Information Sciences 479 (2019) 456–471

Contents lists available at ScienceDirect

Information Sciences

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

Modeling and clustering attacker activities in IoT through machine learning techniques

Peiyuan Sun[a], Jianxin Li[a,*], Md Zakirul Alam Bhuiyan[b], Lihong Wang[c], Bo Li[a]

[a] School of Computer Science and Engineering, Beihang University, Beijing, China
[b] Department of Computer and Information Sciences, Fordham University, NY, USA
[c] National Computer Network Emergency Response Technical Team/Coordination Center of China, Beijing, China

ARTICLE INFO

Article history:
Received 12 August 2017
Revised 16 April 2018
Accepted 21 April 2018
Available online 23 April 2018

Keywords:
IoT
Botnet
Multivariate Hawkes Process

ABSTRACT

With the rise of the Internet of Things, malicious attacks pose serious threats to the massive vulnerable IoT devices. Recently, attackers have initiated increasingly coordinated attack activities commonly pertaining to botnets. However, how to effectively detect the botnet based on attacker activities is proven challenging. In this paper, we propose a Machine Learning-based method for modeling attacker activities based on the following intuitive observations: attackers in the same botnet tend to launch temporally close attacks. We then directly model attack temporal patterns using a special class of point process called Multivariate Hawkes Process. Intuitively, Multivariate Hawkes Process identifies the latent influences between attackers by utilizing the mutually exciting properties. We then cluster the attacker activities based on the inferred weighted influence matrix with resort to the graph-based clustering approach. To evaluate the applicability of our method, we deployed 10 honeypots in a real-world environment, and conduct experiments on the collected dataset. The results show that we can identify the activity pattern and the structure of botnets effectively.

1. Introduction

With the recent rapid development of the Internet of Things (IoT) [1,2,7,14], there has been increasing interest in understanding emerging cyberthreats in IoT. IoT devices are extremely vulnerable and attractive to attackers for their highly heterogeneous components, naive security configurations and weak encryption verification [5,22]. The large number of insecure IoT devices makes them low-hanging targets for attackers to create large-scale botnets [4]. In September 2016, a botnet built from the Mirai malware flooded the DNS provider Dyn with a DDos attack, which is one of the largest on record, exceeding 620 Gbps. Mirai spread by scanning IoT devices and deducing the administrative credentials by means of brute force. The Mirai author claimed that over 380,000 IoT devices were enslaved by the Mirai malware in the attack by October 2016 [12].

Lots of solutions have been proposed to detect botnets [15,16,32,33,39,41,42]. The most common approach is based on the botnet's signature [43]. However, most modern botnets have built-in signature updating mechanisms which renders this

* Corresponding author.
E-mail addresses: sunpy@act.buaa.edu.cn (P. Sun), lijx@act.buaa.edu.cn (J. Li), mbhuiyan3@fordham.edu (M.Z. Alam Bhuiyan), wlh@isc.org.cn (L. Wang), libo@act.buaa.edu.cn (B. Li).

https://doi.org/10.1016/j.ins.2018.04.065
0020-0255/

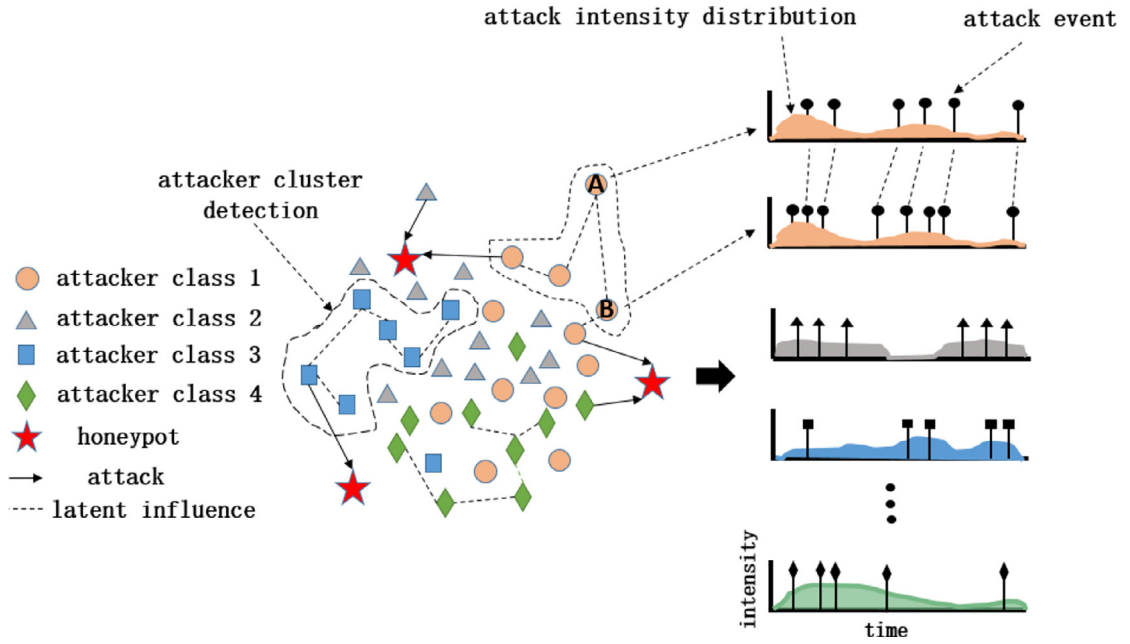

Fig. 1. An illustration of the proposed framework showing the detection of attacker clusters based on the attacker's temporal information.

detection approach more challenging [16]. Another approach exploits the network traffic flow. In [41], the authors developed a technique for identifying P2P bots from file-sharing hosts using features related to traffic volume, "churn" among peers and differences between human-driven and machine-driven traffic. However, traffic related features are not robust, as malicious traffic can exploit multiple methods to blend in with the legitimate traffic. Recent new approaches unveil the botnet through group behavior analysis. Yan et al. [39] proposed a botnet detection system to distinguish P2P bot-infected hosts from legitimate P2P hosts by jointly considering flow-level traffic statistics and network connection patterns. However, due to the dynamics of the Internet traffic and advanced malware obfuscation or polymorphism, these botnet detection approaches are usually not robust and reliable enough to distinguish specific botnets from benign hosts.

While it is highly difficult to distinguish botnets from legitimate networks, honeypots can significantly simplify the detection process. Network security administrators have been used to depend to a great extent on honeypots to understand existing and emerging threats on the Internet. Specially, as regards botnet detection, honeypots play a significant part. Since honeypots have no production activity, all connections to the honeypot are suspects by nature, and therefore can be detected as an unauthorized scan or attack with almost no false positives and negatives [34]. However, the vast amount of captured data by honeypots can easily overwhelm the administrators for its tremendous volume and diversity. It is challenging to model and cluster the attacker's activity pattern which could provide the administrators with high-level vision of the emerging security trends, and subsequently aid them to improve the defense mechanism.

Several works have been conducted to uncover attackers' pattern based on the analysis of honeypot-collected data [3,36]. Various classification and clustering algorithms in the Machine Learning field have been successfully employed in these works with laborious feature engineering. In [36], the researchers exploited the time signature: extracting an exhaustive set of features describing the attack characteristics, and then using these features to identify a time series representing the aggregated source count for a given type of attack. However, one never knows if the set of features that were fed into the classifiers or clustering algorithms is enough. Then the performance is highly sensible to the quality of the features. In this work, we focus on mining attacker patterns based on only attackers' event time history which might occur as a result of scanning activity. Our approach relieves the burden of selecting and extracting features.

We propose a framework to model and cluster attacker activities in IoT using directly attack temporal information. To the best of our knowledge, there has been fairly little related research. The main concept of our work is depicted in Fig. 1.

Main concept. Several honeypots were deployed on a world-wide scale to collect attacks from the Internet. Each attacker is assumed to belong to a botnet which launches coordinated attacks. Then it is reasonable to assume that there exists the latent influence between attack temporal distribution of attackers in the same botnet. We detect the latent influence between attackers *A* and *B* (as nodes A and B illustrated in Fig. 1) by identifying if there exists large enough number of attack events from attacker *B* that follow closely the corresponding events from attacker *A*. Then cluster structure of attackers are identified by grouping all attackers that have strong latent influences between each other. One may image the influence identification procedure as splitting attacker activities into several disjoint time intervals, and then the comparison of two attackers' activity sequence is performed on each interval. However, since the duration and time interval distribution of attackers' activity vary dramatically, it is unreasonable to set a universal time unit to split the sequence. We employ a

probabilistic approach to inferring the latent influence between attackers. Intuitively, the approach assumes a probabilistic distribution with latent parameters assigned with each attacker. Then the activity of this attacker could be characterized by these latent parameters. The latent influence can thus be inferred through a great quantity of inference algorithms in the Machine Learning field.

Multivariate Hawkes Process, as a special type of point process, has been proven to be greatly successful in modeling the temporal pattern of several scenarios. We classify these works into three categories: latent network inference, cascade prediction and streaming data clustering. Yang and Zha [40], Choi et al. [9], Etesami et al. [13], Linderman and Adams [30] and He et al. [17] fall into the field of latent network inference. This problem was raised due to the fact that some real network structures are implicit, with only event sequences on each node observable. Multivariate Hawkes Process was widely adopted in this field under the intuition that an event at a node will induce activity at an adjacent node. This idea was formalized by corresponding each node with a Hawkes Process, and the network structure was discovered by inferring the mutually-exciting intensity between nodes. Cao et al. [6], Cheng et al. [8], Zhao et al. [44] and Li et al. [24] focus on modeling and predicting the cascade in social network. Hawkes Process is ideal for modeling information cascade because each new resharing of a post not only increases its cumulative resharing count, but also exposes new followers who may further reshare the post. The inferred accumulated self-exciting intensity was used to predict the cascade size. Mavroforakis et al. [31], Li et al. [29] and Duet al. [11] considered to discover users activity pattern from streaming data. The temporal pattern was characterized by Hawkes process, and the corresponding semantic was uncovered with resort to Topic Model. According to the above analysis, our work can be attributed to the field of latent network inference.

In our scenario, we model the activities of each attacker as events of a corresponding Hawkes Process, and the mutually exciting nature of Multivariate Hawkes Process emerges as we assume that latent influences lie within the attackers. Intuitively, if several attackers are governed by the same organization or person, they will launch temporally close attacks against the same target, which indicates that an implicit network governs the coordinated attack activities. The nodes in the implicit network represent the attackers, and the edges stand for the latent influences. If the underlying networks are given, we could assign a likelihood to the observed attack activities launched by attackers in the network. In this paper, we adopt a probabilistic model proposed by Linderman and Adams [30] which combined by the Hawkes Process with hierarchical prior network distribution. In this model, the hierarchical network prior encodes the belief in the underlying latent network structure. Then the Bayesian Probabilistic Graphical Model is employed to reason about attacker activities.

In the second step, we cluster all attackers that resemble each other based on the inferred latent network. There have been a great many clustering algorithms for doing this. In our work, we employ a graph-based approach [36] to performing it for convenience.

Our main contributions are summarized as follows:

- We design a framework that models and clusters attacker activities using directly the temporal information without the need for laborious feature engineering.
- We for the first time reformulate the attacker clustering problem as a latent structure discovering subproblem and a graph-based clustering subproblem. Then the Bayesian Probabilistic Graphical Model and quality-based clustering algorithm in Machine Learning are employed to address these two subproblems.
- We deployed 10 honeypots in a real-world environment, and evaluate our method on the collected data. The results show that we can identify the activity pattern and the structure of botnets effectively.

The rest of the paper is organized as follows. Section 2 surveys the related works. Section 3 explains preliminaries. Section 4 explicates the attacker activity model and the clustering algorithm. Section 5 elucidates the security analysis with Machine Learning for our framework. We evaluate our model and analyze the result in Section 6. In the last section, we present the conclusion and future research interest.

2. Related work

In this section, we present a comprehensive review of the related work. Previous studies in the literature can be placed into three categories. We survey each category as follows.

Botnet detection: There has been vast numbers of methods proposed to detect a botnet. The most conventional work was conducted via the botnet's signature [43]. In regard to this, the author proposed a detection algorithm based on traffic behavior analysis by classifying network traffic behavior using Machine Learning techniques. A newer line of work considered extracting the network flow features, based on which Machine Learning classification algorithm was employed to distinguish the botnet [28]. In [35], the Random Forest-based Decision Tree model built in Mahout is applied to the problem of Peer-to-Peer Botnet detection in quasi-real-time. Recent works jointly took into account the network flow feature and the botnet group behavior analysis [32,39]. In [39], the author presented PeerClean, a system that detects P2P botnets in real time by jointly considering flow-level traffic statistics and network connection patterns. In [32], detection is performed based on unifying behavioral analysis with structured graph analysis. The major distinction between our work and this line of work is that we focus on unraveling the attacker pattern and group structure based on the honeypot-collected data instead of discovering and isolating the botnet from the legitimate network.

Attacker pattern mining: Several closely related works can be found in the literature [36]. In [36], the author developed a framework which relies on a quality-based clustering technique specifically designed to identify all groups of highly similar

attack patterns. The primary clustering feature they considered is "time signature". Nevertheless, this work is different from ours, as Thonnard and Dacier [36] focused on the aggregated time series which were based on the preliminary classification of the traffic, while our work concerns the time interval information of the discrete events in continuous time. In [3], the author proposed a probabilistic model which captures the dynamics of attack propagation across the monitored honeypot in the system. While their method utilized the attacker's temporal history, we aim at modeling different aspects of attacker activity patterns.

Similar to attacker pattern mining, another line of work becomes active, which concerns mining people's online learning pattern. Some work in this field draw their attention on employing the temporal information of people's search task [29], online learning activity [31], and meme tracking [17]. In [29], they considered to identify and label search tasks by modeling the query temporal pattern combined with the query's textual content distribution. In [31], the author introduced a novel probabilistic mode, the Hierarchical Dirichlet Hawkes Process, which leverages the Hierarchical Hawkes Process, a popular Bayesian nonparametric model for clustering the topics accross different documents, combined with the Hawkes Process. The work by He et al. [17] concerns simultaneously reasoning about the information diffusion pathways and the topics characterizing the observed textual information. While there exist a myriad of pertinent works in the field of social network, to our knowledge, no proposition has been made to employ the related techniques in the IoT scenario.

Attacker pattern clustering: There exist a plethora of clustering algorithms [38]. In [39], a partition-based clustering method, Affinity Propagation, is used to cluster the similar hosts. In [32], a spectral clustering technique was employed to partition the constructed botnet to separate subgraphs corresponding to different traffic characteristics. In [28], the author utilized the X-means algorithm to group similar flows, which are likely to come from the same botnet, into one cluster. X-means clustering is a variation of K-means that refines cluster assignments by repeatedly attempting subdivision, and keeping the best resulting splits, until some criterion is reached. In this work, our scenario can be formulated as finding the largest group of similar elements in a graph. We then employ a graph-based clustering algorithm [36] to uncover the group structure of attackers.

In conclusion, our framework is comprised of three parts: (1) collecting the attack activities from the Internet by deploying several honeypots on the world-wide scale makes it different from the conventional botnet detection that relies on traffic flow and behavior analysis; (2) inferring the latent influence between attackers by directly modeling the temporal information of attackers is the first trial to employ Multivariate Hawkes Process in IoT scenario and (3) clustering attacker groups by repeatedly merging the similar nodes in a greedy fashion has been proven efficient and effective.

3. Preliminaries

In this section, we briefly discuss the Bayesian Probabilistic Graphical Model and Multivariate Hawkes Process.

3.1. Bayesian probabilistic graphical model

There is a great collection of literature on the Bayesian Probabilistic Graphical Model (BPGM) [25]. Basically, BPGM is a generic model that represents the probability-based relations among random variables by a graph, and is a general method for knowledge representation and inference involving uncertainty. Latent random variables model the unobserved states that induce the real world data. The joint probabilities of parameters, latent variables and observed data can be expressed by specifying a distribution over each latent variable and our prior belief in these variables. Then we can use the tools of Bayesian inference to compute the posterior distributions over parameters and latent variables, thus giving us an insight to the latent state of the data [30].

3.2. Multivariate Hawkes Process

Poisson point process is a fundamental statistical tool to model the discrete points randomly located on timeline. Recently, researchers have successfully employed the point process to model an assortment of data produced by online social networks, such as the event time of retweets and link creation [11]. Poisson point process is often defined as a counting process [23]. A counting process is a stochastic process $(N(t): t \geq 0)$ taking values in $\mathbb{N}_0$ that satisfies $N(0) = 0$, is almost surely finite, and is a right-continuous step function. A counting process can be viewed as a cumulative count of the number of 'arrivals' into a system up to the current time. The conditional intensity function is generally used to characterize the point process. Denoted by $\mathcal{H}_t$, the history of event time $\{t_1, t_2, \ldots, t_n\}$ is up to but not includes time t. Then the intensity function is defined as:

$$\lambda^*(t|\mathcal{H}_t) = \lim_{\Delta t \to 0} \frac{\mathbb{E}[N(t+\Delta t) - N(t)|\mathcal{H}_t]}{\Delta t} \tag{1}$$

which measures the probability for the occurrence of a new event given the history $\mathcal{H}_t$. For simplicity, we denote the $\lambda^*(t|\mathcal{H}_t)$ as $\lambda(t)$ hereafter. The intensity function is often defined in the following form:

$$\lambda(t) = \lambda_0 + \int_0^t \mu(t-\mu) dN(\mu) \tag{2}$$

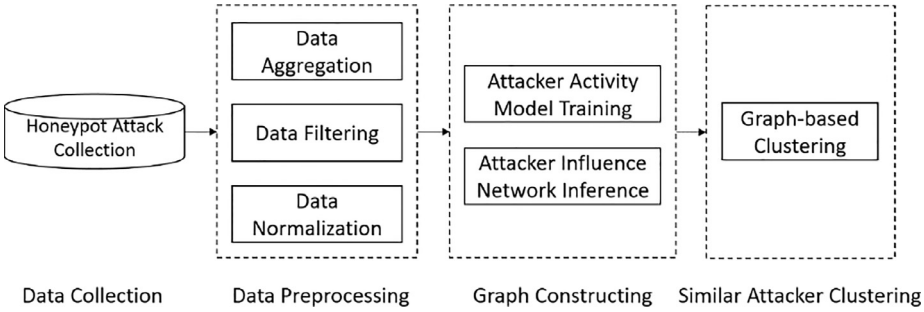

Fig. 2. Work flow of the framework.

for some $\lambda_0 > 0$ and μ: $(0, \infty) \to [0, \infty)$ which are called the background intensity and excitation function respectively. Such a process $N(\cdot)$ is designated as a *Hawkes Process*. The point process representation of temporal data models explicitly the time intervals of randomly located events instead of elaborately picking a time window to aggregate events.

However, the poisson process cannot model the mutually exciting interactions between events (e.g. post event happening to some users can improve the probability of their neighborhood to express their own opinions). Hawkes Process is a natural statistical object to model this phenomenon, and has been widely adopted to model the earthquake aftershocks and neural spike trains [30]. A Multivariate Hawkes Process is defined by the intensity function:

$$\lambda_n(t|\mathcal{H}_t) = \lambda_0^{(0)} + \sum_{m=1}^{M} h_{c_m \to n}(t - s_m) \tag{3}$$

where $\lambda_0^{(0)}$ is node n's background rate. $\mathcal{H}_t$ is the event history before time t. $h_{c_m \to n}(\Delta t)$ is the impulse response of event m added to node n. Intuitively, the background rate models the expected spontaneous event number fired by node n during the observed time window. The impulse response models the time-decayed influence between each pair of nodes. More specifically, following [30], we decompose the impulse response as follows:

$$h_{c_m \to n}(\Delta t) = A_{c_m,n} \cdot W_{c_m,n} \cdot \hbar(\Delta t) \tag{4}$$

where A is the binary adjacency matrix of the hidden network, and W is the influence matrix which models the expected triggered event number on node n by event m. $\hbar$ models the node-independent time-decay influence function.

4. Attacker activity clustering framework

In this section, we first present the Attacker Activity Model to show how to identify the latent influence between attackers by employing the Bayesian Probabilistic Graphical Model in Machine Learning, and then describe the graph-based clustering algorithm. Fig. 2 displays the workflow of the proposed framework. Firstly, attack data collection is conducted by the honeypots deployed on a world-wide scale. Besides the temporal information we are most concerned with, serveral pieces of other attack contextual information were also recorded by our honeypots. Secondly, data preprocessing is performed to provide a clean and well-formatted dataset for the next step. The preprocessing procedure includes: (1) Data aggregation: we aggregated the collected data from different honeypots by identifying the attack source and adjusting the jet lag. (2) Data filtering: attackers that launched less than 30 attacks in the time window are filtered out during this step. (3) Data normalization: since only the time interval information is under our concern, the whole dataset is translated to the origin of the time axis. Thirdly, the data after preprocessing is fed into the proposed attacker activity model. This process is responsible for training a Bayesian Probabilistic Graphical Model by Gibbs sampling algorithm. At the end of this step, we can learn the latent influence strength between attackers. Finally, similar attackers are clustered by applying a graph-based clustering algorithm. In this step, we group all attackers whose latent influence strength are of high similarity.

4.1. Attacker activity model

In the botnet scenario, groups of attackers owned by the same organization or person execute sequentially scans or other subsequent activities to perform an attack. In this context, attackers in the same group may be expected to undertake similar sequences of activities which we view as the *attacker pattern*. We focus on the temporal characteristic of the attack activity sequences.

As described in Section 3.1, the Bayesian Probabilistic Graphical Model is a proven powerful tool to reason about the dynamics and structure of the real world problems. The standard workflow for this style of model is to first propose a generative probabilistic model for the formulated problem. The Generative Probabilistic Model assumed the generative process of the observed data. The most important ingredients within are latent variables which capture the latent states inducing the observed data and the distributions that characterize the relations between latent variables and the observed data. Once

Table 1
Notations used in this paper.

Notations	Definition
T	Observation time window
M	Total attack number
s_m	Timestamp of attack m
c_m	Node of attack m
$\lambda_n^{(0)}$	Spontaneous attack rate of node n
$W_{nn'}$	Influence between attacker n and n'
$A_{nn'}$	Connection between attacker n and n'
$\theta_{nn'}$	Parameter of time-decayed influence between attacker n and n'
ω_m	Origin of attack m

a full story of the generative process is proposed, we can assign a likelihood to the observed data, combined with the latent variables and other parameters in the model. Then the posterior distribution of the latent variables can be obtained by means of Bayesian inference algorithms, such as Markov Chain Monte Carlo or variational inference algorithm.

Generative process: While in [30], the author gives a thorough formulation of the generative process of Hawkes Process on latent network. We next formulate a context-aware generative process of the attacker activity model. The notations used in our model are summarized in Table 1.

We represent each attack activity as a tuple (s_m, c_m), and denote the set of attack activities collected by honeypots as $E = \{\{s_m, c_m\}|m = 1, \ldots, M\}$. Each tuple represents an attack launched by an attacker c_m at time s_m. Every time an attacker c performs an activity, it may choose either to start a new wave of attacks or scans, or to follow-up an already ongoing attack or scan task. As described in Section 3.2, the intensity of launching an attack at time t by an attacker n depends on its spontaneous activity rate and the cumulative influence by all previous activities either by itself or by other attackers. The intensity function is given by

$$\lambda_n(t|\mathcal{H}_t) = \underbrace{\lambda_n^{(0)}}_{new\ attack} + \overbrace{\sum_{m=1}^{M} A_{c_m,n} \cdot W_{c_m,n} \cdot \hbar(t - s_m)}^{follow\ up}. \tag{5}$$

The decomposition of the influence matrix to sparse adjacency matrix and weight matrix is known as a spike-and-slab model [37]. It is worth noting that when the elements in matrix A are all 1s which means the latent network is a full connected graph. Then the model degenerates to the standard Multivariate Hawkes Process. Intuitively, the spike-and-slab variable selection approach incorporate our prior knowledge on the latent variables into the model. Specifically, the sparsity prior is put on the adjacent matrix A in our model. The sparsity assumption is reasonable since most attackers only connect with their neighborhoods in the same botnet. In Section 6, the experiments validate our assumption.

In our scenario, we employ the hierarchical network model *Latent Distance Model* [19] as our prior belief to the adjacency matrix distribution. The *Latent Distance Model* is a generative model for random graphs. The locations are given spherical Gaussian priors, $z_n \sim \mathcal{N}(0, \tau\mathcal{I})$, and the scale is drawn from an inverse gamma prior, $\tau \sim IGa(1, 1)$. The offset is given a standard normal prior, $\gamma_0 \sim \mathcal{N}(0, 1)$.

The weight matrix models the influence strength between attackers. Intuitively, if attacker c is greatly influenced by attacker c', we may expect that attacker c follows closely attacker c''s activity. More specifically, the total number and time interval distribution of these two attackers will expose the similar trends. For the conjugacy of the Bayesian model, we place the gamma prior on the weight distribution following [30], $W_{nn'} \sim Gamma(\kappa_{nn'}, \nu_{nn'})$, where we fix the $\kappa_{nn'} \equiv \kappa$ and then $\mu_{nn'} = \frac{\kappa}{\nu_{nn'}}$ due to the properties of the Gamma distribution. For the same reason, we put the Gamma distribution on attackers' spontaneous activity distribution, $\lambda_n \sim Gamma(\alpha_\lambda, \beta_\lambda)$. The conjugacy property is widely used in the Bayesian model which ensures that the posterior distribution of a parameter will have the same form as its prior distribution. This property can greatly simplify the inference process of the model, since the posterior distribution has a tractable form.

According to the superposition principle of the poisson process, each additive element in Eq. (4.1) can be viewed as an independent poisson process. This principle inspires us to augment each attack tuple (s_m, c_m) to (s_m, c_m, ω_m) where ω_m indicates the origin of each attack. If ω_m is 0, the attack is launched by the attacker spontaneously; otherwise, this variable attributes each attack to its parent attack event. Then the likelihood of the attack events given the spontaneous attack rate, adjacent matrix and influence matrix is:

$$p\Big(\{(s_m, c_m, \omega_m)_{m=1}^M|\{\lambda_n^{(0)}\}\}, \{\{h_{n\to n'}(\Delta t)\}\}\Big) = \prod_{n=1}^{N} \mathcal{PP}(\{s_m : c_m = n \& \omega_m = 0\}|\lambda_n^{(0)})$$
$$\prod_{m=1}^{M}\prod_{n=1}^{N} \mathcal{PP}(\{s_{m'} : c_{m'} = n' \& \omega_{m'} = m\}|h_{c_m\to n'}(t - s_m)), \tag{6}$$

where $\mathcal{PP}$ represents the poisson process.

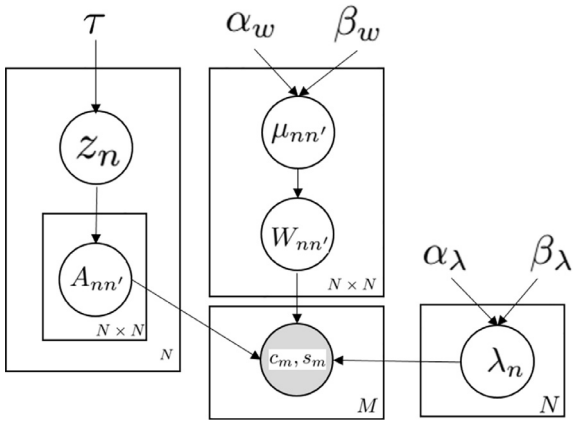

Fig. 3. Attacker activity model.

Having all the ingredients of the model, we give the plate notation of the attack activity model in Fig. 3. The generative process of the observed attack events is presented as follows:

- For each attacker n, draw its spontaneous attack rate $\lambda_n \sim Gamma(\alpha_\lambda, \beta_\lambda)$.
- For each pair of attackers $\{n, n'\}$,
  - draw the sparse connection variable $A_{nn'}$ according to the generative process of the *Latent Distance Model*;
  - draw the latent influence variable $W_{nn'} \sim Gamma(\kappa_{nn'}, \nu_{nn'})$;
- Draw each attack event $\{s_m, c_m, \omega_m\}$ according to the Multivariate Hawkes Process:
  - For each attacker n, draw the spontaneous attack event number $M_0 \sim \int_T \lambda_n^{(0)} dt$ [21];
  - For each spontaneous attacker event e_m, draw the individual time s_m from the density of the intensity function;
  - For each pair of attackers $\{n, n'\}$, first draw the number of impulse event as step 2, and then draw the individual attack time.

As Fig. 3 makes clear, there are three parts composing the attacker activity model. The variable $A_{nn'}$ denotes the connection between attackers. The parameter τ is a network-level parameter, assumed to be sampled once in the process of generating a network which indicates the latent place of the node. The variable Z_n is a node-level variable, sampled once per node. The variable $W_{nn'}$ represents the influence strength between attackers, the parameters α_w and β_w are network-level ones, and the variable $\mu_{nn'}$ is the node-level. The variable λ_n stands for the spontaneous attack rate for each node, and the parameters α_λ, β_λ are in the network-level.

Accordingly, the generative process can be divided into three steps. The first step generates the latent connections between attackers, and is accomplished by the Latent Distance Model. The second step engenders the spontaneous attack rate for each attacker which is assumed to follow the Gamma distribution. The third step produces the attack events for each attacker by means of the Multivariate Hawkes Process.

Inference: Given the attack activity events $E = \{\{s_m, c_m\} | m = 1, \ldots, M\}$ collected by honeypots, our goal is to infer the latent variables A, W, $\{\lambda_n\}$ in the model. Since the model is intractable, we must resort to the approximation methods to perform the Bayesian inference. The two most popular approximation methods for this purpose are variational inference and Markov Chain Monte Carlo (MCMC). We restate the posterior distribution of each latent variable below for completeness and clarification [30].

Sampling weight matrix:

$$p(W_{nn'} | \{s_m, c_m\}, A_{nn'} = 1, \kappa, \nu_{nn'}) = Gamma(W_{nn'} | \tilde{\kappa}_{nn'}, \tilde{\nu}_{nn'}), \tag{7}$$

where

$$\tilde{\kappa}_{nn'} = \kappa + \sum_{m=1}^{M} \sum_{m'=1}^{M} \mathbb{1}[c_m = n] \mathbb{1}[c_{m'} = n'] \mathbb{1}[\omega_{m'} = m], \tag{8}$$

$$\tilde{\nu}_{nn'} = \nu_{nn'} + \sum_{m=1}^{M} \mathbb{1}[c_m = n]. \tag{9}$$

Sampling adjacency matrix:

$$p(A_{nn'} | \{s_m, c_m\}, A_{\neg nn'}, W, \theta, \{z_n\}) = exp\left\{ - \int_0^T \lambda_{n'}(t | \mathcal{H}_t) dt \right\} \prod_{m=1}^{M} \left[\lambda_{n'}(t | \mathcal{H}_t)^{\mathbb{1}[c_m = n']} \right] \cdot p(A_{nn'} | z_n, z_{n'}). \tag{10}$$

Sampling spontaneous attack rate:

$$p(\lambda_n|\{s_m, c_m\}, \alpha_\lambda, \beta_\lambda) = Gamma(\lambda_n|\tilde{\alpha}_n, \tilde{\beta}_n), \tag{11}$$

where

$$\tilde{\alpha}_n = \alpha_n + \sum_{m=1}^{M} \mathbb{1}[c_m = n]\mathbb{1}[\omega_m = 0], \tag{12}$$

$$\tilde{\beta}_n = \beta_n + T. \tag{13}$$

We present the intuitive explanations of the sampling formulas as follows. (1) Sampling Weight Matrix: the influence between n and n' is determined by the proportion of attack event of n', which can be attributed to attacker n, to the total attack event number of n; (2) Sampling Adjacent Matrix: the formula combines the prior network connection probability and the influence between n and n'. (3) Sampling spontaneous attack rate: the formula updates this variable as the averaged spontaneous attack event number during the observation window. Each attack parent variable ω_m is updated by Poisson thinning which can be deemed as sampling from a discrete distribution:

$$\omega_m \sim \{\lambda_{c_m}^{(0)}, \{h_{n\to c_m}(\Delta t)\}\}. \tag{14}$$

The overall inference scheme is illustrated in Algorithm 1.

Algorithm 1 Gibbs sampling algorithm.

Require:
collected attacker activities: $E = \{s_m, c_m\}$,
total attacker number: M,
observation time window: T,
model hyperparameters:$\{\kappa, \nu_{nn'}\}, \{\alpha_\lambda, \beta_\lambda\}$

Random initialize the A, W;
$iter \leftarrow 0$
while $iter < maxIt$ **do**
  for all all collected event m **do**
    sample event m' parent ω_m according to Equation 14
  end for
  for all each attacker n **do**
    sample attacker n's spontaneous attack rate λ_m according to Equation 11
  end for
  for all each each entry $W_{nn'}$ in influence matrix **do**
    sample $W_{nn'}$ according to Equation 10
  end for
  for all each each entry $A_{nn'}$ in adjacency matrix **do**
    sample $A_{nn'}$ according to Equation 11
  end for
  $iter \leftarrow it + 1$
end while
Compute the averaged A and W over samples after burn-in
return (A, W)

4.2. Graph-based clustering algorithm

Our goal of clustering is to group similar attack patterns, so as to provide a high-level modus operandi of global attack phenomena. It could be conducive to the administrators' understanding of the cluster structure of attackers and the temporal pattern of attacks.

As described in Section 2, there exist a plethora of clustering algorithms. K-Means is one of the most popular clustering algorithms due to its efficiency and simplicity. However, its main drawback is the requirement for the cluster number K to be specified before the algorithm is applied. Another classical clustering algorithm is the agglomerative hierarchical clustering. This algorithm starts with every single object in a single cluster. Then, in successive iteration, it agglomerates the closest pair of clusters by satisfying some similarity threshold, until all of the data is in one cluster. Likewise, it is difficult to define the threshold at which the hierarchical tree is cut [36].

The criteria of a proper clustering algorithm in our scenario is to tackle the trade-off between the diversity and accuracy of clustering results to good effect. We here select a graph-based clustering approach following [36] which is based on the influence weight matrix inferred by the attacker activity model. This algorithm works as follows: a candidate clique is

formed by commencing with the first attacker. The algorithm computes the links between the considered attacker and all the remaining ones. All the attackers, which are at least similar with this attacker in q, are merged into the current clique. If the clique size is at least 2 (or another threshold value defined by the user), the algorithm extracts the clique members from the data set, and saves them for later visual inspection. The procedure continues with the next available pattern in the reduced data set. The algorithm is shown in Algorithm 2 [36].

Algorithm 2 FindCliques(V,q).

$n \leftarrow V[0]$
$V \leftarrow V \backslash V[0]$
$sim[0] \leftarrow 1$
$i \leftarrow 0$
for $n' \in V$ **do**
  $i \leftarrow i + 1$
  $sim[i] \leftarrow W_{nn'}$
end for
$clique \leftarrow \{j | sim[j] < q, \forall j \in (1, \ldots, |sim|)\}$
$S \leftarrow S \cup clique$
$S \leftarrow FindCliques(V \backslash S, q)$
return (S)

It can be easily found that the graph-based algorithm possesses several advantages over other clustering techniques: firstly, neither the total number of cliques nor the clique size are needed prior to executing the algorithm; at the end, all cliques will have the expected quality level; and finally, this straightforward algorithm is of low computational complexity, and thus requires a short running time.

5. Security analysis with machine learning

The main objective of our framework is to obtain a realistic picture of the botnet structure on the Internet by analyzing the honeypot-collected attack data. Then, although the honeypot techniques eliminate the problem of distinguishing attackers from benign hosts, it is still highly challenging to detect the cluster structure of attackers due to the huge volume and diversity of collected data.

A multitude of Machine Learning techniques have been introduced to address these problems. Some of them are also evadable through encryption [20,26]. The essential idea of most previous research is to train a classifier based on selected features, and cluster the attackers based on feature similarities. Yet extensive features have been proven favorable for learning attack patterns, such as the geographical location of the attackers, the propagation scheme or the number of packets and bytes exchanged during the attack session. Our analysis draws upon the temporal information of attacks. The primary reason is that synchronization of the activities launched by a botnet is the most essential characteristic which can hardly be obscured by the spatial diversity, malware obfuscation or other camouflage techniques. On the other hand, feature engineering can be a laborious task. One can never be sure whether the selected set of features are adequate for analysis.

However, it is of great challenges to model and cluster the attackers based on their temporal patterns due to the variety of the temporal characteristics, including the duration of attacks, the total number of attacks, the interval distribution of attacks, lags between similar temporal patterns, jitters, etc. Then, most conventional algorithms or even the classical Machine Learning techniques like classification and clustering algorithms will fail to address this problem. The reason is that it is difficult to extract the proper feature set for constructing a classifier or performing a clustering algorithm.

We then consider the attack temporal data as sampling from a poisson process, and employ the Multivariate Hawkes Process, a mathematical tool for analyzing the interactions between poisson processes to measure the similarity between attack temporal data. These mathematical tools are assembled together by the Bayesian probabilistic graphical model, one of the most important techniques in the Machine Learning field. As a result, our framework can relieve the burden of extracting botnet features, compared with previous works built upon classifiers or clustering algorithms. Since the Bayesian probabilistic graphical model is an unsupervised Machine Learning method, the task of labeling training data is also eliminated from our framework. Another advantage of our framework, due to the unsupervised nature of the attacker activity model, is that new emerging attack temporal patterns can be quickly detected so long as the honeypot has collected enough attack data. It is for the reason that no static feature set is maintained in our framework which is commonly criticized in the security community for assuming a static profile of the botnet.

Another most concerned problem for clustering attackers in the same botnet is the robustness, since modern botnets tend to adopt increasingly stealthy ways to perform malicious activities that are extremely hard to observe in the network. These changes correspond to the feature variation in the Machine Learning techniques. Many traffic flow-based or group behavior-based approaches suffered tremendously from these adversarial techniques. However, our framework can alleviate this problem to some extent since we directly model the synchronization characteristic of the botnets instead of features.

Table 2
Dataset statistical characteristic.

Honeypot IP	Honeypot geolocation	Unique attacker count	Attack count
114.215.17.58	Beijing, China	337	1735
120.76.53.242	Hangzhou, China	819	30,204
122.112.235.239	Hangzhou, China	165	960
122.112.235.27	Hangzhou, China	611	4711
139.159.221.18	Shenzhen, China	355	4793
139.159.221.19	Shenzhen, China	333	7166
139.159.221.20	Shenzhen, China	291	8718
47.88.212.109	Singapore	1050	26,124
47.88.77.143	San Mateo, USA	833	37,368
47.89.26.43	Hongkong, China	1351	120,184

Finally, the detection efficiency also emerges as a significant issue for security community, due to great benefits in the timely detection of the attack pattern. The Gibbs sampling algorithm employed in our framework is, however, a time-consuming method for inference in the Bayesian probabilistic graphical model. Future work will enhance the efficiency by leveraging other fast inference algorithms without sacrificing accuracy. As the first step toward detecting the botnet by using only the temporal infromation effectively, we adopt the Gibbs sampling algorithm.

6. Experiments

6.1. Data collection

To capture the real-time adversarial access towards Industrial Control Systems/Supervisory Control and Data Acquisition System (ICS/SCADA), since November 2016, ten honeypots have been deployed on a world-wide scale using the reference in reliable clouds [27], e.g, Alibaba cloud environment. We reformulated the device information to realize a real-world camouflage of industrial device. These honeypots exposed multiple industrial protocols including Modbus, Siemens S7-Comm, Guardian_ast, Kamstrup_382, Bacnet, http and ipmi. The honeypot body is based on the Conpot implementation, an open-source implementation of ICS honeypot, since Conpot is a recent project in active development, providing a low-interaction ICS & SCADA honeypot which is designed to be easy to deploy and extend for large scale. The optimization module focuses on the functional expansion and anti-detection of Conpot. Each cloud server instance contains multiple honeypots equipped with routing strategy, firewall and Intrusion Detection System (IDS). If an attacker in the network passes through the firewall, it will interact with the specified honeypot instance, and its attack fingerprints will be collected and stored into the database. Each recorded attack consists of several pieces of contextual information, such as time, protocol, request, response, destination IP, source IP, country, subdivision, city and coordinate.

In the data preprocessing stage, we firstly aggregate the attack temporal data by adjusting each honeypot's time zone to China Standard Time. Then we attribute each attack event and its timestamp into the attack source's queue. Finally, the whole temporal data are translated to the origin.

6.2. Dataset analysis

In this paper, we select the attack data collected by our deployed honeypots during November 2016 to April 2017 as evaluation datasets. Overall, we collected 241,963 attacks during the analysis period. Similar to [3], the "heavy-hitters" phenomenon emerges as top 10 attackers were responsible for the 38.64% of the attacks. Table 2 lists some statistical characteristics of our dataset. Fig. 4 illustrates the overall aggregated and four representative attackers activities during the observation window. We can observe that each attacker's activity varies dramatically in attack duration and activity pattern. For example, attacker 1 reveals a continuous and relatively smooth activity curve, while attacker 2 shows a wave-like activity pattern. Attackers 3 and 4 manifest an ephemeral spike in the figure.

In this paper, we focus on the time interval distribution of attacks launched by an attacker rather than the aggregated time series of attack count. Fig. 5 illustrates the time interval distribution of attackers 1–4, respectively, in detail. In Fig. 5, the X-axis represents each attack event, and the Y-axis stands for the time when each attack occurs. It is obvious that there exists a one-to-one mapping between the aggregated attack time series and the time interval distribution. Specifically, attacker1's activity is composed of many relatively short yet evenly distributed waves during the whole observation window. Three pieces of curves in attacker2's activity correspond to its three waves in Fig. 4. Attackers 3 and4's densely distributed attack events in a relatively short time interval emerge as an ephemeral spike.

Intuitively, as described in Section 4.1, we can model each attacker's activity using Hawkes Process in which each start of a new attack wave is a response of its spontaneous attack intensity or other attacker's activity in the same group. And the subsequent attack activities in the wave is a response of the preceding attack events in this wave. Fig. 6 explicates the versatile simulation ability of Hawkes Process. We randomly select six attackers from our dataset, and simulate its temporal distribution by tuning the parameters in the Hawkes Process Model. It should be noted that each attacker's activity varies

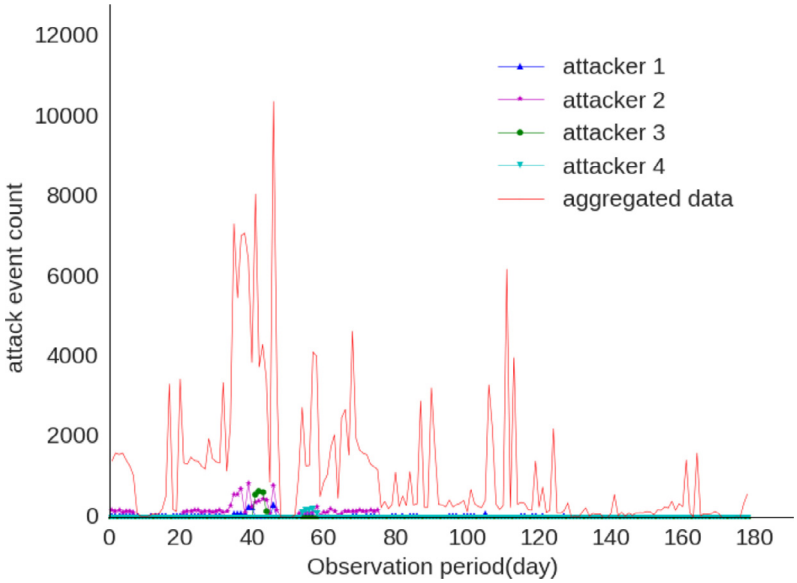

Fig. 4. Aggregated and four representative attackers' activities during the observation window.

Table 3
Comparison of predictive log likelihood.

Attacker activity model	Predictive log likelihood
Latent Distance Network Prior	**−9.50**
Multivariate Hawkes Process	−10.84
Erdos Renyi Network Prior	−11.20
Stochastic Block Network Prior	−16.56

in many aspects: duration of an attack, attack count, the discretization degree and the attack wave distribution. However, we can fit all these aspects well to some extent by simply tuning the parameters in the model.

6.3. Evaluation and results

We only consider the attackers who launched more than 30 attacks during the observation window. After this preprocessing step, the evaluation dataset includes 207,202 attacks performed by 342 attackers. We assume that only close enough attacks will be considered as coordinated activities in the same botnet. Then we set the maximum valid time interval as 100 s which indicates that attack intervals above this threshold will not be deemed to have latent influences.

We compare different classical graph priors [30] with the Latent Distance Model:

- Dense Model. This model assumes that each attacker connects with each other, and then the attacker activity model degenerates to the standard Multivariate Hawkes Process.
- Erdos-Renyi model. It is also known as Bernoulli model, and each connection is assumed as an independent and identically distributed random variable which takes the value 1 with p and the value 0 with probability $1 - p$.
- Stochastic Block Model. This model tends to produce graphs containing communities. Each attacker in this model is assumed to belong to one botnet. Then the connection probability between two attackers depends on whether they are in the same botnet.

Evaluation metrics: Predictive log likelihood is a widely accepted metric for measuring the fitness of the Bayesian probabilistic graphical model [17,29,30]. It is defined as the log likelihood on the test data. We split the first 80% of our dataset as the training set and the left 20% as the test set. The models are trained on the first 144 days (80% of the observation window) and tested on the following 36 days. For each model, we performed Gibbs sampling algorithm for 1000 iterations.

Results: In Fig. 7, we present how the predictive log likelihood of each model changes during the Gibbs sampling iterations. For illustration clarity, the predictive log likelihood is normalized by the number of events. As can be found in Fig. 7, after the very first few rounds (also known as the burn-in period), the predictive log likelihood of each model begins to stabilize around some fixed values. We drop the first half samples, and compute the average value of influence matrix W and predictive log likelihood over the second half of the samples. Table 3 compares the averaged predictive log likelihood of each model. According to Table 3, the attacker activity model with the latent distance network prior fits our dataset better than it does with Erdos Renyi prior, stochastic block prior and the standard Multivariate Hawkes Process, meaning that

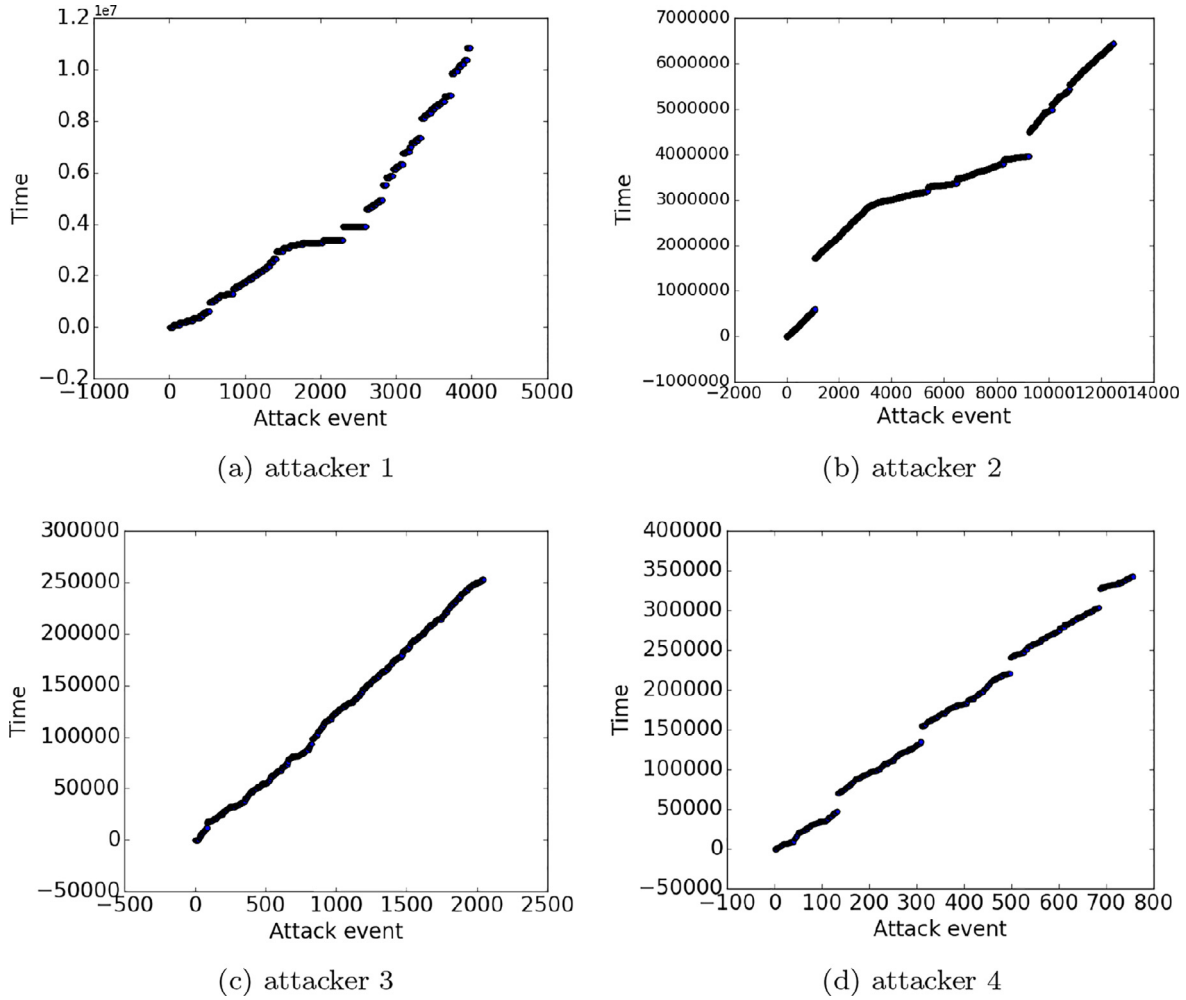

Fig. 5. Time interval distribution of attackers 1–4. The *X*-axis stands for the attack event, and the *Y*-axis represents the event time. The slope of each segment in the curve illustrates the frequency of the attack event occuring during the unit time window, and the gap between two segments in the figure illustrates that between two waves of attacks.

the sparsity prior introduced in the model leads to better predictive performance. The Erdos Renyi model is slightly worse than the Multivariate Hawkes Process. Nevertheless, the sparsity property gained by the network prior is comparable to the performance loss. However, the stochastic block network prior leads to a weird degradation of the predictive performance, which may be an interesting problem to be explored in the future.

To illustrate the fitness of the attacker activity model with the temporal data, we conduct another experiment as [29]. In Fig. 8, we demonstrate the Q–Q plot of the simulated activity temporal data with inferred Hawkes parameters versus the real dataset. The *X* and *Y*-axes represents the theoretical quantiles, and empirical quantiles, respectively. The Q–Q plot is a probability plot, which is a graphical method for comparing two probability distributions by plotting their quantiles against each other. If the two distributions being compared are similar, the points in the Q–Q plot will approximately lie on the diagonal. If the distributions are linearly related, the points in the Q–Q plot will approximately lie on a line, but not necessarily on the diagonal. As can be found in Fig. 8, our attacker activity model can fit the empirical data well to some extent.

For illustration, we applied Lambiotte's modularity algorithm to our dataset. In Fig. 9, we visualize the inferred latent network and the community structure. There are total 17 communities in the graph, among which 6 communities encompass 20–30 attackers, and others contain only about 5 attackers. Due to space limitations, we cannot provide all the details of the figure. To refine this clustering result, we then employ our graph-based clustering algorithm to yield a better result.

The first interesting point of our result is that attackers in some clusters tend to belong to the same IP subnet. This observation validates the effectiveness of our approach, as this characteristic of botnet has been studied in [10]. Fig. 10 illustrates four of these clusters. It should be noted that these attackers activity durations varied from 1,000,000 to 10,000,000 s,

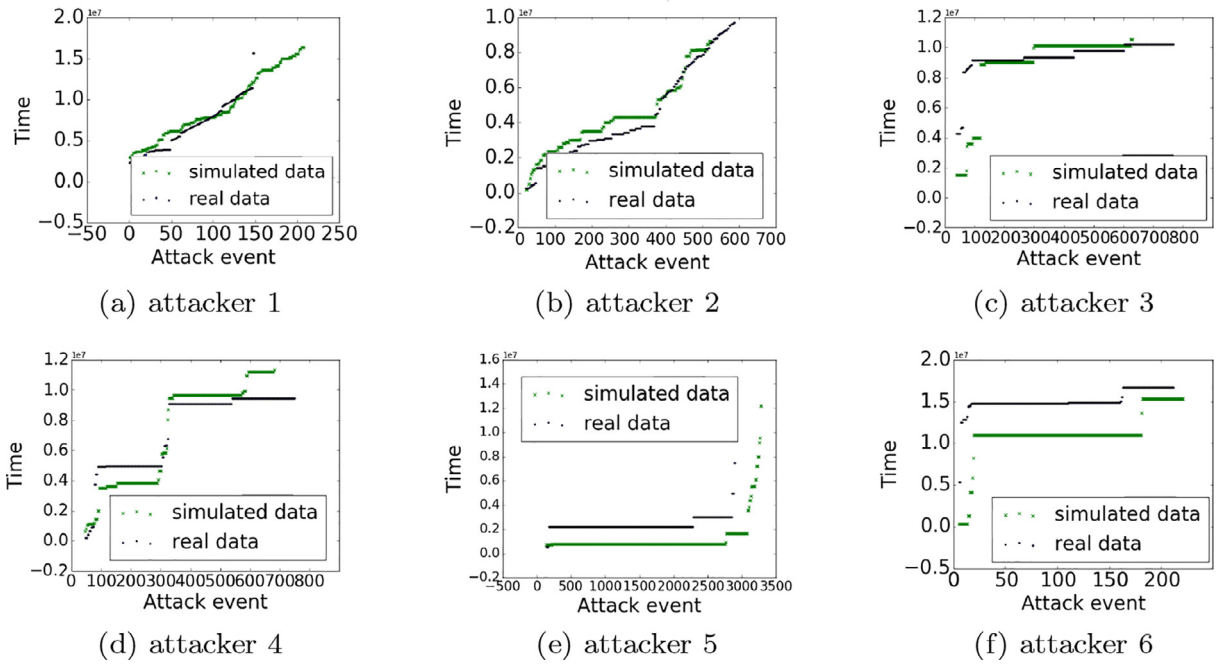

Fig. 6. Real and simulated time interval distribution of randomly selected six attackers.

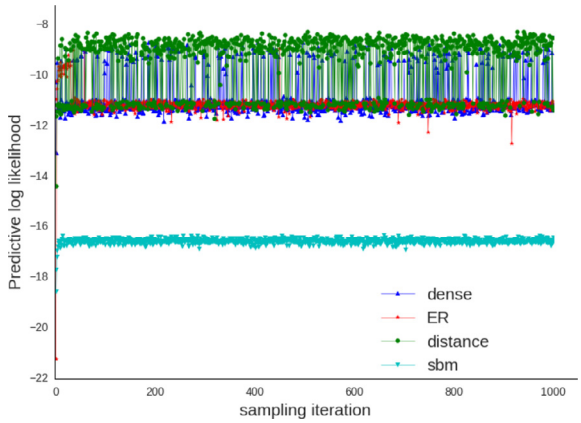

Fig. 7. Predictive log likelihood during the Gibbs sampling iterations.

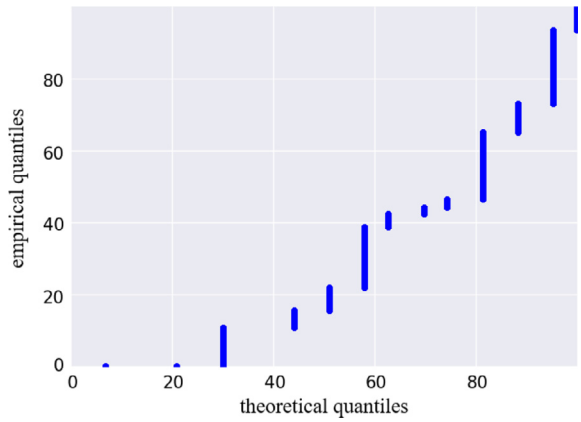

Fig. 8. The Q–Q plot of the simulated activity temporal data with inferred Hawkes parameters versus the real dataset.

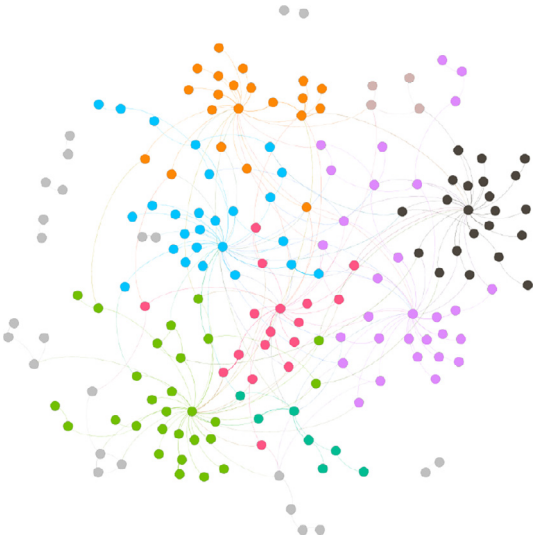

Fig. 9. Inferred latent network between attackers in our dataset.

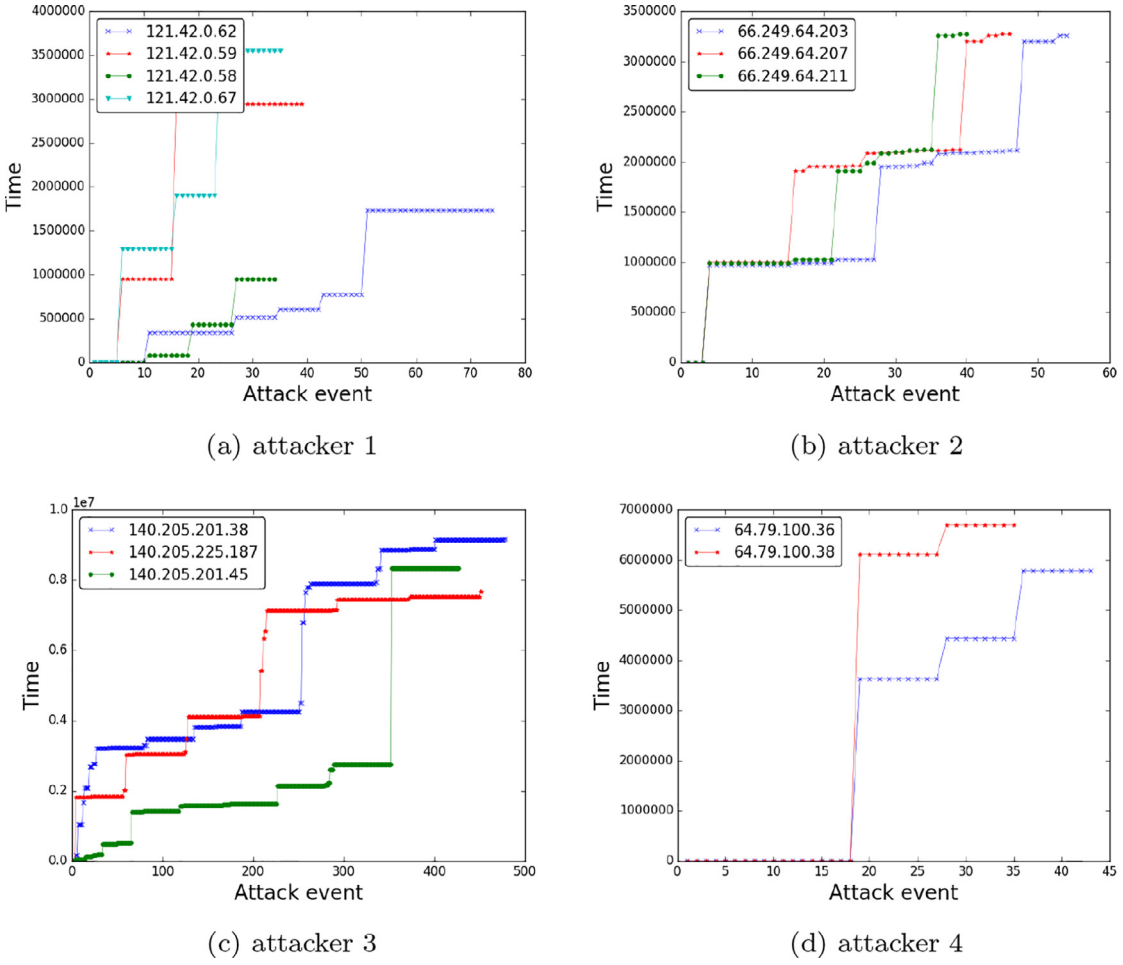

Fig. 10. Examples: Attackers in the same cluster tend to belong to the same IP subnet.

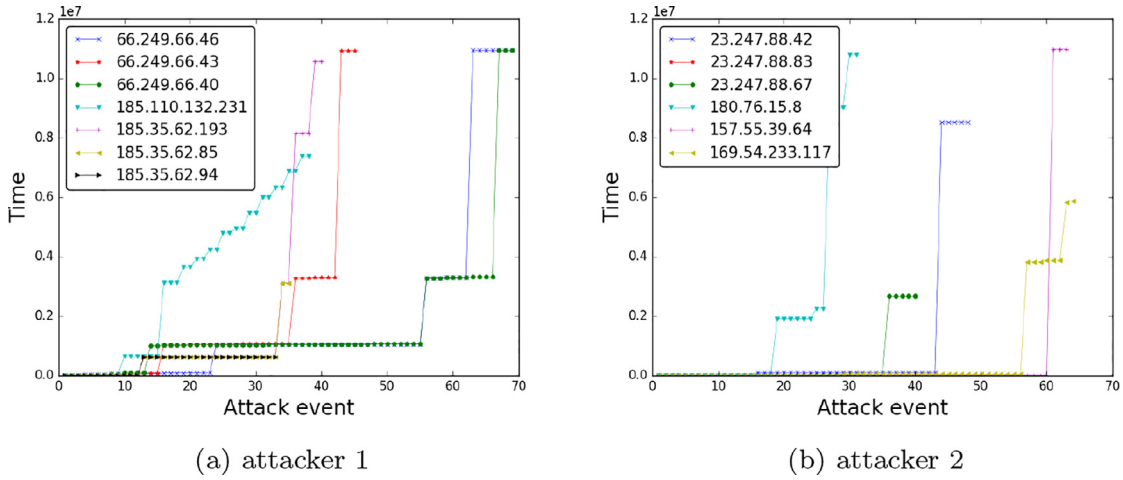

Fig. 11. Examples: attackers belonging to two IP subnets have similar temporal patterns.

the total event number varied from 45 to 500, and there exist lags and jitters between attackers. However, our framework can still detect the latent connections between them.

The second noteworthy observation is that some botnets contain attackers from different IP subnets. Fig. 11 illustrates two examples of these botnets. As can be found from the figures, after some simple transformations (translation, compression, etc.), these attacker sequences can be deemed as sampling from the same temporal distribution. This observation indicates that these two botnets include some attackers from one IP subnet and others from different regions of the Internet.

These two interesting observations serve as explications for the result 3 in which the latent distance network model fits the empirical data better than other modes do. According to [18], the Latent Distance Model encodes the belief that connection probability should decrease with distance between latent locations. Specifically, attacker's subnet can provide a good indication of the spatial uncleanliness of certain networks, i.e. the tendency for compromised hosts to stay clustered within unclean networks, especially for zombie machines belonging to botnets [10]. Intuitively, we expect these locations to mirror the true place fields, since nearby hosts in the same subnet are likely to have correlated attack event histories. Results in Figs. 10 and 11 reveal that hosts in the botnet display the tendency of spatial uncleanliness.

7. Conclusion and future work

In this work, a novel framework has been proposed for modeling and clustering attacker activity pattern based on the data collected from honeypots deployed on a world-wide scale. Our framework combines a Bayesian probabilistic graphical model and a graph-based clustering algorithm. The Bayesian probabilistic graphical model leverages the attacker activity temporal information with Multivariate Hawkes Process, and incorporates the sparse network prior knowledge by the latent distance network model. Basically, Multivariate Hawkes Process utilizes the mutual exciting property to identify the latent influence between attackers, while the Latent Distance Model captures the sparse interaction property. The graph-based clustering algorithm then refines the clustering result by eliminating the attackers that are not significantly similar to others.

To the best of our knowledge, this is the first method for analyzing attacker pattern without laborious feature engineering. Moreover, the graph-based clustering algorithm employed in our model requires neither prior knowledge on the number of clusters nor the cluster size. Therefore, our method is also of great generality. We evaluate the proposed framework on our collected dataset, and compare it with several other models. Experiments results show that our framework is able to effectively model and cluster the attackers based on their activity temporal information effectively.

There remain some defects to be resolved in our future work. One defect is that if the attackers launch the attack in an unsynchronized manner, the accuracy of the attack modeling might be affected, so some methods should be designed to merge the clusters detected by our current approach.

References

[1] S. Andreev, Y. Koucheryavy, Internet of things, smart spaces, and next generation networking, in: Proceedings of the 12th International Conference on Next Generation Teletraffic and Wired/Wireless Advanced Networking, NEW2AN, Springer, 2012.
[2] L. Atzori, A. Iera, G. Morabito, The internet of things: a survey, Comput. Netw. 54 (15) (2010) 2787–2805.
[3] A. Bar, B. Shapira, L. Rokach, M. Unger, Identifying attack propagation patterns in honeypots using Markov chains modeling and complex networks analysis, in: Proceedings of IEEE International Conference on Software Science, Technology and Engineering, 2016, pp. 28–36.
[4] E. Bertino, N. Islam, Botnets and internet of things security, Computer 50 (2) (2017) 76–79.
[5] M.Z.A. Bhuiyan, G. Wang, J. Wu, J. Cao, X. Liu, T. Wang, Dependable structural health monitoring using wireless sensor networks, IEEE Trans. Dependable Secure Comput. 14 (4) (2017) 363–376.
[6] Q. Cao, H. Shen, K. Cen, W. Ouyang, X. Cheng, Deephawkes: bridging the gap between prediction and understanding of information cascades, in: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, ACM, 2017, pp. 1149–1158.

[7] L. Chen, M. Tseng, X. Lian, Development of foundation models for internet of things, Front. Comput. Sci. China 4 (3) (2010) 376–385.
[8] J. Cheng, L. Adamic, P.A. Dow, J.M. Kleinberg, J. Leskovec, Can cascades be predicted? in: Proceedings of the 23rd International Conference on World Wide Web, ACM, 2014, pp. 925–936.
[9] E. Choi, N. Du, R. Chen, L. Song, J. Sun, Constructing disease network and temporal progression model via context-sensitive Hawkes process, in: Proceedings of the 2015 IEEE International Conference on Data Mining, ICDM, IEEE, 2015, pp. 721–726.
[10] M.P. Collins, T.J. Shimeall, S. Faber, J. Janies, R. Weaver, M.D. Shon, J. Kadane, Using uncleanliness to predict future botnet addresses, in: Proceedings of ACM SIGCOMM Conference on Internet Measurement 2007, San Diego, California, USA, October, 2007, pp. 93–104.
[11] N. Du, M. Farajtabar, A. Ahmed, A.J. Smola, L. Song, Dirichlet-Hawkes Processes with Applications to Clustering Continuous-Time Document Streams(2015) 219–228.
[12] S. Edwards, I. Profetis, Hajime: analysis of a decentralized internet worm for iot devices, Rapidity Netw. 16 (2016) https://security.rapiditynetworks.com/publications/2016-10-16/hajime.pdf.
[13] J. Etesami, N. Kiyavash, K. Zhang, K. Singhal, Learning Network of Multivariate Hawkes Processes: A Time Series Approach, in: Alexander Ihler, Dominik Janzing (Eds.), Proceedings of the Thirty-Second Conference on Uncertainty in Artificial Intelligence (UAI'16), AUAI Press, Arlington, Virginia, United States, 2016, pp. 162–171.
[14] C. Fernandez-Gago, F. Moyano, J. Lopez, Modelling trust dynamics in the internet of things, Inf. Sci. 396 (2017) 72–82.
[15] S. Garca, M. Grill, J. Stiborek, A. Zunino, An empirical comparison of botnet detection methods, Comput. Secur. 45 (2014) 100–123.
[16] N. Goodman, A Survey of Advances in Botnet Technologies (2017) arXiv:1702.01132.
[17] X. He, T. Rekatsinas, J. Foulds, L. Getoor, Y. Liu, HawkesTopic: A Joint Model for Network Inference and Topic Modeling from Text-Based Cascades2015871–880.
[18] P. Hoff, Modeling homophily and stochastic equivalence in symmetric relational data, in: Advances in Neural Information Processing Systems, Curran Associates Inc., 2008, pp. 657–664.
[19] P.D. Hoff, Modeling homophily and stochastic equivalence in symmetric relational data, in: Proceedings of International Conference on Neural Information Processing Systems, 2007, pp. 657–664.
[20] Z. Huang, S. Liu, X. Mao, K. Chen, J. Li, Insight of the protection for data security under selective opening attacks, Inf. Sci. 412 (2017) 223–241.
[21] J.F.C. Kingman, Poisson Processes, Wiley Online Library, 1993.
[22] C. Kolias, G. Kambourakis, A. Stavrou, J. Voas, Ddos in the iot: mirai and other botnets, Computer 50 (7) (2017) 80–84.
[23] P.J. Laub, T. Taimre, P.K. Pollett, Hawkes processes, Quant. Financ. 48 (2) (2015).
[24] C. Li, J. Ma, X. Guo, Q. Mei, Deepcas: an end-to-end predictor of information cascades, in: Proceedings of the 26th International Conference on World Wide Web, International World Wide Web Conferences Steering Committee, 2017, pp. 577–586.
[25] H. Li, W. Hao, W. Gan, C. Gang, Survey of probabilistic graphical models, in: Proceedings of Web Information System and Application Conference, 2014, pp. 275–280.
[26] J. Li, Y. Zhang, X. Chen, Y. Xiang, Secure attribute-based data sharing for resource-limited users in cloud computing, Comput. Secur. 72 (2018) 1–12.
[27] J. Li, Towards an efficient snapshot approach for virtual machines in clouds, Inf. Sci. 379 (2017) 3–22.
[28] K. Li, C. Liu, X. Cui, POSTER: a lightweight unknown HTTP botnets detecting and characterizing system, in: Proceedings of ACM Sigsac Conference on Computer and Communications Security, 2014, pp. 1454–1456.
[29] L. Li, H. Deng, A. Dong, Y. Chang, H. Zha, Identifying and Labeling Search Tasks via Query-based Hawkes Processes2014731–740.
[30] S.W. Linderman, R.P. Adams, Discovering latent network structure in point process data, Comput. Sci. 32 (2014) 1413–1421.
[31] C. Mavroforakis, I. Valera, M. Gomez-Rodriguez, Modeling the Dynamics of Online Learning Activity on the Web (2016) 1421–1430.
[32] S. Nagaraja, Botyacc: unified P2P botnet detection using behavioural analysis and graph analysis, in: Proceedings of European Symposium on Research in Computer Security, Springer, 2014, pp. 439–456.
[33] S. Nagaraja, P. Mittal, C.-Y. Hong, M. Caesar, N. Borisov, BotGrep: finding P2P bots with structured graph analysis, in: Proceedings of Usenix Security Symposium, 2010, pp. 95–110.
[34] M. Nawrocki, M. Wählisch, T.C. Schmidt, C. Keil, J. Schönfelder, A Survey on Honeypot Software and Data Analysis (2016) arXiv:1608.06249.
[35] K. Singh, S.C. Guntuku, A. Thakur, C. Hota, Big data analytics framework for peer-to-peer botnet detection using random forests, Inf. Sci. 278 (19) (2014) 488–497.
[36] O. Thonnard, M. Dacier, A framework for attack patterns' discovery in honeynet data, Digital Invest. 5 (2008) S128–S139.
[37] T.J. Mitchell, J.J. Beauchamp, Bayesian variable selection in linear regression, J. Am. Stat. Assoc. 83 (404) (1988) 1023–1032.
[38] D. Xu, Y. Tian, A comprehensive survey of clustering algorithms, Ann. Data Sci. 2 (2) (2015) 165–193.
[39] Q. Yan, Y. Zheng, T. Jiang, W. Lou, Y.T. Hou, PeerClean: unveiling peer-to-peer botnets through dynamic group behavior analysis, in: Computer Communications, 2015, pp. 316–324.
[40] S.-H. Yang, H. Zha, Mixture of mutually exciting processes for viral diffusion, in: Proceedings of International Conference on Machine Learning, 2013, pp. 1–9.
[41] T.F. Yen, M.K. Reiter, Are your hosts trading or plotting? Telling P2P file-sharing and bots apart, in: Proceedings of IEEE International Conference on Distributed Computing Systems, 2010, pp. 241–252.
[42] J. Zhang, R. Perdisci, W. Lee, U. Sarfraz, X. Luo, Detecting stealthy P2P botnets using statistical ttraffic fingerprints, in: Proceedings of IEEE/IFIP International Conference on Dependable Systems & Networks, 2011, pp. 121–132.
[43] D. Zhao, I. Traore, B. Sayed, W. Lu, S. Saad, A. Ghorbani, Botnet detection based on traffic behavior analysis and flow intervals, J. Comput. Secur. 39 (2013).
[44] Q. Zhao, M.A. Erdogdu, H.Y. He, A. Rajaraman, J. Leskovec, Seismic: a self-exciting point process model for predicting tweet popularity, in: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2015, pp. 1513–1522.