This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2020.3036664, IEEE Transactions on Dependable and Secure Computing

Threat-Event Detection for Distributed Networks Based on Spatiotemporal Markov Random Field

Haishou Ma, Yi Xie, Shensheng Tang, Jiankun Hu, Xingcheng Liu

***Abstract*—Distributed threat-events are one of the main challenges facing the networks. Although a lot of research has been conducted for these issues, the situation has not been significantly improved. Different from existing victim-centric approaches, in this work we propose a new network-centric approach for the detection of distributed threat-events. The distributed network is treated as a holistic system that consists of spatially interconnected network elements. Network events are detected by the dynamic behavior analysis of the distributed networks. We develop a model consisting of two-layer random fields to describe the time-varying traffic forwarding behavior of the distributed networks. The bottom layer describes the interaction and influence of the network elements under the action of network events. Markovianity is adopted to characterize the spatiotemporal context of each network element's behavior patterns. The top layer describes each network element's traffic features driven by the underlying behavior patterns. A Gaussian mixture model is used to capture the statistical features of the network traffic for each behavior pattern. We derive algorithms for parameter estimation and event detection. Numerical experiments using real datasets and different network scenarios are presented to validate the proposed approach. Performance-related issues and comparison with related works are discussed.**

***Index Terms*—Spatiotemporal context, Distributed networks, Threat-event detection, Markov random field.**

I. Introduction

Distributed threat-events (DTEs) have been one of the major challenges facing the network. Their main feature is that massive physical (or virtual) computing and communication resources are abused for malicious purposes. A DTE manifests itself in various ways, such as distributed denial of service (DDoS) attacks [1], virus and worm propagation [2].

Recently, with the rapid improvement of the emerging Internet technologies, e.g., software-defined networking (SDN), network virtualization (NV), Internet of Things (IoT) and the 5^{th} generation (5G) network, the role of networks is changing. It is no longer limited to acting as a data-pipe but carrying more and more functions and services, such as network-based data caching [3], virtualized data transmission [4], multi-homing services [5], and interconnection of heterogeneous networks [6]. Therefore, besides the specific attack targets, the hazard of DTEs also seriously threatens the availability and stability of the infrastructure and various communication services, such as wide area networks, enterprise networks, data center networks and social networks. For example, in the scenario of virtualized network slicing [4], a DTE that targets at a specific victim and spreads on a physical infrastructure or a virtual-layer service network will compromise the performance of all related network communication services, which adversely affects the interests of many parties such as infrastructure operators, virtual network tenants, service providers, and massive end customers. Therefore, detecting and resolving DTEs from the perspective of the network-side has become one of the major challenges for network operators, which is also the main motivation of this work.

Literature survey shows that in the past two decades a lot of research has been conducted for detecting various types of DTEs, e.g., DDoS [7], malware [8], ransomware [9], rumors and spams [10]. However, most of them focus on the protection of the specific victims rather than the communication networks, which makes them follow a victim-centric approach (VCA) and unsuitable for distributed networks. The main limitation of these VCAs is that the controllable resources (including network facilities and available time for analysis and response) are usually severely squeezed by the ongoing DTE. Due to this reason, collaborative attack detection (CAD) has received extensive attention, such as correlation analysis [11], multi-domain filtering [12], interaction of adjacent autonomous systems (ASes) [13], and traceback through routers [14]. These works mitigate the impacts of the above VCA limitation by adding measurement probes around the victims, which enables the victims to obtain more information. From the perspective of their working principle, they are not network-oriented solutions, because they do not provide a holistic view and modeling method for the time-varying behavior of distributed networks. Moreover, they don't take full advantage of the inherent distributed interconnection features of networks to achieve collaborative early detection.

Motivated by these pioneer works, we propose a new network-centric approach (NCA) for detecting DTEs from the perspective of distributed networks. The proposed NCA is expected to improve the performance of DTE detection from three aspects: (i) The VCAs can only deal with the threats it faces, while the NCA can deal with the threats against different targets, because the network is the only means of spreading threat events. (ii) The VCAs are difficult to resist large-scale

H. Ma and X. Liu are with the School of Electronics and Information Technology, Sun Yat-Sen University, Guangzhou 510006, P.R. China. E-mail: mahaishou_a@qq.com, isslxc@mail.sysu.edu.cn

Y. Xie is with the School of Data and Computer Science (and the GuangDong Province Key Laboratory of Information Security Technology), Sun Yat-Sen University, Guangzhou 510006, P.R. China. E-mail: xieyi5@mail.sysu.edu.cn,

S. Tang is with the Dept. of Electrical and Computer Engineering, St Cloud State University, St Cloud, MN 56301, USA. E-mail: stang@stcloudstate.edu

J. Hu is with the School of Engineering and Information Technology, University of New South Wales at the Australian Defence Force Academy, Canberra, ACT 2600, Australia. E-mail: j.hu@adfa.edu.au

Corresponding author: Yi Xie

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2020.3036664, IEEE Transactions on Dependable and Secure Computing

threat events due to the limited resources, while the NCA can use distributed links to mitigate, capture and trace back the threat events. (iii) Compared with the VCAs, the NCA can perceive threat events earlier and implement early defense measures. In this work, we treat the distributed network as a holistic system that consists of interconnected network elements (NEs) and acts as a carrier for network events. Because the propagation of network events is achieved by means of data transmission (e.g., IP packets) and drives the traffic forwarding behavior of each NE located on the propagation paths, the spatiotemporal distribution of a network's behavior patterns can be used as the fingerprint for the network event detection. The key issue is to derive a reasonable model for describing the time-varying behavior process of the distributed network. To this end, we adopt the framework of Markov random fields (MRFs) [15] in this work. The proposed model consists of two-layer random fields. In the bottom layer, we define a finite state set to denote the behavior patterns for each NE under the actions of network events and adopt Markovianity to characterize the spatial and temporal context of each NE's behavior patterns. The spatial context of the behavior patterns describes the correlation and interaction of the interconnected NEs, while the temporal context describes the time-varying process of the same NE's behavior patterns. The top layer represents the traffic features observed at each NE driven by the underlying behavior patterns. We further use a Gaussian mixture model (GMM) to capture the statistical characteristics of the network traffic for each underlying behavior pattern.

Based on this model, each NE is able to automatically determine the ongoing network event based on the spatiotemporal Markovianity of the behavior patterns, while the network operator can infer the macroscopic evolution of the ongoing network event via the spatiotemporal distribution of the NEs' behavior patterns. Thus, the DTE detection ultimately comes down to the problems of model learning and behavior pattern recognition. To solve these problems, we derive the parameter estimation and event detection algorithms for the proposed model based on the algorithms of expectation-maximization (EM) and maximum a posteriori (MAP). We conduct two experiments using the open real datasets and different distributed network scenarios to validate the proposed solution. Performance-related issues and comparison with related works are discussed. In summary, the main contributions of this paper are threefold:

- A new NCA is proposed to detect DTEs for distributed networks. We treat a distributed network as a holistic system, and achieve the DTE detection through the dynamic behavior characteristics of the network.
- A two-layer model is developed to formulize the proposed NCA based on an MRF-framework. Markovianity is used to describe the spatial and temporal context of a distributed network's time-varying behavior patterns.
- Algorithms for model learning and event detection are derived based on the EM and MAP algorithms. Performance evaluation results are presented based on two experiments with real datasets.

The rest of this paper is organized as follows. In Section II, related works are surveyed. In Section III, we introduce the proposed scheme. In Section IV, experiments and results are presented. Some issues related to this work are discussed in Section V. Finally, Section VI is devoted to the conclusions and the further work.

II. Related works

In this section, we will briefly summarize the existing DTE detection techniques. Based on the published surveys [10] [16] [17], we classify the related works into four categories by the types of their approaches.

The threshold based approaches have been widely used in industrial applications. The data come from monitoring systems, while the threshold values are usually pre-defined by experts. When the measured values exceed the preset threshold range, it indicates the occurrence of a network event [18]. Yang et al. [19] proposed a Two-Phase Self-Join (TPSJ) scheme to evaluate self-join queries for an event detection in sensor networks. Krishnamachari et al. [20] used a wireless sensor network (WSN) for detecting an environmental phenomenon in a distributed manner. The threshold-based approaches can meet the requirements of simple engineering applications. The main issue is that they rely too much on prior knowledge and lack flexibility, as the single static threshold is not suitable to the complex and varying network environments. Moreover, these methods can only report the current state of network events but cannot characterize their dynamic evolution process. Thus, it is difficult for them to achieve early warning and response.

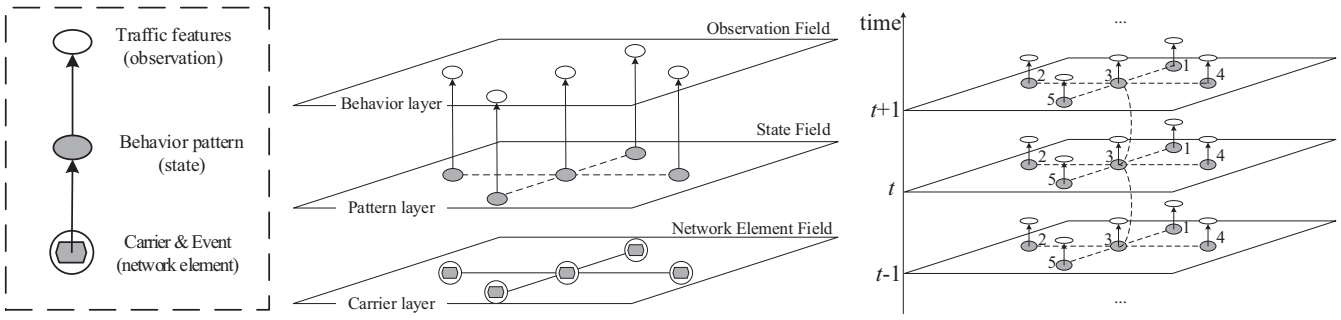
Probabilistic reasoning based approaches also have been used to model and identify the behavior of distributed scenarios. In Wen's work [21], temporal dynamics and spatial dependence were taken into account to model and identify the propagation of social network worms. In the work of Karyotis [22], a stochastic framework was proposed for modeling the communication network under random attacks. These works take into account spatial interactions of adjacent nodes but ignore time-varying factors. Moreover, they are more concerned with the probability that the individuals in the network are affected, but not the event detection.

Signal processing is another commonly used method. In Sadreazami's work [23], the measurements of a sensor network were modeled by a graph signal whose statistical properties were utilized for intrusion detection. In Illiano's work [24], wavelet transform was proposed to detect the spoofed and masked events for wireless sensor networks, including malicious data injections and false network events. Jiang et al. [25] grasped the time and frequency features of the network-wide traffic based on a transform domain analysis, and utilized the difference in high frequency features to detect the abnormal network traffic. The signal processing approaches are always with high computation complexity. For example, the graph-signal based algorithm requires an eigenvalue decomposition on the graph Laplacian. Moreover, because signal models in different applications and scenarios are very different from each other, it is not easy to ensure versatility.

Machine learning has recently received wide attention in various fields. Wu et al. [26] presented a framework for

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2020.3036664, IEEE Transactions on Dependable and Secure Computing



(a) Single-point model. (b) Distributed spatial model. (c) Spatiotemporal context model.

Fig. 1: Modeling approach.

detecting remote access Trojans at the area network borders. A Naive Bayes classifier is used to detect the anomaly from the IP flows. Peng et al. [27] proposed a clustering method for an intrusion detection system based on the mini batch KMeans. Yang et al. [28] proposed a Gaussian-mixture model-based detection scheme to mitigate data integrity attacks in a smart grid, which operates through narrowing the range of normal data. Li et al. [29] presented a robust multivariate probabilistic calibration model for network-wide anomaly detection and localization. They applied the latent variable probability theory with multivariate t-distribution to establish the normal traffic model, and detected the network anomaly by the Mahalanobis distance of samples. Recently, we have also noticed that deep neural networks (NNs) have been widely used in intrusion detection, such as recurrent neural networks (RNNs) [30], convolutional neural networks (CNNs) [31] and long short-term memory (LSTM) [32]. The first limitation of them is that they are victim-centric rather than network-centric. The second is that most of the NN-based schemes cannot provide an reasonable mathematical analysis for the physical processes of the problems to be solved. Moreover, a lot of labelled data are required for them, which limits their application in real-world scenarios.

Overall, although DTE detection has received wide attention and made some progress, most of them still follow the victim-centric architecture. The detection methods designed according to the characteristics of the network-side are still rare. Machine learning methods based on deep NNs may be a new direction in the future. However, these technologies are not yet suitable for the unsupervised scenarios considered in this work. Moreover, almost every solution is bundled with a specific application scenario, which leads to the lack of versatility and flexibility.

III. The Proposed Approach

A. Rationale

Different from existing works, here we introduce a new NCA for DTE detection from the perspective of network-side. We derive the proposed spatiotemporal behavior model in three steps and start with a single-point model.

Figure 1(a) shows a model to describe the working mechanism of an NE[1]. It consists of four basic elements, including the entity of an NE (the bottom circle), an ongoing network event (the gray hexagon inside the bottom circle), the traffic forwarding behavior patterns (the middle gray ellipse) of the NE, and the measurable traffic features (the top ellipse), e.g., arrival rate. Their relationship is as follows: (i) the NE is a data forwarding device, and acts as a carrier or container for the ongoing network event; (ii) the patterns of an NE's traffic forwarding behavior are controlled by the current event; (iii) the external observable traffic features are generated by the underlying pattern of traffic forwarding behavior. In this model, an NE's behavior pattern and the event that it encounters are closely related, and further influence the NE's external traffic features. From the perspective of DTE detection, the external traffic features refer to the observable physical phenomena and metrics, while both the behavior pattern and the event represent the unmeasurable (hidden) logical factors that can only be inferred by the external traffic features. For example, when a virus propagation event passes through an NE, the NE will be forced to switch its current forwarding pattern to another one that meets the requirements for virus propagation and further causes a large number of port scanning traffic to be forwarded from this NE. Combined with time series analysis methods, the single-point model can be used for anomaly detection, which has been widely adopted by VCAs [26].

However, the main drawback of the single-point model is that it can only implement the local detection, but cannot capture the interactivity of the behavior patterns and the spatial distribution of events. Thus, it is unable to provide the overall view of the events from the network-side perspective. For this reason, we extend it to a distributed spatial model shown in Fig. 1(b) that adopts "spatial field" to describe the spatial context and interactivity of the interconnected NEs. In physics, a field is a physical quantity represented by a number or tensor that has a value for each point in space-time. Here, we use three fields to build the distributed spatial model. The bottom is "NE-field" that consists of interconnected NEs. The solid lines shown in the NE layer of Fig. 1(b) denote the physical/virtual links connecting the NEs. This layer acts as the distributed carrier of network events. The middle is "state-

[1]In this work, we do not distinguish the type of NE. An NE can be a physical/virtual router, switch, gateway, proxy or any other device that participates in packet/data forwarding.

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2020.3036664, IEEE Transactions on Dependable and Secure Computing

field" that describes the spatial context of the behavior patterns for the interconnected NEs. The top is "observation-field" that is composed of the external behavior features of each NE. Based on this model, we can estimate the spatial distribution of the behavior patterns for all NEs, and further infer the potential attributes of the ongoing network event, such as the type and the purpose.

Because time continuity is a prominent feature in the evolution of network events, the behavior patterns of an NE are not only spatially correlated, but also time-dependent, i.e., the traffic forwarding behavior patterns that an NE has experienced in the past usually have an important impact on its behavior in the near future. Therefore, we further extend the above distributed spatial model to a spatiotemporal context model shown in Fig. 1(c) that describes the time-varying spatial state field and the observation field[2]. In this model, the interconnected gray ellipses denote the spatiotemporal distribution of the event-related behavior patterns and form the spatiotemporal state-field (SSF), while the white ellipses denote the external traffic features emitted by the NEs and form the spatiotemporal observation-field (SOF). For example, compared with Fig. 1(b), in Fig. 1(c) two dotted lines are used to connect NE3's behavior patterns adopted at $t-1$, t, and $t+1$, which describes the temporal context of the traffic forwarding behavior patterns.

Based on this spatiotemporal behavior model, the network-centric DTE detection can be attributed to inferring the hidden SSF via the measurable SOF, which is a typical distributed unsupervised labeling problem. Unlike the existing victim-centric anomaly detection, it involves the analysis and inference of spatiotemporal context.

B. Formulation

According to the above modeling approach, we use an undirected graph $G =< \mathcal{K}, \mathcal{E} >$ to describe the underlying distributed network, where $\mathcal{K} = \{1, ..., K\}$ and $\mathcal{E}$ denote the sets of NEs and the edges between two adjacent NEs, respectively. We use $v_{t,k} \in \mathcal{V}$ to denote the k^{th} NE observed at the t^{th} time slot, where $k \in \mathcal{K}, t \in \mathcal{T} = \{1, ..., T\}$, and $\mathcal{V} = \{v_{1,1}, \cdots, v_{1,K}, v_{2,1}, \cdots, v_{2,k}, \cdots, v_{T,1}, \cdots, v_{T,K}\}$ is the collection of all $v_{t,k}$ for all (t, k). Let $v_t = \{v_{t,1}, ..., v_{t,K}\}$ denote a snapshot or slice of spatial topology of NEs observed at time t. Then, the snapshot sequence of the distributed network varying with time can be simply expressed as $v_{1:T} = (v_1, ..., v_T)$. Similarly, let $S_{t,k}$ and $O_{t,k}$ denote the random variables of hidden state and observation of $v_{t,k}$, respectively. The lowercase variables $s_{t,k} \in \mathcal{S}$ and $o_{t,k} \in \mathcal{O}$ denote the instances of $S_{t,k}$ and $O_{t,k}$, respectively. $\mathcal{S}$ is the set of all possible states, while $\mathcal{O}$ is the collection of all candidate observations. Let $S = \{S_{t,k} | t \in \mathcal{T}, k \in \mathcal{K}\}$ and $O = \{O_{t,k} | t \in \mathcal{T}, k \in \mathcal{K}\}$ respectively denote the random variables of the entire hidden SSF and SOF, while $s \in \mathbb{S}$ and $o \in \mathbb{O}$ denote the instances of S and O, respectively. $\mathbb{S}$ and $\mathbb{O}$ are the sets of all possible configurations of hidden SSF and SOF, respectively.

[2]In order to highlight the spatiotemporal context model, we omitted the NE-field here.

Based on the previous section, network-centric DTE detection can be solved by inferring the hidden SSF via the measurable SOF, which is equivalent to seeking an optimal $\hat{s}$ given o and the parameter set Ω of the model, i.e., seeking an optimal s that satisfies formula (1)

$$\hat{s} = \arg\max_{s \in \mathbb{S}} \{\Pr[s|o, \Omega]\}. \tag{1}$$

Based on the Bayes theorem, the posterior probability $\Pr[s|o, \Omega]$ can be calculated by the emission probability $\Pr[o|s, \Omega]$ and the prior probability $\Pr[s|\Omega]$, as shown in formula (2) where the term $\Pr[o]$ is a constant:

$$\Pr[s|o, \Omega] = \Pr[o|s, \Omega] \cdot \Pr[s|\Omega] / \Pr[o]. \tag{2}$$

The conditional probability $\Pr[o|s, \Omega]$ in formula (2) represents the probability distribution of external behavior features when the underlying behavior pattern is given. It describes the relationship between the SOF and hidden SSF. In order to make the model tractable, we subject the model to the following constraints: the hidden state (i.e., the underlying behavior pattern) of a node is affected and determined by the hidden states of its spatiotemporal neighbors, while a node's observation is only controlled by its current hidden state and is independent of the observations and hidden states of other nodes. This constraint has been widely adopted by the hidden Markov models; it has also been proved to be reasonable and effective for model simplification. Thus, the joint conditional probability $\Pr[o|s, \Omega]$ can be further calculated by the product shown in formula (3):

$$\Pr[o|s, \Omega] = \prod\nolimits_{t,k} \Pr[o_{t,k}|s_{t,k}, \Omega]. \tag{3}$$

Then, the Gaussian mixture model (GMM) is employed to formulize the local emission probability $\Pr[o_{t,k}|s_{t,k}, \Omega]$ of formula (3). Here we let $a_q(r)$ denote the local emission probability $\Pr[o_{t,k}|s_{t,k}, \Omega]$ and calculate it by equation (4):

$$\begin{aligned} a_q(r) &= \Pr[O_{t,k} = r | S_{t,k} = q] \\ &= \sum_{n=1}^{N} w_{n,q} \Pr[O_{t,k} = r | S_{t,k} = q, \theta_{n,q}] \\ &= \sum_{n=1}^{N} w_{n,q} \frac{1}{\sqrt{2\pi\sigma_{n,q}^2}} \exp(-\frac{(r - \mu_{n,q})^2}{2\sigma_{n,q}^2}), \end{aligned} \tag{4}$$

where $w_{n,q} \geqslant 0$, $\sum_{n=1}^{N} w_{n,q} = 1$, the subscript n denotes the n^{th} Gaussian component. The parameter set is $\theta_{n,q} = (\mu_{n,q}, \sigma_{n,q}, w_{n,q})$ that includes the means $\mu_{n,q}$, the variances $\sigma_{n,q}^2$ and the weights $w_{n,q}$. $r \in \mathcal{O}$ and $q \in \mathcal{S}$ are the values of observation and state, respectively.

For the term of prior probability $\Pr[s|\Omega]$ in formula (2), it describes the joint probability of all NEs' behavior patterns within a given space-time area. To derive the solution for $\Pr[s|\Omega]$, we first define the local prior probability as:

$$b_{t,k}(q) = \Pr[S_{t,k} = q | s_{\tau,\kappa}, \forall v_{\tau,\kappa} \in \mathcal{V} - \{v_{t,k}\}, \lambda], \tag{5}$$

where $\mathcal{V} - \{v_{t,k}\}$ is the NEs excluding the $v_{t,k}$ and $q \in \mathcal{S}$. The $b_{t,k}(q)$ represents the probability of the state q taken by the NE $v_{t,k}$ when the states of the remaining NEs are given. It essentially describes the spatiotemporal context of each NE's

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2020.3036664, IEEE Transactions on Dependable and Secure Computing

behavior pattern, i.e., the traffic forwarding behavior pattern of each NE is related to all other NEs due to the network's connectivity. Existing research shows that the influence of nodes decreases with propagation in a network. Based on this property, in this work we only consider the one-hop spatiotemporal context and ignore the influence of multi-hop neighbors. Then the local prior probability $b_{t,k}(q)$ is simplified by formula (6):

$$b_{t,k}(q) \approx \Pr[S_{t,k} = q | s_{\mathbb{N}^S_{t,k}}, s_{\mathbb{N}^T_{t,k}}, \lambda], q \in \mathcal{S}, \tag{6}$$

where $\mathbb{N}^T_{t,k}$ and $\mathbb{N}^S_{t,k}$ denote the corresponding one-hop temporal and spatial neighbors of $v_{t,k}$, respectively. Actually, formula (6) shows the first-order spatiotemporal Markovianity of the hidden SSF. Based on the Hammersley-Clifford theorem [15], the local prior probability $b_{t,k}(q)$ is equivalent to the Gibbs distribution that can be calculated by:

$$b_{t,k}(q) = \exp(-U_{t,k}(q|\lambda))/Z_{t,k}(\lambda), q \in \mathcal{S}, \tag{7}$$

where λ denotes the parameter set used to describe the interaction between the neighboring NEs. $Z_{t,k}(\lambda) = \sum_{q \in \mathcal{S}} \exp(-U_{t,k}(q))$ and $U_{t,k}(q)$ are the marginal partition function and marginal energy function, respectively. The $U_{t,k}(q)$ is calculated by formula (8):

$$U_{t,k}(q) = \varepsilon_{t,k} \sum\nolimits_{v_{\tau,\kappa} \in \{\mathbb{N}^S_{t,k}, \mathbb{N}^T_{t,k}\}} V_{t,k}(q, s_{\tau,\kappa}), \tag{8}$$

where $\varepsilon_{t,k} = 1/(|\mathbb{N}^S_{t,k}| + |\mathbb{N}^T_{t,k}|)$ denotes the normalized factor of node energy used to eliminate the influence of the number of neighbors in space and time on the calculation of energy function. $|\mathbb{N}^S_{t,k}|$ and $|\mathbb{N}^T_{t,k}|$ are the number of spatial and temporal neighbors of $v_{t,k}$, respectively. Here the Potts model [33] is used to calculate the partition function. Let $V_{t,k}(q, s_{\tau,\kappa})$ denote the potential function of the two neighboring NEs. Thus, for the second order neighborhood system the potential function is defined by

$$V_{t,k}(q, s_{\tau,\kappa}) = \begin{cases} 0, & (s_{\tau,\kappa} = q) \\ \beta, & (s_{\tau,\kappa} \neq q) \end{cases}, v_{\tau,\kappa} \in \{\mathbb{N}^S_{t,k}, \mathbb{N}^T_{t,k}\}, \tag{9}$$

where β denotes the parameter associated with the pairwise interactions between two NEs.

Directly evaluating the prior probability $\Pr[s|\Omega]$ shown in formula (2) is prohibitive even for problems of moderate size, since there are a combinatorial number of NEs in $\mathcal{V}$ for a state set $\mathcal{S}$. To make $\Pr[s|\Omega]$ solvable, we utilize pseudolikelihood [34] to replace it approximately. Thus, combining the formulas (5) and (6), the $\Pr[s|\Omega]$ can be calculated by:

$$\begin{aligned} \Pr[s|\Omega] &\simeq \prod_{v_{t,k} \in \mathcal{V}} \Pr[s_{t,k} | s_{\tau,\kappa}, \forall v_{\tau,\kappa} \in \mathcal{V} - \{v_{t,k}\}, \lambda] \\ &\simeq \prod_{v_{t,k} \in \mathcal{V}} \Pr[s_{t,k} | s_{\mathbb{N}^S_{t,k}}, s_{\mathbb{N}^T_{t,k}}, \lambda] \\ &= \prod_{v_{t,k} \in \mathcal{V}} b_{t,k}(s_{t,k}). \end{aligned} \tag{10}$$

Based on the above derivation, we define the model's parameter set as $\Omega = \{\mu_{n,q}, \sigma_{n,q}, w_{n,q}, \beta\}, n \in \{1, ..., N\}, q \in \mathcal{S}$. The behavior pattern inference algorithm and model learning algorithm will be introduced in the following.

Algorithm 1 SSF Inference Algorithm

1: **function** SSF(o, Ω)
2: **Initialize** : $s^{(0)}$;
3: **for all** $v_{t,n} \in \mathcal{V}$ **do**
4: **for all** $q \in \mathcal{S}$ **do**
5: $a_q(r) = \sum_{n=1}^{N} w_{n,q} \Pr[O_{t,k} = r | S_{t,k} = q, \theta_{n,q}]$;
6: $b_{t,k}(q) = \Pr[S_{t,k} = q | s_{\mathbb{N}^S_{t,k}}, s_{\mathbb{N}^T_{t,k}}, \lambda]$;
7: $\xi_{t,k}(q) = a_q(r) b_{t,k}(q)$;
8: **end for**
9: $s_{t,k} \leftarrow \arg\max_{q \in \mathcal{S}} \xi_{t,k}(q)$;
10: **end for**
11: $\forall v_{t,k} \in \mathcal{V} : \hat{s}_{t,k} \leftarrow s_{t,k}$;
12: **return** $\hat{s}$;
13: **end function**

C. Behavior patterns inference

The essence of behavior patterns inference is to estimate the underlying SSF given the parameters of the model. It is based on the MAP criterion and formula (1). **Algorithm 1** shows the pseudocode of our iterative algorithm.

The input is the observations o and parameters of model Ω. The output is the optimal state field $\hat{s}$ to be estimated. In the initialization process (the 2^{nd} line), $s^{(0)}$ is obtained by the prior knowledge on SOF and SSF, or by the clustering methods, e.g., KMeans. For each NE (the 3^{rd} line), the algorithm traverses all the potential states (the 4^{th} line), then the state with the maximum probability is chosen as the optimal result (the 9^{th} line). Note that the probability of a potential state includes two parts based on formula (2): the first part shown in the 5^{th} line is the local likelihood probability $a_q(r)$ given by formula (4); the second part shown in the 6^{th} line is the partial prior probability $b_{t,k}(q)$ given by formula (7). The algorithm estimates the SSF by the iterated conditional mode (ICM) [15] that maximizes local conditional probabilities sequentially. This makes the behavioral pattern inference algorithm independent of the network size and more flexible.

Since each state represents a specific behavior pattern associated with the ongoing network event encountered by an NE, the DTE detection is equivalent to inferring the hidden SSF. In addition, the SSF also provides a global view of the network event from the network-side perspective.

D. Model Learning

Similar to most machine learning-based applications, model learning is required before it is applied. The training data are the historical observations collected from the distributed network. Because the SSF is usually unknown/unlabelled, the model learning has to adopt an unsupervised learning method. The EM algorithm [35] is a classic method to find maximum likelihood parameters of a statistical model when the equations cannot be solved directly. For the above double-layer random field model, the Q function is defined by

$$Q(\Omega|\Omega^{(l)}) = E_s\{\ln \Pr[o, s|\Omega] | o, \Omega^{(l)}\}, \tag{11}$$

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2020.3036664, IEEE Transactions on Dependable and Secure Computing

where $\Omega^{(l)}$ and Ω denote the parameter sets obtained in the l^{th} iteration and to be estimated in the $(l+1)^{th}$ iteration, respectively. The learning algorithm consists of two steps for each iteration:

- The E-step: calculating the $Q(\Omega|\Omega^{(l)})$;
- The M-step: finding the optimal parameter set by $\Omega^{(l+1)} = \arg\max_{\Omega} Q(\Omega|\Omega^{(l)})$;
- $l = l+1$, repeating the above two steps until meeting the condition of convergence.

A computable form of the Q function is shown in formula (12), where $T_1(\mu_{n,q}^{(l+1)}, \sigma_{n,q}^{(l+1)}, w_{n,q}^{(l+1)})$ and $T_2(\beta^{(l+1)})$ denote the first term and the second term on the right-side of the third equal sign, respectively. Then, the model's parameters can be estimated by maximizing $T_1(\mu_{n,q}^{(l+1)}, \sigma_{n,q}^{(l+1)}, w_{n,q}^{(l+1)})$ and $T_2(\beta^{(l+1)})$ independently since they are not related.

$$\begin{aligned} &Q(\Omega|\Omega^{(l)}) = E_s\{\ln \Pr[o,s|\Omega]|o,\Omega^{(l)}\} \\ &= \sum_{s\in\mathbb{S}} \Pr[s|o,\Omega^{(l)}] \cdot \ln \Pr[o|s,\Omega] \Pr[s|\Omega] \\ &= \sum_{q\in\mathcal{S}} \sum_{t,k} \Pr[q|o_{t,k},\Omega^{(l)}] \cdot \ln\{\sum_{n=1}^{N} w_{n,q} \Pr[o_{t,k}|q,\Omega]\} + \\ &\quad \sum_{q\in\mathcal{S}} \sum_{t,k} \Pr[q|o_{t,k},\Omega^{(l)}] \cdot \ln \Pr[S_{t,k}=q|\Omega] \\ &= T_1(\mu_{n,q}^{(l+1)}, \sigma_{n,q}^{(l+1)}, w_{n,q}^{(l+1)}) + T_2(\beta^{(l+1)}) \end{aligned} \tag{12}$$

The parameters $\Omega = \{\mu_{n,q}, \sigma_{n,q}, w_{n,q}\}, n \in \{1,...,N\}, q \in \mathcal{S}$ in the term T_1 can be estimated by solving the following differential equations:

$$\begin{cases} \frac{\partial}{\partial \mu_{n,q}^{(l+1)}}[T_1(\mu_{n,q}^{(l+1)}, \sigma_{n,q}^{(l+1)}, w_{n,q}^{(l+1)})] = 0 \\ \frac{\partial}{\partial \sigma_{n,q}^{(l+1)}}[T_1(\mu_{n,q}^{(l+1)}, \sigma_{n,q}^{(l+1)}, w_{n,q}^{(l+1)})] = 0 \\ \frac{\partial}{\partial w_{n,q}^{(l+1)}}[T_1(\mu_{n,q}^{(l+1)}, \sigma_{n,q}^{(l+1)}, w_{n,q}^{(l+1)})] = 0. \end{cases} \tag{13}$$

Then, the parameters are calculated by the following formulas:

$$\begin{cases} \mu_{n,q}^{(l+1)} = \frac{\sum_{t,k} \gamma_{t,k}^{(l)}(n,q) o_{t,k}}{\sum_{t,k} \gamma_{t,k}^{(l)}(n,q)} \\ (\sigma_{n,q}^{(l+1)})^2 = \frac{\sum_{t,k} \gamma_{t,k}^{(l)}(n,q)(o_{t,k} - \mu_{n,q}^{(l+1)})^2}{\sum_{t,k} \gamma_{t,k}^{(l)}(n,q)} \\ w_{n,q}^{(l+1)} = \frac{\sum_{t,k} \gamma_{t,k}^{(l)}(n,q)}{\sum_{t,k} \sum_{n=1}^{N} \gamma_{t,k}^{(l)}(n,q)}, \end{cases} \tag{14}$$

where $\gamma_{t,k}^{(l)}(n,q)$ is calculated by formula (15):

$$\gamma_{t,k}^{(l)}(n,q) = [\frac{\Pr^{(l)}[q|o_{t,k}]}{\Pr^{(l)}[o_{t,k}]}][\frac{w_{n,q}^{(l)} \Pr[o_{t,k}|q,\theta_{n,q}^{(l)}]}{\sum_{n=1}^{N} w_{n,q}^{(l)} \Pr[o_{t,k}|q,\theta_{n,q}^{(l)}]}]. \tag{15}$$

In this formula, the $\Pr^{(l)}[q|o_{t,k}]$ is calculated by formula (16):

$$\overset{(l)}{\Pr}[q|o_{t,k}] = \frac{\Pr[o_{t,k}|q] \cdot \Pr^{(l)}[S_{t,k}=q|s_{\mathbb{N}_{t,k}^S}^{(l)}, s_{\mathbb{N}_{t,k}^T}^{(l)}]}{\Pr^{(l)}[o_{t,k}]}, \tag{16}$$

Algorithm 2 Model Learning Algorithm

1: **function** Learning(o)
2: **Initialize** : $l \leftarrow 0, \hat{\Omega}^{(0)} \leftarrow \{\mu_{n,q}^{(0)}, \sigma_{n,q}^{(0)}, w_{n,q}^{(0)}, \beta^{(0)}, n \in \{1,...,N\}, q \in \mathcal{S}\}, \mathcal{L}^{(l)} \leftarrow 0, C_{em}$;
3: **repeat**
4: $\hat{s}^{(l)} \leftarrow$ SSF($o, \hat{\Omega}^{(l)}$);
5: Update $\{\hat{\mu}_{n,q}, \hat{\sigma}_{n,q}, \hat{w}_{n,q}\}$ based on formula (14);
6: Update $\{\hat{\beta}\}$ based on formula (18);
7: $l \leftarrow l+1$;
8: $\hat{\Omega}^{(l)} \leftarrow \{\hat{\mu}_{n,q}, \hat{\sigma}_{n,q}, \hat{w}_{n,q}, \hat{\beta}\}$;
9: $\mathcal{L}^{(l)} \leftarrow \sum_q \sum_{t,k} \ln a_q(o_{t,k}|\hat{\Omega}^{(l)}) b_{t,k}(q|\hat{s}^{l-1}, \hat{\Omega}^{(l)})$;
10: **until** $|\mathcal{L}^{(l)} - \mathcal{L}^{(l-1)}| \leq C_{em}$
11: $\hat{\Omega} \leftarrow \hat{\Omega}^{(l)}$;
12: **return** $\hat{\Omega}$;
13: **end function**

where $s_{\mathbb{N}_{t,k}^S}^{(l)}$ and $s_{\mathbb{N}_{t,k}^T}^{(l)}$ denote the states of the spatial and temporal neighbors of $v_{t,n}$ in the l^{th} iteration, respectively. The likelihood $\Pr^{(l)}[o_{t,k}]$ has the form:

$$\overset{(l)}{\Pr}[o_{t,k}] = \sum_{q\in\mathcal{S}} \Pr[o_{t,k}|q] \cdot \overset{(l)}{\Pr}[S_{t,k}=q|s_{\mathbb{N}_{t,k}^S}^{(l)}, s_{\mathbb{N}_{t,k}^T}^{(l)}]. \tag{17}$$

In contrast to most of the MRF-based works that ignore the state field parameters or directly specify its value through artificial experience [36], here we estimate the value of β via the learning algorithm without manual intervention:

$$\frac{\partial}{\partial \beta^{(l+1)}}[T_2(\beta^{(l+1)})] = 0. \tag{18}$$

It should be noted that there is no close-form solution for β. Thus, β can only be calculated by numerical solutions, e.g., Newton's Method [37] or gradient descent [38].

The pseudocode of the Model Learning Algorithm is shown in **Algorithm 2**. The input is the historical observations o collected from the distributed scenarios, i.e. the training data of the model. The output is the parameter set $\hat{\Omega}$ of the model. In the initialization process (the 2^{nd} line), the initialization model parameters $\hat{\Omega}^{(0)}$ can be obtained by the prior knowledge of SOF and SSF. The state field $\hat{s}^{(l)}$ in the l^{th} iteration can be inferred according to the training data o and the l^{th} iteration parameter set $\hat{\Omega}^{(l)}$ via **Algorithm 1** (the 4^{th} line). $\{\hat{\mu}_{n,q}, \hat{\sigma}_{n,q}, \hat{w}_{n,q}\}$ are updated (the 5^{th} line) based on formula (14), while $\{\hat{\beta}\}$ is updated (the 6^{th} line) based on formula (18). The logarithmic likelihood $\mathcal{L}^{(l)}$ is calculated for each iteration (the 9^{th} line). If the difference between the logarithmic likelihoods calculated by two adjacent iterations is less than the algorithm convergence condition C_{em}, then the iteration is stopped (the 10^{th} line) and the parameter set $\hat{\Omega}$ of the model is generated. C_{em} in the algorithm denotes the given convergence condition for the iteration of the algorithm. To control the iteration process of the algorithm, we let $L = \Pr[o|\Omega] = \sum_{s\in\mathbb{S}} \Pr[o,s|\Omega]$ denote the overall likelihood that measures the fitting degree of the model to the training

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2020.3036664, IEEE Transactions on Dependable and Secure Computing

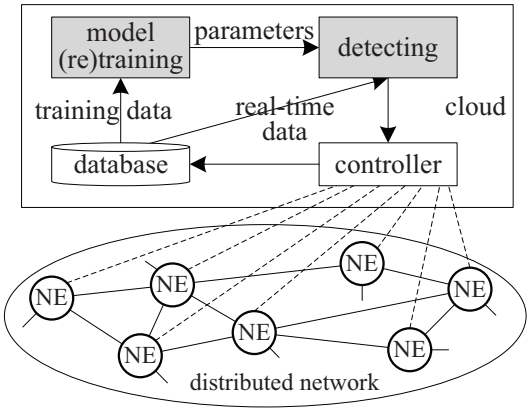


Fig. 2: An application instance.

data. Then, the logarithmic likelihood $\mathcal{L}$ is defined by

$$\begin{aligned}\mathcal{L} &= \sum\nolimits_{s\in\mathbb{S}} \ln \Pr[o, s|\Omega] \\ &\simeq \sum\nolimits_{q\in\mathcal{S}} \sum\nolimits_{t,k} \ln \Pr[o_{t,k}, q|s_{\mathbb{N}^S_{t,n}}, s_{\mathbb{N}^T_{t,n}}, \Omega] \\ &= \sum\nolimits_{q\in\mathcal{S}} \sum\nolimits_{t,k} \ln[a_q(o_{t,k}) b_{t,k}(q)].\end{aligned} \tag{19}$$

E. An application instance

Figure 2 shows an application instance for the proposed network-centric DTE detection scheme. The entire system consists of two parts: the cloud and the distributed network. The cloud is responsible for the core functions of the proposed scheme, including data storage, model training/retraining, detection and interaction with NEs. It is loosely coupled with the application scenario. The distributed network consists of interconnected NEs, its definition and implementation may differ in different scenarios. For example, when it is applied to a packet switching network, an NE can be integrated into any physical or virtual network relay device via the network function virtualization (NFV) technologies, e.g., router, switch and gateway. Similarly, the connection of adjacent NEs can be a physical or virtual link. When it is applied to a logical network, such as the social network, an NE is the client software of the social network installed on the user's terminal device. The main function of NEs is the network measurement and local detection. The following are the details of each function module:

NE: (i) each NE measures the required data and feeds them back to the controller; (ii) each NE periodically exchanges the states with its neighbors; (iii) each NE evaluates its own hidden state based on the states of its spatiotemporal neighbors.

Controller: (i) it receives the measurement data from the NEs and forwards them to the database; (ii) it implements unmanned intelligent management according to the detection result, e.g., adjusting the working mode of the NEs, resource rescheduling and starting the emergency measures.

Database: it is responsible for data storage.

Training: (i) it implements the model training based on **Algorithm 2** and historical data; (ii) it dynamically updates the model's parameters via periodic re-training or on-line update algorithms [39].

Detecting: it uses the trained model to detect the DTEs based on **Algorithm 1** and releases the results to the controller.

IV. Experiment

In this section, we evaluate the performance of the proposed spatiotemporal context approach (STCA) for the DTE detection by two independent experiments.

A. General experimental information

Scenarios. The experiment scenarios include a DDoS attack detection in an IP network and the Short Message Service (SMS) worm detection in a Social Network (SN). The simulation is designed according to the instance shown in Fig. 2.

Baselines. Because VCAs focus on the single-point protection rather than distributed networks, we do not compare with this type of methods in the experiment. The essence of most existing CAD-based approaches is multi-point monitoring rather than considering the holistic time-varying behavior of the distributed networks, which allows them to be regarded as a special case or a simplified version of the proposed approach, i.e., most of them can be derived by the proposed approach when it only considers the observation features of some sampling nodes and ignores the spatiotemporal context. Thus, we did not compare these methods with ours one by one, but only used the KMeans algorithm (KMeans) [27] and the Gaussian mixture models (GMM) [28] as a representative for the performance comparison. Moreover, considering that the proposed approach in this work is an unsupervised learning method, we employ three other unsupervised event detection schemes for the performance evaluation[3]. These schemes were all designed for security detection and published in recent years, including Birch algorithm (Birch) [40], Ward agglomerative hierarchical approach (Ward) [41], and Kernel-based fuzzy c-means algorithm (KFCM) [42].

Evaluation metrics. We use the Accuracy, Macro-F1 score and False Positive Rate (FPR) as the evaluation metrics that are defined by formulas (20), (21) and (22) respectively:

$$Accuracy = \frac{number\ of\ correct\ detections}{number\ of\ detections}, \tag{20}$$

$$Macro\ F1 = \frac{1}{N}\sum\nolimits_{i=1}^{N} F1_i, \tag{21}$$

$$FPR = \frac{1}{N}\sum\nolimits_{i=1}^{N} FPR_i, \tag{22}$$

where N denotes the number of event-related behavior patterns. The subscript i denotes the i^{th} pattern. $F1_i$ and FPR_i are calculated by formulas (23) and (24), respectively:

$$F1_i = \frac{2\cdot(precision_i \cdot recall_i)}{(precision_i + recall_i)}, i = 1, 2, ..., N \quad , \tag{23}$$

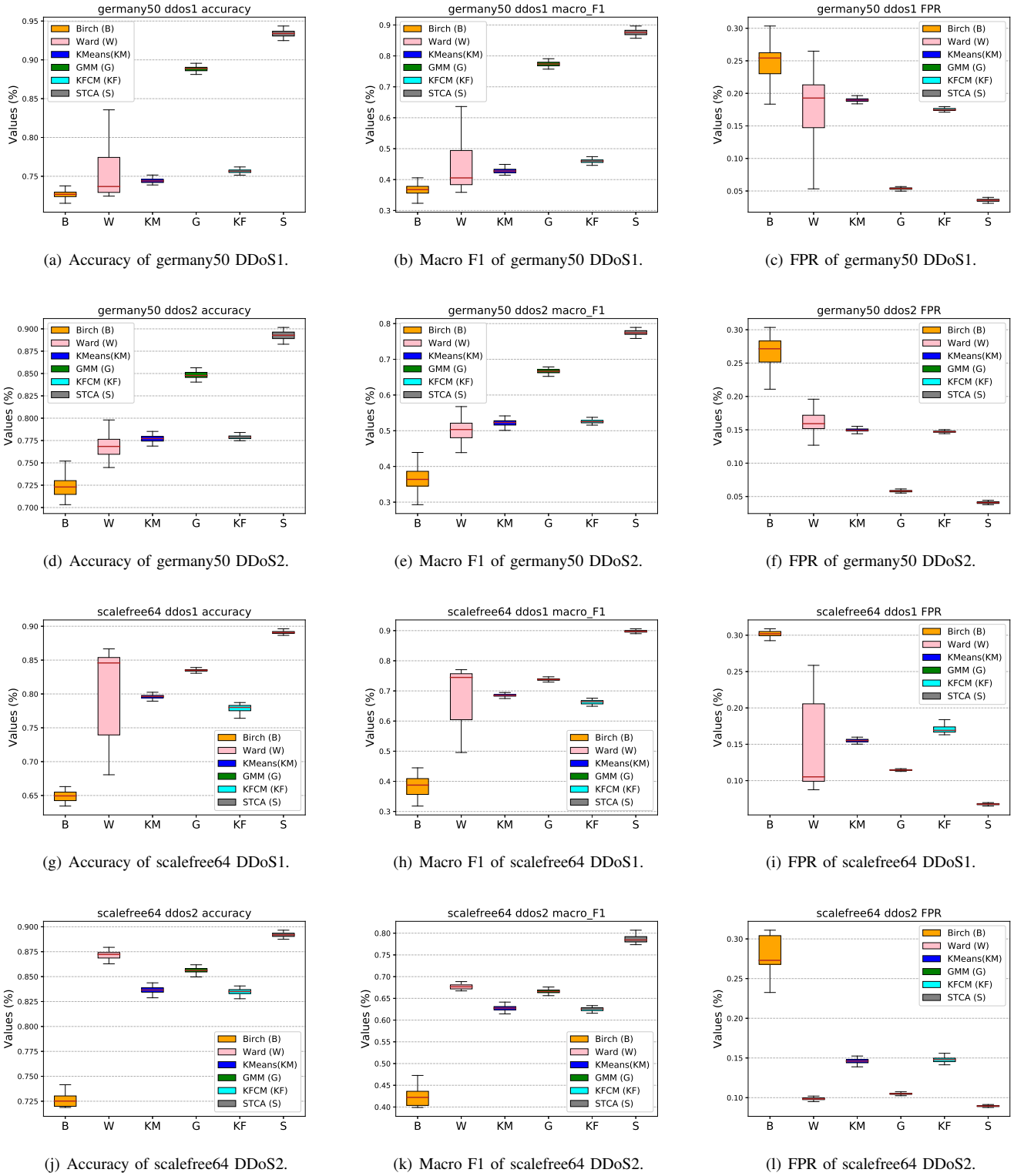
$$FPR_i = \frac{falsepositive_i}{negative_i}, i = 1, 2, ..., \quad , \tag{24}$$

where $precision_i$ and $recall_i$ denote the precision and recall of the i^{th} pattern, respectively. Thus, the $F1$ score can be regarded as a weighted average of the precision and recall.

[3]We do not use deep NN based approaches for performance evaluation because they are usually only applicable to supervised scenarios rather than the unsupervised.

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2020.3036664, IEEE Transactions on Dependable and Secure Computing



(a) Accuracy of germany50 DDoS1. (b) Macro F1 of germany50 DDoS1. (c) FPR of germany50 DDoS1.

(d) Accuracy of germany50 DDoS2. (e) Macro F1 of germany50 DDoS2. (f) FPR of germany50 DDoS2.

(g) Accuracy of scalefree64 DDoS1. (h) Macro F1 of scalefree64 DDoS1. (i) FPR of scalefree64 DDoS1.

(j) Accuracy of scalefree64 DDoS2. (k) Macro F1 of scalefree64 DDoS2. (l) FPR of scalefree64 DDoS2.

Fig. 3: Accuracy, Macro-F1 and FPR of the Internet Scenarios.

In formula (24), $falsepositive_i$ and $negative_i$ represent the number of false positives and the total number of negatives of the i^{th} pattern, respectively.

Simulation execution environment. The simulation is developed by MATLAB R2015b and run on a general computer configured with Intel Core i7 CPU at 3.60GHz and 32G RAM[4].

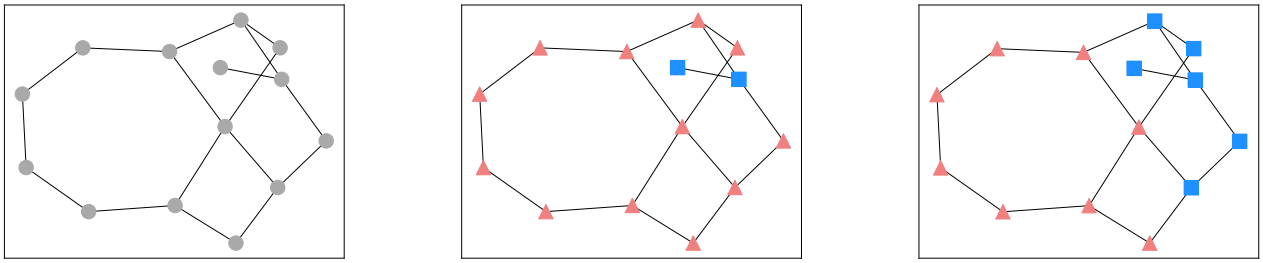
[4]The code and data can be found on the github (https://github.com/SYSUNetlab/DET2019) for academic research purposes.

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2020.3036664, IEEE Transactions on Dependable and Secure Computing

TABLE I: Performance comparison in the Internet scenario.

	Metrics (avg $\pm\sigma$)	Birch (%)	Ward (%)	KMeans (%)	GMM (%)	KFCM (%)	STCA (%)
germany50 ddos1	Accuracy	72.69 ± 0.67	76.69 ± 5.54	74.43 ± 0.27	88.85 ± 0.35	75.65 ± 0.23	93.39 ± 0.45
	Macro F1	36.8 ± 2.35	47.6 ± 13.54	42.81 ± 0.71	77.42 ± 0.74	45.97 ± 0.6	87.55 ± 0.9
	FPR	24.67 ± 2.7	17.29 ± 5.92	18.97 ± 0.26	5.37 ± 0.16	17.51 ± 0.21	3.58 ± 0.21
germany50 ddos2	Accuracy	72.38 ± 1.08	77.21 ± 2.23	77.69 ± 0.35	84.82 ± 0.41	77.87 ± 0.19	89.23 ± 0.61
	Macro F1	36.76 ± 3.57	50.84 ± 4.61	52.19 ± 0.85	66.61 ± 0.64	52.55 ± 0.45	77.35 ± 1.28
	FPR	26.16 ± 3.29	15.66 ± 2.74	14.95 ± 0.23	5.82 ± 0.14	14.71 ± 0.13	4.11 ± 0.22
scalefree64 ddos1	Accuracy	64.86 ± 0.71	80.82 ± 6.59	79.6 ± 0.29	83.47 ± 0.22	77.85 ± 0.66	89.08 ± 0.25
	Macro F1	38.28 ± 3.02	69.19 ± 9.6	68.52 ± 0.51	73.74 ± 0.41	66.17 ± 0.96	89.78 ± 0.41
	FPR	30.2 ± 0.39	14.17 ± 6.06	15.51 ± 0.22	11.46 ± 0.1	17.07 ± 0.58	6.76 ± 0.11
scalefree64 ddos2	Accuracy	73.04 ± 2.05	84.73 ± 5.3	83.65 ± 0.28	85.62 ± 0.26	83.42 ± 0.48	89.16 ± 0.33
	Macro F1	43.08 ± 4.26	63.57 ± 8.85	62.76 ± 0.51	66.62 ± 0.39	62.42 ± 0.72	78.46 ± 1.38
	FPR	27.6 ± 3.05	12.85 ± 6.26	14.59 ± 0.27	10.49 ± 0.11	14.85 ± 0.53	8.96 ± 0.14



(a) Time 1. (b) Time 2. (c) Time 3.

Fig. 4: Spatiotemporal detection result in a part of germany50 topology (the grey circle, the peach triangle, and the blue rectangle denote the State1, State2 and State3, respectively).

B. DDoS attack detection in Internet

Different from the traditional VCAs for DDoS attack detection, our scheme applies the spatiotemporal context approach to detect the DDoS attack activities from the view of the network-side. It needs the data collected from a distributed network, especially the forwarding logs of the relay devices like routers and switches. However, as far as we know, currently no public data sources can meet this requirement. To make the experiment credible and reproducible, we adopted a compromised approach in which real network traffic is replayed in the simulated topologies. The simulation details are as follows.

Topology. We adopted two topologies in this experiment. The first one is a real network topology from a German research network that consists of 50 nodes and 88 links [43]. It is denoted by "germany50" in the following. More real network topologies can be found from the Internet Topology Zoo[5]. The second one is generated based on a scale-free model that is widely considered to be one of the most similar models to the real Internet [44]. The generated scale-free topology contains 64 nodes and 125 links. We denote it by "scalefree64" in the following. In order to replay the real network traffic in the above topologies, we mount virtual terminals on the NEs. These terminals are used to replay and receive the traffic. The forwarding strategy of each NE is based on the shortest path algorithm. We develop an independent node to implement the functions of the modules of the cloud shown in Fig. 2, including the controller, database, (re)training, and detecting.

Traffic. To make the experiment reproducible, all replayed traffic is from real world data. It consists of two parts: background traffic and attack traffic. The background traffic comes from a campus network[6] and the open traffic archives of MAWI[7]. This traffic lasts about three days and shows significant daily variation characteristics. The attack traffic comes from the "DDoS Attack 2007" provided by the open traffic database of CAIDA[8]. Two types of DDoS attacks are considered for the experiment: "ddos1" is a direct attack that attacks the victim's network bandwidth resource, while "ddos2" is an SYN flood attack that attacks the victim's system resource. Some terminals are randomly selected as attack nodes that are responsible for the replay of the attack traffic in addition to the background traffic. Considering all raw traffic data of both CAIDA and MAWI are collected from a single monitoring point instead of the NEs of a distributed network, we assign the traffic to the terminals based on the address prefixes of its source and destination addresses, i.e., each terminal is bound to a fixed network prefix.

We define three discrete states {State1, State2, State3} to denote "normal pattern", "suspected pattern" and "attack pattern" for each NE' behavior pattern, respectively. The traffic features used for modeling include the entropy of the destination IP address and the arrival rate of packets per flow

[5]http://topology-zoo.org/

[6]http://www.sysu.edu.cn

[7]http://mawi.wide.ad.jp/mawi/

[8]http://www.caida.org/home/

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2020.3036664, IEEE Transactions on Dependable and Secure Computing

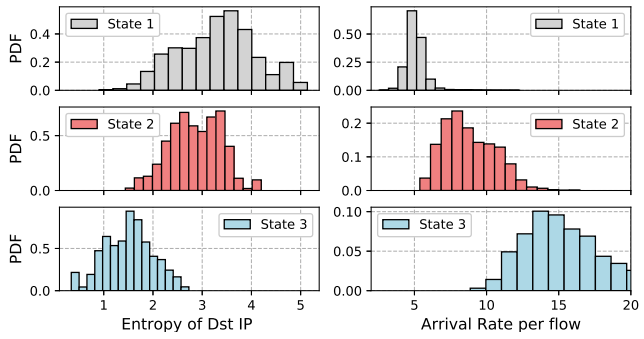


Fig. 5: State distribution of germany50 ddos1.

that have been widely used in the traditional VCAs for DDoS attack detection [45].

We run each of the algorithms for ten times with ten-fold cross-validation, i.e., 100 trials for each algorithm. Fig. 3 shows the Accuracy, Macro F1 and FPR of the six schemes in four scenarios, including germany50 DDoS1, germany50 DDoS2, scalefree64 DDoS1 and scalefree64 DDoS2. The baseline methods consist of Birch(B), Ward(W), KMeans(KM), GMM(G) and KFCM(KF). We use the box plots to indicate the degree of dispersion (spread), skewness and median for the metrics. As the figures show, the results of the proposed STCA are very concentrated for all metrics. From the perspective of the median, the STCA consistently and significantly outperforms all baselines in four cases.

Table I includes the average values and the confidence intervals (i.e. the standard deviation) for Accuracy, Macro F1 and FPR for the five baseline methods. In the four cases, Accuracy and Macro F1 of the STCA are better than other algorithms, while the FPR of the STCA is lower than other algorithms. The gain of the STCA is mainly benefited from the combination of an NE's observation and the state information of its spatiotemporal neighbors. The result is encouraging as both the performance and the stability of the proposed approach are better than the others for all metrics and scenarios.

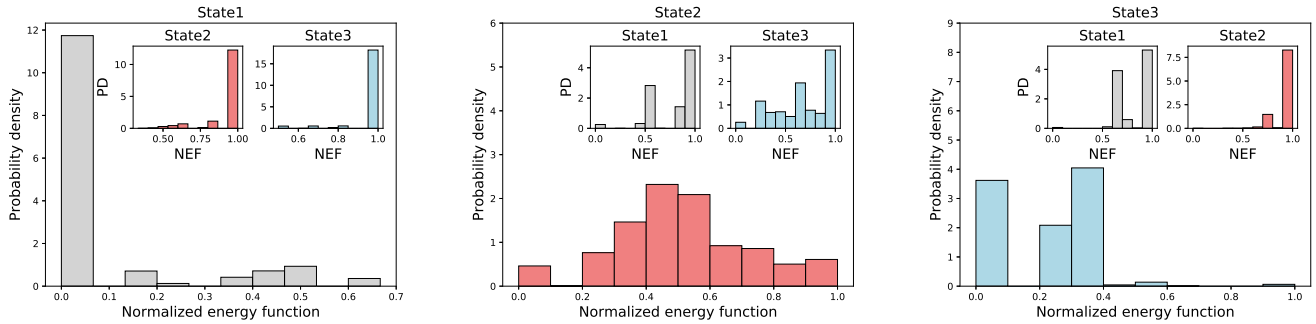
Fig. 4 shows the spatiotemporal detection result of the STCA in a part of the germany50 topology. The gray circle denotes the "normal" state that implies the normal working pattern of the NEs. The peach triangle denotes the "suspected attack" state, which means the behavior pattern of the NE deviates from the normal range, but has not reached a significant degree of abnormality. Based on the experiments, this type of state mainly appears near the source of the attack or on the attack paths. The blue rectangle denotes the "under attack" state; it usually refers to the NEs close to the victim or the victim itself. The result shows that the STCA can provide the macroscopic global perspective of the network during a DDoS attack, which indicates the states of the NEs along the attack path or near the victim varying with the development of the attack event. It is valuable for the early intelligent response to emergencies, because the temporal and spatial distribution of the state changes can reveal some key attributes of the ongoing attack event. For example, the spatial distribution of the state transitions of the nodes shows the scale and direction of the attack event. Transition from State1 to State2 can indicate the network location of the attack source, while the transition from State2 to State3 may imply the potential attack direction and target. Similarly, the time-varying process of the spatial distribution of the state transitions can be used to estimate the speed and intensity of the ongoing attack event. This information can further trigger the management system to automatically intervene in the forwarding behavior of the network, including migrating attack targets, modifying forwarding strategies, trapping attack traffic, etc. Moreover, the spatiotemporal distribution of state transitions can be used to further analyze and mine the knowledge of the attack events, which can be fed back to the management system to improve the subsequent detection performance.

Fig. 5 shows the relationship between the observation distribution and the hidden states of the germany50 ddos1. Other cases are similar and are not shown here due to space limitation. The left column indicates the probability density of the information entropy of the destination IP address in the three states. The right column indicates the probability density of the arrival rate of packets per flow in the three states. In normal scenarios, the destination addresses of traffic are relatively uniform, while the arrival rate fluctuates around a relatively stable level. Therefore, the distributions of the entropy and the arrival rate are located at a relatively high area and a relatively low position, respectively, e.g., State1. In the attack scenario, the traffic increases and its destination addresses become concentrated, which causes the distributions of the entropy and arrival rate to move to small and large regions, respectively, e.g., State3. Moreover, in Fig. 5 the distributions of the states overlap each other, which indicates that an observation feature falling in the overlapping area may come from different states. For example, in the normal scenario (State1) a news hot spot that suddenly appears on a server will attract a lot of visits and traffic, which will lead to a phenomenon similar to an attack scenario (State3). Refining the definition of the states may help improve the results, but it cannot eliminate the issue. Therefore, if the model only depends on the observed features to decide the hidden states, it may make a wrong decision and lead to a high FPR and low accuracy, e.g., the baseline methods. This is one of the reasons why we develop the spatiotemporal MRF, because it allows the proposed approach to use both the observed features and the information of the spatiotemporal neighbors, which eventually improves its performance.

In Fig. 6 we investigate the relationship between the state selection and energy function of a node when the states of its neighbors are known. Here, we only take the results of the germany50 ddos1 as examples for analysis and omit the remaining due to the limited space. In the main diagram of Fig. 6(a), we calculate the normalized energy function (NEF) of all nodes that select State1 as the optimal state and show their statistical distribution. In order to explore the influence of a node's selection of different states on its energy function, we sequentially replace the optimal state (State1) with other states (State2 and State3) for those nodes shown in the main diagram, and display the corresponding distributions of their normalized energy functions in the embedded diagrams. We do similar processing in Fig. 6(b) and 6(c) where we analyze the

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2020.3036664, IEEE Transactions on Dependable and Secure Computing



(a) State1. (b) State2. (c) State3.

Fig. 6: Distribution of energy function for state selection in the Internet scenarios.

TABLE II: Performance comparision in the SN scenario.

	Metrics (avg $\pm\sigma$)	Birch (%)	Ward (%)	KMeans (%)	GMM (%)	KFCM (%)	STCA (%)
smallworld256 SI	Accuracy	80.38 $\pm$ 10.29	85.03 $\pm$ 5.71	88.37 $\pm$ 0.66	89.04 $\pm$ 0.55	88.17 $\pm$ 0.59	95.22 $\pm$ 1.89
	Macro F1	78.73 $\pm$ 10.62	84.1 $\pm$ 5.53	87.68 $\pm$ 1.24	88.3 $\pm$ 1.02	87.73 $\pm$ 0.86	93.06 $\pm$ 1.47
	FPR	19.3 $\pm$ 7.89	14.7 $\pm$ 3.98	11.27 $\pm$ 0.69	11.05 $\pm$ 0.61	11.32 $\pm$ 0.35	6.87 $\pm$ 2.07
smallworld256 SIS	Accuracy	79.14 $\pm$ 9.93	86.21 $\pm$ 4.74	88.89 $\pm$ 0.54	89.33 $\pm$ 0.54	88.68 $\pm$ 0.44	94.67 $\pm$ 1.29
	Macro F1	77.96 $\pm$ 9.9	85.5 $\pm$ 4.64	88.36 $\pm$ 0.91	88.74 $\pm$ 0.79	88.29 $\pm$ 0.57	93.34 $\pm$ 0.99
	FPR	19.8 $\pm$ 6.75	13.66 $\pm$ 3.33	10.86 $\pm$ 0.52	10.85 $\pm$ 0.55	10.93 $\pm$ 0.28	6.29 $\pm$ 1.05
scalefree256 SI	Accuracy	81.43 $\pm$ 8.39	85.98 $\pm$ 4.1	89.44 $\pm$ 0.43	89.63 $\pm$ 0.4	89.26 $\pm$ 0.43	94.65 $\pm$ 1.16
	Macro F1	80.03 $\pm$ 9.12	85.37 $\pm$ 4.08	89.02 $\pm$ 0.75	89.17 $\pm$ 0.63	89.02 $\pm$ 0.59	93.53 $\pm$ 1.19
	FPR	18.47 $\pm$ 7.43	13.8 $\pm$ 3.17	10.43 $\pm$ 0.42	10.46 $\pm$ 0.42	10.49 $\pm$ 0.28	6.53 $\pm$ 1.16
scalefree256 SIS	Accuracy	83.07 $\pm$ 5.39	85.89 $\pm$ 3.97	89.4 $\pm$ 0.45	89.4 $\pm$ 0.46	89.35 $\pm$ 0.32	94.66 $\pm$ 1.18
	Macro F1	82.28 $\pm$ 6.09	85.59 $\pm$ 4.19	89.31 $\pm$ 0.48	89.31 $\pm$ 0.49	89.32 $\pm$ 0.34	93.61 $\pm$ 0.98
	FPR	17.15 $\pm$ 5.51	13.97 $\pm$ 3.66	10.57 $\pm$ 0.47	10.58 $\pm$ 0.48	10.6 $\pm$ 0.3	6.31 $\pm$ 1.02

nodes whose optimal states are State2 and State3, respectively. From these results, we can see that the energy distributions of the optimal states prefer the low value region while the non-optimal states are in the high value region. This result is consistent with the purpose of the energy function: when the state selected by a node is consistent with its neighbors, the value of its energy function is very small; otherwise, the value of its energy function will increase with the degree of difference and reduce the probability of choosing the different states. This indicates that in the proposed approach, the spatiotemporal context of the state field can be used to decide the optimal hidden state of each node and is not limited to the external observation features, which partially solves the issue caused by the overlapping of the output distributions of the states and improves the accuracy of the event discrimination.

C. SMS worm detection in SN

In the second part of the experiments, we evaluate the performance of the proposed scheme via a worm propagation. Each NE corresponds to a social network account. When an NE gets infected, it sends the SMS spam to the others. The details of the simulation are as follows.

Topology. According to the previous analysis of real social networks [46], we generate two topologies for the simulation of a social network. The first one is based on a small-world model and contains 256 nodes and 512 links. We denote it by "smallworld256". The second one is based on a scale-free model, and contains 256 nodes and 509 links. We denote it by "scalefree256". Different from the first part of the experiments, there is no terminal here, and each NE in these topologies represents a social network user. In the initial stage, a small number of NEs are randomly selected as the sources of the worm.

propagation. The SMS worm propagation is controlled by a Susceptible-Infected (SI) model and a Susceptible-Infected-Susceptible (SIS) model, both of which are widely used in worm propagation dynamics analysis [47]. We adopt a linear probability of infections, i.e., an NE's infected probability is proportional to the number of its infected neighbors.

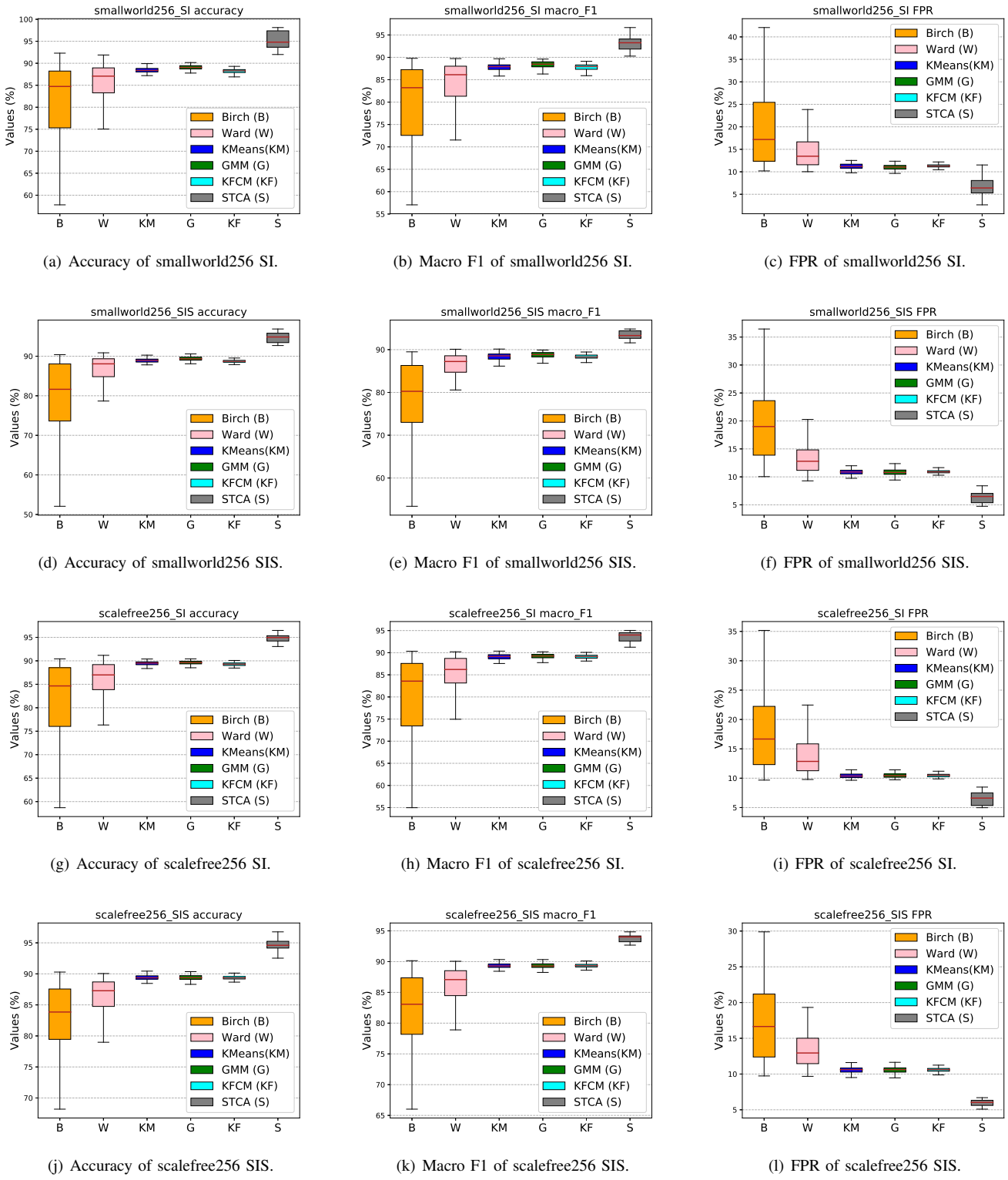
The "average SMS text length" is used as the observable behavior characteristic for the worm detection [48]. Two states are defined to denote the "susceptible" and "infected", respectively. The change in states reflects the propagation of the worm in the network.

Like the first part of the experiments, we run each of the algorithms for ten times ten-fold cross-validation. The experimental scenario includes smallworld256 SI, smallworld256 SIS, scalefree256 SI, and scalefree256 SIS. Fig. 7 shows the comparison of Accuracy, Macro F1 and FPR for the five baselines respectively, while Table II includes the average values and the confidence intervals. From the perspective of the data divergence, the performance of the proposed STCA is better than the Birth and Ward, and a little worse than the KMean, GMM and KFCM. However, for the median of the results, the STCA is significantly better than the others for all metrics.

Fig. 8 shows a part of the smallworld256 topology that visualizes the state field varying over time. The results indicate that the state field of the proposed scheme can be used to track and forecast the development trend of the worm propagation,

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2020.3036664, IEEE Transactions on Dependable and Secure Computing



(a) Accuracy of smallworld256 SI. (b) Macro F1 of smallworld256 SI. (c) FPR of smallworld256 SI.

(d) Accuracy of smallworld256 SIS. (e) Macro F1 of smallworld256 SIS. (f) FPR of smallworld256 SIS.

(g) Accuracy of scalefree256 SI. (h) Macro F1 of scalefree256 SI. (i) FPR of scalefree256 SI.

(j) Accuracy of scalefree256 SIS. (k) Macro F1 of scalefree256 SIS. (l) FPR of scalefree256 SIS.

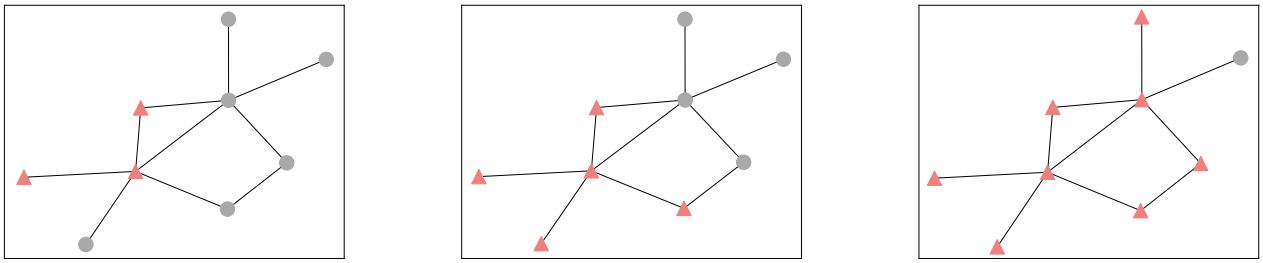
Fig. 7: Accuracy, Macro-F1 and FPR of the SN Scenarios.

which is valuable for an early threat response.

In Fig. 9, we take the smallworld256 SI as an example to show the relationship between the observation distribution for each hidden state. The other cases are not shown here due to space limitation. The results show that each state can be considered as a cluster of the observations. However, as there are no clear boundaries between the different clusters, the distribution of the observations corresponding to each state overlap with each other. Similar to the previous experiments, the proposed STCA compensates for the loss caused by the state overlapping through neighbor information, which makes it have better performance in the experiments. On the other

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2020.3036664, IEEE Transactions on Dependable and Secure Computing



(a) Time 1. (b) Time 2. (c) Time 3.

Fig. 8: Spatiotemporal detection result in a part of smallworld256 topology (the gray circle and the peach triangle denote the State1 and State2 respectively).

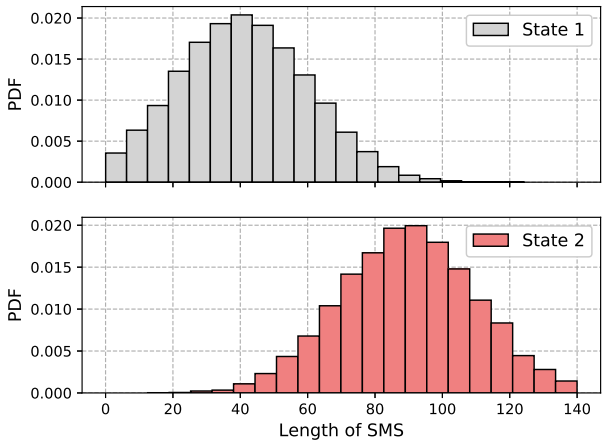


Fig. 9: State distribution of the smallworld256 SI.

hand, in order to verify the validity of the Gaussian distribution fitting, we perform the Jarque-Bera (JB) test [49] to verify whether the observations match a Gaussian distribution. For the sample data of State 1 and State 2, we compute the test statistic JB according to the skewness and the kurtosis of the samples. At the 5% significance level, the critical value for the test is 5.43. The test statistic JB is 1.45 and 0.8 for the samples of State 1 and State 2, respectively. Both of them are less than the critical value, thus the null hypothesis ("the data follow a Gaussian distributed") cannot be rejected, which indicates that the data fit a Gaussian distribution.

Fig. 10 shows the selection-probability of an NE when the states of its neighbors are given. The abscissa indicates the NEF, while the vertical axis denotes the probability density according to which a node selects the corresponding state. This result again shows that the working pattern of an NE is greatly affected by its neighbors. Based on this result, we can predict the future development trend of the network events through the state field, including the direction and the speed.

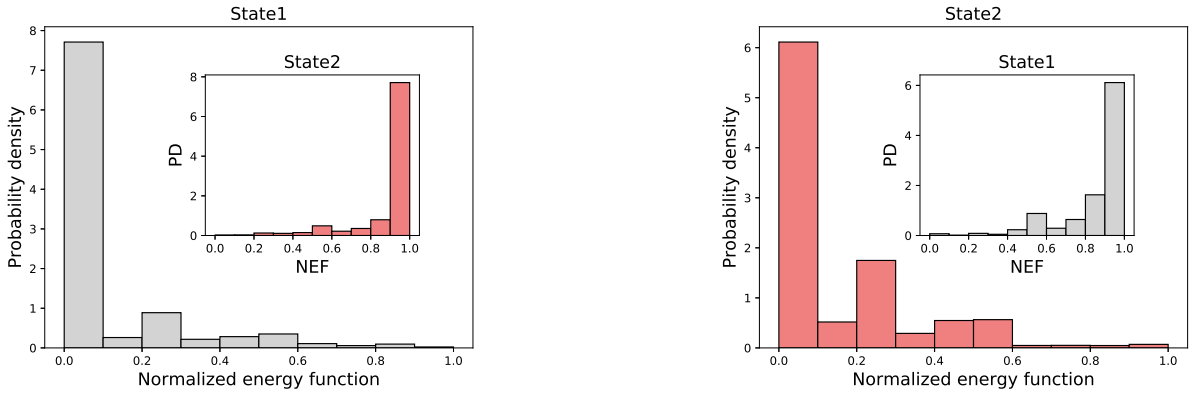
V. Discussion

A. Methodology

In contrast to the well-known Markov chain model and classical time series analysis methods that can only describe the physical process of a chain-like structure, MRF inherently has an intuitive network structure, which makes it very suitable for the characterization of contextual constraints and the derivation of the probability distribution of interacting features in two-dimensional space. For this reason, MRF has been widely employed to 2/3D spatial modeling, such as image and video analysis. Considering the similarity between the time-varying behavior process of the distributed network and the 3D video, we choose the MRF-framework to solve our problem. We also noticed that deep neural network (DNN) and its various variants (e.g., CNN, RNN and LSTM) have become a popular technology in different fields. The main reasons why we adopt the MRF-framework rather than the DNN-based models in this work are as follows. (i) The key link to the DTE detection is to model the time-varying process of the distributed network behavior. The net structure of the MRF allows it to effectively describe the spatiotemporal contextual interaction of each NE's behavior, and lets it exhibit mathematical analyzability for the described physical process. Although the DNN-based methods show good classification performance in some applications, they usually work as a black-box, which makes it difficult for them to provide understandable mathematical analysis for the physical process of the application problem to be resolved. (ii) Due to the mathematical analyzability, the MRF-framework is easy to couple with labelled and unlabelled real training data without human intervention and shows good stability, such as the parameter estimation and inference. However, the model learning of the DNN-based methods relies on a large amount of labelled data, which limits their usability in most practical applications, e.g., unsupervised learning involved in this work. Moreover, there are a large number of redundant parameters in the DNN-based models, which affect their convergence and operational efficiency. (iii) There are many mature theories and methods to support the expansion of the MRF for new modeling problems, such as the EM and MAP algorithms, Hammersley-Clifford theorem and iterated conditional modes.

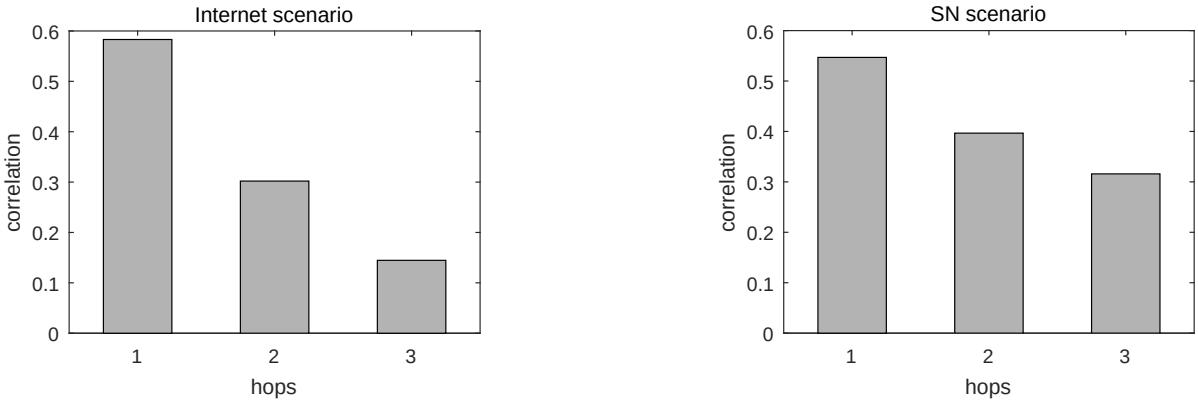
This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2020.3036664, IEEE Transactions on Dependable and Secure Computing



(a) Nodes select state1. (b) Nodes select state2.

Fig. 10: Distribution of energy function for state selection in the SN scenarios.



(a) Internet (b) SN

Fig. 11: Correlation vs. hops.

Yet in most of the DNN-based applications, model design and parameter estimation lack the necessary theoretical guidance because of their unanalyzability. The reliance on artificial experience leaves them with a lack of convenient and consistent approaches to deal with different application problems and thus becomes unstable. Combining the advantages of both the MRF-framework and the DNN-based models may be another optional scheme that will be explored in our future research.

In this work, GMM was used to describe the distribution of traffic features for each behavioral pattern. The reasons for adopting GMM mainly come from two aspects [50]: (i) it has been proved that a finite mixture of Gaussian components can model any continuous distribution with arbitrary precision if a sufficient number of components are provided and the parameters of the model are chosen correctly; (ii) GMM can be applied to a wide range of problems without any assumption with respect to the distribution properties of the raw data analyzed.

The normalization of the energy function shown in formula (8) is an important part of this work. We map each network element (e.g., router or switch) to a node in a graph, and only focus on their packet forwarding behavior rather than their actual role and category in the network. The heterogeneity of the nodes mainly comes from the huge difference in the degree of the nodes. Unlike image processing based on regular grid, there are significant differences in the degree of network nodes, which makes it impossible to adopt a unified indicator to decide the optimal state for every node. For example, the traffic behavior that is evaluated as normal by the high-degree nodes will be considered as abnormal by the low-degree nodes. Thus, we normalize the energy function of each node in the proposed approach, which maps the heterogeneous nodes to the same metric space, so that different types of nodes can share the same state space. Experiments show that this approach is simple and effective. It eliminates the influence of the heterogeneity of the nodes to a certain extent.

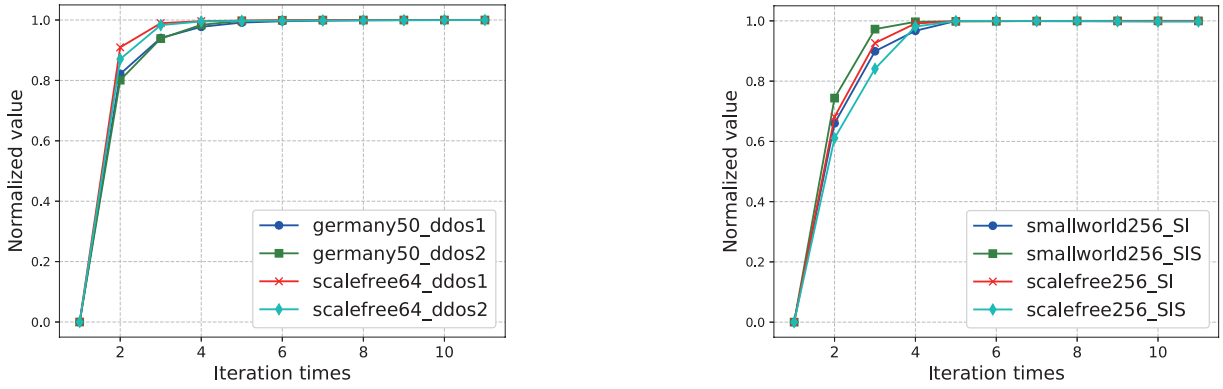
B. Configuration

Markovianity. The purpose of using the first-order Markovianity is to reduce the complexity of the spatiotemporal behavior model and highlight the core idea of the proposed scheme. Its essence is to simplify the model at the expense of higher-order information loss.

It is reasonable to adopt the first-order Markovianity in this work, because a large number of existing studies have shown that in a network the influence of nodes decreases with propagation. It means that the impact from the one-hop neighboring NEs is much larger than that of their multi-hop neighbors. This property was found in various types of networks, e.g.,

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2020.3036664, IEEE Transactions on Dependable and Secure Computing



(a) Internet (b) SN

Fig. 12: Convergence of the model

communication network [47] and social network [51]; it was also verified by our experiments shown in Fig. 11. Moreover, the above experiments also showed that this simplification does not have a significant impact on the final detection performance. On the contrary, the overall performance of the proposed scheme is better than other baseline methods. In fact, the proposed scheme is not limited to the first-order Markovianity. If there are enough computing resources, it can be extended to a higher-order hidden MRF that can describe the spatiotemporal context of the multi-hop neighbors [52]. The key technology is to solve the parameter estimation of the higher-order energy function. An alternative method is the mean field theory that has been widely used to solve multi-body problems. Because it adopts the pseudo-likelihood function to do approximate calculations, the spatiotemporal correlation described by the Markovianity is achieved during the iterative calculation process of the local mean values and the local mean field conditional probabilities, i.e., the current updated value of each node will affect all other nodes in the next iteration.

Number of states. As far as we know, currently there is no analytical approach for solving this issue for the Markov-family models. For the proposed scheme, there are at least three alternative ways to estimate the number of model states. (i) The number of hidden states can be defined according to the prior knowledge of network events [53]. (ii) If the network event is unknown, we can utilize the criteria or procedures based on exhaust algorithms to determine the number of the states [50], such as Akaike information criterion (AIC), Hannan-Quinn information criterion, Bayesian information criterion(BIC), the integrated classification likelihood criterion, the mutual information and the Greedy mixture learning. (iii) Similar to most Markov-based applications, the number of the states can be selected by experimenting the pending number one by one. On the other hand, theoretically each state should correspond to a meaningful entity, yet the actual results do not often happen as such, because the models with large number of states may easily lead to over-fitting. Moreover, some states may be merged during the model training, which causes the final states to be inconsistent with the physical entities, i.e., the actual number of states is much smaller than the theoretical value. Actually, the change in the number of states does not cause significant impact in this work.

Selection of traffic features. This work focuses on the network-centric DTE detection instead of feature engineering. Thus, we did not introduce new traffic features for the DTE detection in the above experiments, but adopted the traffic features that have been widely used in the VCAs for DDoS attack detection and SMS spam detection. In fact, the proposed scheme does not limit what traffic features should be used; it can support various types of traffic features such as scalars and vectors. Because there are no universal traffic features that can meet the requirements of all threat event detections, in practical applications, traffic features should be determined according to the target event to be detected.

Topology size. We did not adopt large-scale network topologies in the experiments. The reasons mainly come from two aspects. (i) Based on the theory of modeling, the proposed scheme achieved the purpose of event detection through multiple local iterations. Thus, the topology size has no significant impact on the model's performance except for the computing time, which shows the scalability of the proposed scheme. (ii) The growth of topology size seriously affects the efficiency of the simulation, as the operation and computation of all NEs and links are implemented on the same server. Therefore, in this work we have limited the maximum number of NEs to less than 256, which can balance the need for performance evaluation and the available computing resources.

C. The States of the model

States are usually defined manually in order to concisely describe a real event. They are just the abstract and logical expressions of the real event. Even for the same event, it is difficult to form a consistent definition of the states because the opinions of analysts are different.

In this work, we used an unsupervised learning method to make the proposed model automatically extract state information from the training data. Due to the limitations of the data size and the scale of the selected features, the states that the model learns from the given data are not exactly the same as those defined by humans; they can only capture partial components of the real states projected in the selected

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2020.3036664, IEEE Transactions on Dependable and Secure Computing

learning data, i.e., abstraction of real-world states, rather than the complete and objective view obtained by the human brain. This issue exists widely in most current machine learning tasks. A widely accepted solution is to capture more state information of the physical processes by increasing the scale of effective data and features.

As the purpose of this work is to develop a general NCA for the detection of distributed threat-events, we haven't explored the extraction of training data and features in depth, but only follow the methods validated by other works. In fact, the proposed NCA and algorithms are loosely coupled with the selection of training data and features. Thus, not only is it not limited to what is presented in the experiments, but it is able to support large-scale training data and features, which enables the state description ability of the proposed approach to be improved through massive learning data.

D. Performance

Algorithm complexity. In **Algorithm 1**, the proposed scheme only needs to traverse all NEs in the network. For a given NE, it only needs its observation, its previous state, and the states of its neighboring NEs to infer its current state. Hence the computation complexity of the proposed scheme is $O(N \cdot |\mathcal{S}|)$, where N is the number of samples and $|\mathcal{S}|$ is the number of states. For the computation complexity of the baseline methods, Birch is $O(N)$; Ward is $O(N^3)$; KMeans is $O(N \cdot |\mathcal{S}|)$, GMM is $O(N \cdot |\mathcal{S}|)$; and KFCM is $O(N \cdot |\mathcal{S}|)$. Therefore, the complexity of the proposed scheme is acceptable.

Convergence. The convergence of the EM-based learning algorithm has been proved theoretically for MRFs by previous studies [15]. Although the proposed model is not exactly the same as the classical MRF, it is based on the framework of the MRF. Thus, we do not make the theoretical proof for its convergence in this work, but only show the convergence process observed in the experiments. Fig. 12 shows the convergence of model training in the experimental scenarios. The abscissa indicates the number of iterations. The vertical axis denotes the normalized log-likelihood $\mathcal{L}$ that is used to evaluate the convergence of the model and to control the iteration for training. Both of the results show that the model is convergent after 5 iterations, which further verifies the effectiveness of the proposed training algorithm.

In general, the merits of the proposed scheme are threefold. (i) The proposed scheme achieves the DTE detection from the perspective of network-side, which provides a global perspective for the distributed scenarios. (ii) It utilizes the spatiotemporal context approach to model the dynamic evolution process of distributed network events and help increase the detection performance. (iii) The proposed scheme has a low computation complexity and can quickly converge.

VI. Conclusion and Future Work

In this work, a new network-centric approach was designed for distributed threat-event detection. The proposed approach treated a distributed network as a holistic system and characterized its spatiotemporal dynamic evolution process from the perspective of network-side. Then, network events were detected by the dynamic behavior analysis of the distributed network. We introduced the rationale of deriving the spatiotemporal behavior model in detail, and developed a two-layer hidden MRF to formulize the proposed model and achieve the numerical detection. Algorithms were derived for the model learning and event detection based on the EM algorithm and the MAP criterion, respectively. In the experiments, we evaluated the performance of the proposed approach through two independent distributed network scenarios. The results of three evaluation metrics showed that the proposed approach was superior to other five widely used baseline methods. Since the approach is not limited to a specific network scenario, it is expected to be applicable to different types of threat-event detection in various distributed scenarios.

In order to highlight the rationale of the proposed approach and make the computation tractable, we have adopted some simplified methods in this work, including manually defining the number of the model's states, only using the first-order Markovianity to describe the spatiotemporal context, making the independence constraint on the observed features of NEs, and merely considering the cloud-based deployment method. In addition, due to the limited space, we only showed the results of two typical DTE detection scenarios in the experiment. However, these simplifications are not mandatory for the proposed approach; they are optimizable and solvable. Thus, it is foreseeable that the proposed approach is not limited to what is described in this work. The limitations and some interesting issues arising from this work will be further explored in our future research, such as higher-order hidden MRF, fusing the MRF-framework and deep NN based methods, combining cloud computing and edge computing systems for the deployment of the proposed scheme, and applying it to a wider range of application scenarios like blockchain and IoT scenarios.

Acknowledgment

The authors would like to thank all anonymous reviewers for their valuable comments to improve this work.

References

[1] S. Dong, K. Abbas, and R. Jain, "A survey on distributed denial of service (DDoS) attacks in SDN and cloud computing environments," *IEEE Access*, vol. 7, pp. 80 813–80 828, 2019.

[2] S. Peng, S. Yu, and A. Yang, "Smartphone malware and its propagation modeling: A survey," *IEEE Communications Surveys Tutorials*, vol. 16, no. 2, pp. 925–941, Second 2014.

[3] M. Zhang, H. Luo, and H. Zhang, "A survey of caching mechanisms in information-centric networking," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 3, pp. 1473–1499, 2015.

[4] I. Afolabi, T. Taleb, K. Samdanis, A. Ksentini, and H. Flinck, "Network slicing and softwarization: A survey on principles, enabling technologies, and solutions," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 3, pp. 2429–2453, 2018.

[5] J. J. Pufang MA, Jiali YOU, "An efficient multipath routing schema in multi-homing scenario based on protocol-oblivious forwarding," *Frontiers of Computer Science*, vol. 14, no. 4, p. 144501, 2020.

[6] M. Peng, C. Wang, J. Li, H. Xiang, and V. Lau, "Recent advances in underlay heterogeneous networks: Interference control, resource allocation, and self-organization," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 2, pp. 700–729, 2015.

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2020.3036664, IEEE Transactions on Dependable and Secure Computing

[7] S. T. Zargar, J. Joshi, and D. Tipper, "A survey of defense mechanisms against distributed denial of service (DDoS) flooding attacks," *IEEE Communications Surveys Tutorials*, vol. 15, no. 4, pp. 2046–2069, Fourth 2013.

[8] Y. Ye, T. Li, D. Adjeroh, and S. S. Iyengar, "A survey on malware detection using data mining techniques," *ACM Computing Surveys (CSUR)*, vol. 50, no. 3, p. 41, 2017.

[9] B. A. S. Al-rimy, M. A. Maarof, and S. Z. M. Shaid, "Ransomware threat success factors, taxonomy, and countermeasures: a survey and research directions," *Computers & Security*, vol. 74, pp. 144–166, 2018.

[10] J. Jiang, S. Wen, S. Yu, Y. Xiang, and W. Zhou, "Identifying propagation sources in networks: State-of-the-art and comparative studies," *IEEE Communications Surveys Tutorials*, vol. 19, no. 1, pp. 465–481, Firstquarter 2017.

[11] J. Zheng, Q. Li, G. Gu, J. Cao, D. K. Yau, and J. Wu, "Realtime DDoS defense using COTS SDN switches via adaptive correlation analysis," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 7, pp. 1838–1853, 2018.

[12] B. Rashidi, C. Fung, and E. Bertino, "A collaborative DDoS defence framework using network function virtualization," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 10, pp. 2483–2497, 2017.

[13] S. Simpson, S. N. Shirazi, A. Marnerides, S. Jouet, D. Pezaros, and D. Hutchison, "An inter-domain collaboration scheme to remedy DDoS attacks in computer networks," *IEEE Transactions on Network and Service Management*, vol. 15, no. 3, pp. 879–893, 2018.

[14] S. Yu, W. Zhou, S. Guo, and M. Guo, "A feasible ip traceback framework through dynamic deterministic packet marking," *IEEE Transactions on Computers*, vol. 65, no. 5, pp. 1418–1427, 2015.

[15] S. Z. Li, *Markov random field modeling in image analysis*. Springer Science & Business Media, 2009.

[16] Y. Zhang, N. Meratnia, and P. J. Havinga, "Outlier detection techniques for wireless sensor networks: A survey." *IEEE Communications Surveys Tutorials*, vol. 12, no. 2, pp. 159–170, 2010.

[17] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Communications Surveys Tutorials*, vol. 18, no. 2, pp. 1153–1176, Secondquarter 2016.

[18] D. Abadi, S. Madden, and W. Lindner, "Reed: Robust, efficient filtering and event detection in sensor networks," *in Proceedings of the 31st international conference on Very large data bases, VLDB Endowment*, pp. 769–780, 2005.

[19] X. Yang, H. B. Lim, T. M. Özsu, and K. L. Tan, "In-network execution of monitoring queries in sensor networks," in *Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '07. New York, NY, USA: ACM, 2007, pp. 521–532.

[20] B. Krishnamachari and S. Iyengar, "Distributed bayesian algorithms for fault-tolerant event region detection in wireless sensor networks," *IEEE Transactions on Computers*, vol. 53, no. 3, pp. 241–250, 2004.

[21] S. Wen, W. Zhou, J. Zhang, Y. Xiang, W. Zhou, and W. Jia, "Modeling propagation dynamics of social network worms," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 8, pp. 1633–1643, Aug 2013.

[22] V. Karyotis, "A Markov random field framework for modeling malware propagation in complex communications networks," *IEEE Transactions on Dependable and Secure Computing*, pp. 1–1, 2018.

[23] H. Sadreazami, A. Mohammadi, A. Asif, and K. N. Plataniotis, "Distributed-graph-based statistical approach for intrusion detection in cyber-physical systems," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 4, no. 1, pp. 137–147, March 2018.

[24] V. P. Illiano, L. Muoz-Gonzlez, and E. C. Lupu, "Don't fool me: Detection, characterisation and diagnosis of spoofed and masked events in wireless sensor networks," *IEEE Transactions on Dependable and Secure Computing*, vol. 14, no. 3, pp. 279–293, May 2017.

[25] D. Jiang, Z. Xu, P. Zhang, and T. Zhu, "A transform domain-based anomaly detection approach to network-wide traffic," *Journal of Network and Computer Applications*, vol. 40, pp. 292–306, 2014.

[26] S. Wu, S. Liu, W. Lin, X. Zhao, and S. Chen, "Detecting remote access trojans through external control at area network borders," in *2017 ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS)*, May 2017, pp. 131–141.

[27] K. Peng, V. C. M. Leung, and Q. Huang, "Clustering approach based on mini batch kmeans for intrusion detection system over big data," *IEEE Access*, vol. 6, pp. 11 897–11 906, 2018.

[28] X. Yang, P. Zhao, X. Zhang, J. Lin, and W. Yu, "Toward a Gaussian-mixture model-based detection scheme against data integrity attacks in the smart grid," *IEEE Internet of Things Journal*, vol. 4, no. 1, pp. 147–161, Feb 2017.

[29] Y. Li, X. Luo, Y. Qian, and X. Zhao, "Network-wide traffic anomaly detection and localization based on robust multivariate probabilistic calibration model," *Mathematical Problems in Engineering*, vol. 2015, 2015.

[30] C. Yin, Y. Zhu, J. Fei, and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," *Ieee Access*, vol. 5, pp. 21 954–21 961, 2017.

[31] R. Vinayakumar, K. Soman, and P. Poornachandran, "Applying convolutional neural network for network intrusion detection," in *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*. IEEE, 2017, pp. 1222–1228.

[32] A. H. Mirza and S. Cosan, "Computer network intrusion detection using sequential LSTM neural networks autoencoders," in *2018 26th Signal Processing and Communications Applications Conference (SIU)*. IEEE, 2018, pp. 1–4.

[33] Z. Tu and S.-C. Zhu, "Image segmentation by data-driven Markov chain Monte Carlo," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 657–673, May 2002.

[34] J. Besag, "Statistical analysis of non-lattice data," *The statistician*, pp. 179–195, 1975.

[35] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the royal statistical society. Series B (methodological)*, pp. 1–38, 1977.

[36] Y. Zhang, J. M. Brady, and S. Smith, "Hidden Markov random field model for segmentation of brain MR image," in *Medical Imaging 2000: Image Processing*, vol. 3979. International Society for Optics and Photonics, 2000, pp. 1126–1138.

[37] R. L. Burden and J. D. Faires, *Numerical Analysis (9th Edition)*. Brooks/Cole, Cengage Learning, 2011.

[38] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proceedings of COMPSTAT'2010*. Springer, 2010, pp. 177–186.

[39] J. Lin and D. Zhou, "Online learning algorithms can converge comparably fast as batch learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 6, pp. 2367–2378, June 2018.

[40] D. Saravanan and S. Srinivasan, "Video data mining information retrieval using birch clustering technique," in *Artificial Intelligence and Evolutionary Algorithms in Engineering Systems*, L. P. Suresh, S. S. Dash, and B. K. Panigrahi, Eds. New Delhi: Springer India, 2015, pp. 583–594.

[41] F. Murtagh and P. Legendre, "Ward's hierarchical agglomerative clustering method: Which algorithms implement ward's criterion?" *Journal of Classification*, vol. 31, no. 3, pp. 274–295, Oct 2014.

[42] Y. Ding and X. Fu, "Kernel-based fuzzy c-means clustering algorithm based on genetic algorithm," *Neurocomputing*, vol. 188, pp. 233 – 238, 2016, advanced Intelligent Computing Methodologies and Applications.

[43] S. Orlowski, R. Wessäly, M. Pióro, and A. Tomaszewski, "Sndlib 1.0–survivable network design library," *Netw.*, vol. 55, no. 3, pp. 276–286, May 2010.

[44] S. H. Strogatz, "Exploring complex networks," *Nature*, vol. 410, no. 6825, p. 268, 2001.

[45] X. Ma and Y. Chen, "DDoS detection method based on chaos analysis of network traffic entropy," *IEEE Communications Letters*, vol. 18, no. 1, pp. 114–117, January 2014.

[46] C. C. Zou, D. Towsley, and W. Gong, "Modeling and simulation study of the propagation and defense of internet e-mail worms," *IEEE Transactions on Dependable and Secure Computing*, vol. 4, no. 2, pp. 105–118, April 2007.

[47] Y. Wang, S. Wen, Y. Xiang, and W. Zhou, "Modeling the propagation of worms in networks: A survey," *IEEE Communications Surveys Tutorials*, vol. 16, no. 2, pp. 942–960, Second 2014.

[48] Q. Xu, E. W. Xiang, Q. Yang, J. Du, and J. Zhong, "SMS spam detection using noncontent features," *IEEE Intelligent Systems*, vol. 27, no. 6, pp. 44–51, Nov 2012.

[49] C. M. Jarque and A. K. Bera, "A test for normality of observations and regression residuals," *International Statistical Review/Revue Internationale de Statistique*, pp. 163–172, 1987.

[50] G. J. McLachlan, S. X. Lee, and S. I. Rathnayake, "Finite mixture models," *Annual review of statistics and its application*, vol. 6, pp. 355–378, 2019.

[51] J. Zhao, L. Yu, and J.-R. Li, "Node influence calculation mechanism based on bayesian and semiring algebraic model in social networks," *Acta Phys. Sin.*, vol. 62, no. 13, pp. 130 201–130 201, 2013.

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2020.3036664, IEEE Transactions on Dependable and Secure Computing

[52] Z. L. Wenjie LIU, "An efficient parallel algorithm of n-hop neighborhoods on graphs in distributed environment," *Frontiers of Computer Science*, vol. 13, no. 6, p. 1309, 2019.

[53] J. Pohle, R. Langrock, F. M. van Beest, and N. M. Schmidt, "Selecting the number of states in hidden Markov models: pragmatic solutions illustrated using animal movement," *Journal of Agricultural, Biological and Environmental Statistics*, vol. 22, no. 3, pp. 270–293, 2017.

Haishou Ma received his bachelor's and master's degrees from Sun Yat-Sen University, Guangzhou, China in 2016 and 2019, respectively. He is currently an engineer at Huawei Technologies Co Ltd. His current research interests includes home networking optimization, WiFi MAC scheduling algorithm design and implementation.

Yi Xie is currently an Associate Professor at the School of Data and Computer Science, Sun Yat-Sen University. He received the B.Sc., M.Sc. and Ph.D. degrees from Sun Yat-Sen University, Guangzhou, China. He was a visiting scholar at George Mason University and Deakin University during 2007 to 2008, and 2014 to 2015, respectively. He won the outstanding doctoral dissertation award of the Chinese Computer Federation (CCF) in 2009. His recent research interests include networking, cyber security and behavior modeling. Some of his works have been published in IEEE top journals, such as ToN, TPDS, TBD, TCSS and Sensors. He has received eight research grants and has served as a young Associate Editor for a Springer journal named Frontiers of Computer Science.

Shensheng Tang is currently with the Department of Electrical and Computer Engineering in St Cloud State University, USA. He received his Ph.D. from The University of Toledo, USA. He has eight years of product design and development experience in electronics and wireless industry, as hardware engineer, system engineer, and manager respectively. His current research interests include embedded systems, networking (wired, wireless), Internet of things (IoT), and modeling and performance evaluation. He has served or is serving as an editor or Guest Editor for International Journals and a TPC member of international conferences. He is a senior member of IEEE.

Jiankun Hu is currently a Professor with the School of Engineering and IT. University of New South-Wales, Canberra, Australia. He is also an invited expert of Australia Attoney-General's Office, assisting the draft of Australia National identity Management Policy. He has received nine Australian Research Council (ARC) Grants and has served at the Panel on Mathematics, Information, and Computing Sciences, Australian Research Council ERA (The Excellence in Research for Australia) Evaluation Committee 2012. His research interests are in the field of cyber security covering intrusion detection, sensor key management, and biometrics authentication. He has many publications in top venues, including the IEEE Transaction on Pattern Analysis and Machine Intelligence, the IEEE Transaction Computers, the IEEE Transaction on Parallel and Distributed Systems, the IEEE Transaction on Information Forensics and Security, Pattern Recognition, and the IEEE Transactions on Industrial Informatics. He is an Associate Editor of the IEEE Transaction on Information Forensics and Security.

Xincheng Liu (IEEE SM'12) received the B.E. and M.E. degrees in electrical engineering from the Huazhong University of Science and Technology, Wuhan, China, and the Ph.D. degree from Sun Yat-sen University (SYSU), Guangzhou, China. He received the Royal Society K. C. Wong Fellowship of the U.K. for his Post-Doctoral Research from the University of Southampton, Southampton, U.K., from 2002 to 2003. He was a Visiting Scientist with Oregon State University, Corvallis, OR, USA, from 2004 to 2005. He is currently a Full Professor with the School of Electronics and Information Technology, SYSU. He is also a primary investigator of several projects on wireless communications and networking. He has authored and co-authored over 140 peer-reviewed papers in journals and conferences. His main research interests include wireless sensor networks, Internet of Things, and channel coding theory and applications. Dr. Liu is a Senior Member of the China Computer Federation and the China Institute of Communications.

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/