiple HMM parameters provides more observational information to make stable predictions in the short time window.

F. Error Rate Evaluation

The proposed AHMM is evaluated on 500 different batches of attack samples at each observation length (T), and we get the detection ERR under different multi-step attacks in Fig. 10. We can see the AHMM has a smaller average ERR relative to the general Multi-Step HMM. Specifically, in Fig. 10a, the average ERR of the DDoS1.0 attack state prediction at T=20 is around 10%, while the other methods are higher than 20%. Similarly, from Fig. 10b, our scheme has an improved performance relative to general Multi-Step HMM under the DDoS 2.0 scenario. Additional multi-step attacks yielded similar trends and our AHMM has a smaller mean error, which is controlled to ±1%–4%. Table IX gives the accuracy at different time windows T, which indicates that our proposed method not only has better stability and better state prediction accuracy. It is observed that the aggregation of multiple HMMs will utilize more parameters while being able to further improve generalization.

G. Attack Chain Trace and Advanced Warning

In this section, we perform state prediction for a particular attack in a real situation and achieve advanced warning of the attack with an observation length T = 30. Inspired by [37], different predicted instances were specified to obtain a 95% probability. In Fig. 11, the two red bars in the predicted instances to be in each state indicate that the current state process has completed 95%, the three shades denote the different attack steps, the horizontal coordinate is the observation point, and the blue line is the attack probability. The DoS attack prediction in Fig. 11a provides an alert at observation points 120 and 257 on the arrival of the next attack phase, 29 and 49 observation lengths in advance. By the blue lines and shaded areas, we can see the progress of the entire attack. Similarly. the prediction of the arrival of the next attack phase of the MITM in Fig. 11b is advanced by 44 and 27 observation lengths. Advanced warning is very important to help the system react correctly before the damage is caused by compromised devices, especially for smart home devices that are easy to hack. From Fig. 11, we can also see the inference for each state and the probability of the attack reaching the final state, which can be used for state tracking and analysis.

H. Assessment of Resource Consumption

Resource consumption is another main metric of the proposed scheme when compared with existing methods. Table X shows the resource consumption for each phase of the test, where the average time interval between system traffic arrivals is 1.004s. We can see the AHMM reduces the number of parameters by using a subset selection strategy related to HMM. Our AHMM and VAE increase the test delay accordingly, but it is still within the time required for the system traffic to arrive. Fig. 12 shows the cumulative distribution function (CDF) of the time consumed by the *MS-ZeroWall* and Multi-Step HMM in real tests and the time interval between the actual arrival of the system network traffic. We can see that

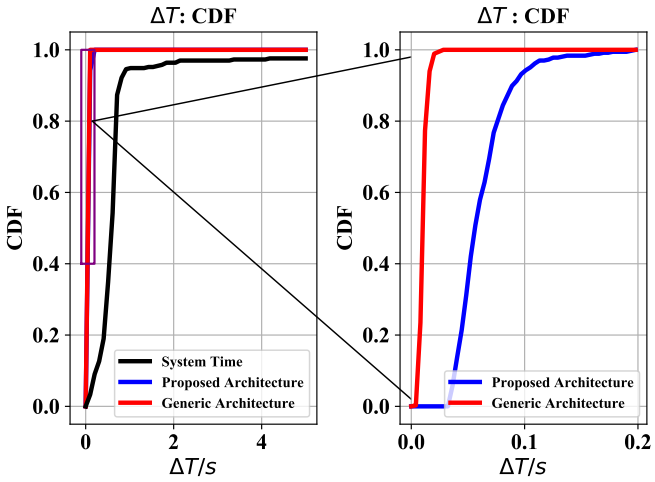


Fig. 12. The CDF of time consumption). The horizontal axis represents time and the vertical axis represents the probability distribution.

TABLE X
RESOURCE CONSUMPTION

Model	Params(kb)	CPU(s)	Traffic Average Delay(s)
K-means	41	0.001	1.004
VAE	336	0.003	
HMM	20	0.012	
AHMM	15	0.062	

our approach, in which CDF of time consumption is mainly between 0.03s and 0.12s increases the time delay relative to the general method, but is broadly lower than the time between system traffic intervals (that is between 0.2s and 1.1s). We believe our approach increases the time and space consumption to some extent, but it is still within the acceptable range of the system. Therefore, future systems can be further considered to deploy on edge devices with limited resources.

VI. CONCLUSION AND FUTURE WORK

In this paper, we propose a multi-step attack prediction system for IoT environments. The identification of different multi-step attacks including zero-day multi-step threats is implemented and can perform automated modeling of multi-step attacks. We propose the AHMM algorithm to improve the performance of multi-step prediction in low latency time windows and finally demonstrate the tracking of multi-stage attacks as well as early warning. However, some shortcomings exist and will be our future work. Due to the limited experiments, we only evaluate four multi-step attack scenarios, but our scheme can be extended for more complex attack-event chains. Our system is expected to be implemented and deployed in future IoT environments.

REFERENCES

[1] M. A. Al-Garadi, A. Mohamed *et al.*, "A survey of machine and deep learning methods for Internet of Things (IoT) security," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 1646–1685, Apr. 2020.

[2] M. Alaa, A. A. Zaidan *et al.*, "A review of smart home applications based on Internet of Things," *Journal of Network and Computer Applications*, vol. 97, pp. 48–65, Nov. 2017.

[3] *Internet of Things (IoT) security and privacy recommendations*. Accessed: 2016. [Online]. Available: https://www.bitag.org/report-internet-of-things-security-privacy-recommendations.php

[4] I. Andrea, C. Chrysostomou *et al.*, "Internet of Things: Security vulnerabilities and challenges," in *Proc. 2015 IEEE Symposiumon Computers and Communication (ISCC)*, 2015, pp. 180–187.

[5] R. Doshi, N. Apthorpe *et al.*, "Machine learning DDoS detection for consumer Internet of Things devices," in *Proc. IEEE Security Privacy Workshops (SPW)*, 2018, pp. 29–35.

[6] R. Coulter, Q.-L. Han, L. Pan, J. Zhang, and Y. Xiang, "Data-driven cyber security in perspective—intelligent traffic analysis," *IEEE Transactions on Cybernetics*, vol. 50, no. 7, pp. 3081–3093, 2019.

[7] S. Tang, Z. Gu *et al.*, "Smart home IoT anomaly detection based on ensemble model learning from heterogeneous data," in *Proc. IEEE International Conference on Big Data (Big Data)*, 2019, pp. 4185–4190.

[8] F. Hussain, R. Hussain *et al.*, "Machine learning in IoT security: Current solutions and future challenges," *IEEE Communications Surveys & Tutorials*, 2020.

[9] E. Anthi, L. Williams *et al.*, "A supervised intrusion detection system for smart home IoT devices," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 9024–9053, 2019.

[10] R. Gan, Y. Xiao, J. Shao, and J. Qin, "An analysis on optimal attack schedule based on channel hopping scheme in cyber-physical systems," *IEEE Transactions on Cybernetics*, 2019.

[11] B. B. Zarpelao, R. S. Miani *et al.*, "A survey of intrusion detection in internet of things," *Journal of Network & Computer Applications*, vol. 84, pp. 25–37, Apr. 2017.

[12] T. Shawly, A. Elghariani *et al.*, "Architectures for detecting interleaved multi-stage network attacks using hidden Markov models," *IEEE Transactions on Dependable and Secure Computing*, 2019.

[13] P. Holgado, V. A. Villagrá *et al.*, "Real-time multistep attack prediction based on hidden Markov models," *IEEE Transactions on Dependable and Secure Computing*, vol. 17, no. 1, pp. 134–147, 2020.

[14] A. A. Ramaki, A. Rasoolzadegan *et al.*, "A systematic review on intrusion detection based on the hidden Markov model," *Statistical Analysis and Data Mining: The ASA Data Science Journal*, vol. 11, no. 3, pp. 111–134, 2018.

[15] M. Roesch *et al.*, "Snort: Lightweight intrusion detection for networks." in *Lisa*, vol. 99, no. 1, 1999, pp. 229–238.

[16] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[17] M. E. Aminanto, R. Choi *et al.*, "Deep abstraction and weighted feature selection for Wi-Fi impersonation detection," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 3, p. 621–636, 2018.

[18] D. Wu, H. Shi *et al.*, "A feature-based learning system for Internet of Things applications," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1928–1937, 2018.

[19] X. Zhou, Y. Hu *et al.*, "Variational lstm enhanced anomaly detection for industrial big data," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 5, pp. 3469–3477, 2020.

[20] Y. Mirsky, T. Doitshman *et al.*, "Kitsune: an ensemble of autoencoders for online network intrusion detection," *arXiv preprint arXiv:1802.09089*, 2018.

[21] R. Tang, Z. Yang *et al.*, "Zerowall: Detecting zero-day web attacks through encoder-decoder recurrent neural networks," in *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*. IEEE, 2020, pp. 2479–2488.

[22] Y.-H. Yoo, U.-H. Kim, and J.-H. Kim, "Recurrent reconstructive network for sequential anomaly detection," *IEEE Transactions on Cybernetics*, 2019.

[23] X. Li, M. Xu, P. Vijayakumar, N. Kumar, and X. Liu, "Detection of low-frequency and multi-stage attacks in industrial internet of things," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 8, pp. 8820–8831, 2020.

[24] N. Asadi, A. Mirzaei, and E. Haghshenas, "Creating discriminative models for time series classification and clustering by hmm ensembles," *IEEE Transactions on Cybernetics*, vol. 46, no. 12, pp. 2899–2910, 2016.

[25] L. Wang, A. Liu *et al.*, "Using attack graphs for correlating, hypothesizing, and predicting intrusion alerts," *Computer communications*, vol. 29, no. 15, pp. 2917–2933, 2006.

[26] D. S. Fava, S. R. Byers *et al.*, "Projecting cyberattacks through variable-length markov models," *IEEE Transactions on Information Forensics and Security*, vol. 3, no. 3, pp. 359–369, 2008.

[27] H. Du, D. F. Liu *et al.*, "Toward ensemble characterization and projection of multistage cyber attacks," in *2010 Proceedings of 19th International Conference on Computer Communications and Networks*. IEEE, 2010, pp. 1–8.

[28] A. A. Ramaki, M. Amini *et al.*, "Rteca: Real time episode correlation algorithm for multi-step attack scenarios detection," *computers & security*, vol. 49, pp. 206–219, 2015.

[29] H. A. Kholidy, A. Erradi *et al.*, "A finite state hidden Markov model for predicting multistage attacks in cloud systems," in *Proc. IEEE 12th Int. Conf. Dependable Autonomic Secure Comput.*, 2014, p. 14–19.