Digital Signal Processing ••• (••••) •••–•••

Contents lists available at ScienceDirect

Digital Signal Processing

www.elsevier.com/locate/dsp

A Bayesian change point model for detecting SIP-based DDoS attacks

Barış Kurt [a,*], Çağatay Yıldız [a], Taha Yusuf Ceritli [a], Bülent Sankur [b], Ali Taylan Cemgil [a]

[a] Department of Computer Engineering, Bogazici University, 34342 Bebek, Istanbul, Turkey
[b] Department of Electrical and Electronics Engineering, Bogazici University, 34342 Bebek, Istanbul, Turkey

ARTICLE INFO

Article history:
Available online xxxx

Keywords:
VoIP security
SIP
DDoS
Simulation
Bayesian change point models

ABSTRACT

Session Initiation Protocol (SIP), as one the most common signaling mechanism for Voice Over Internet Protocol (VoIP) applications, is a popular target for the flooding-based Distributed Denial of Service (DDoS) attacks. In this paper, we propose a DDoS attack detection framework based on the Bayesian multiple change model, which can detect different types of flooding attacks. Additionally, we propose a probabilistic SIP network simulation system that provides a test environment for network security tools.

1. Introduction

VoIP is the technology of carrying voice and multimedia communications through the Internet Protocol (IP) networks. Due to its multimedia support and low infrastructure cost, VoIP systems are worldwide taking over circuit-switched telephone networks. With the introduction of 5G, the VoIP is predicted to become the dominant methodology for voice and multimedia communications.

The VoIP systems transfer voice and multimedia data between communicating parties through the packet-switched IP networks based on data transfer protocols, such as the Real-time Transport Protocol (RTP). In addition, they require session-level signaling protocols for managing their communication sessions. Considering its lightweight nature, simplicity and ease of implementation, the SIP [1] is one of the most popular open standard signaling protocols designed for VoIP. SIP provides signaling functions necessary to register clients, check their locations and availability, exchange information on their data transmission capabilities, and provide handshakes necessary for connection setups.

Despite all their attractive features, the downside is that VoIP systems are more vulnerable to security threats compared to their circuit switched predecessors. There are two basic sources of security threats for VoIP systems. Firstly, VoIP systems are affected by all the lower protocol layer threats, e.g., layer threats. Secondly, VoIP systems using open standards protocols suffer from many protocols-specific vulnerabilities, in other words, they are prone to security threats specifically designed to exploit the vulnerabilities of the underlying signaling protocols [2], [3]. These protocol-specific attacks are usually not classified as network attacks by the conventional network-level security systems. Therefore, VoIP systems need extra security mechanisms for detecting and preventing VoIP specific attacks.

One of the most frequently observed type of cyber-attack is the DDoS flooding attack [4], which is typically realized by sending a vast amount of network protocol messages to a victim. These types of attacks aim to exploit the weaknesses in the SIP protocol or faults due to some poor implementation. An example of such DDoS flooding attack is the *INVITE* attack. In this case, the attacker tries to set up communication with many SIP users by sending INVITE requests to the SIP proxy server. The server, which maintains a table for each SIP session, holds an entry for each INVITE request and awaits response from the call receiver for a fixed amount of time. Eventually, the server reaches its memory capacity while trying to keep track of an excessive amount of connections. Another typical DDoS attack is the SYN-flooding [5], where a target network proxy is forced to maintain a barrage of Transmission Control Protocol (TCP) sessions, and eventually becomes unresponsive due to overutilization of its resources. Thus DDoS flooding attacks aim to cripple a target system by overusing and eventually depleting its resources, such as bandwidth, CPU or memory, and making it unable to respond to the requests of its legitimate subscribers.

DDoS attacks can have negative impact on business since a target system cannot provide services to its customers during attacks. The downtime of servers creates revenue loss and reputation damage, which in turn leads to loss of revenue as well, for service providers. Furthermore, the productivity of workforce is reduced as employees cannot use affected systems for operations. Among victims of DDoS attacks, well-known companies can be found. For instance, GitHub was under attack for six days [6]. Another victim of such attacks was BBC where an online DDoS tool named BangStresser, which delivers attacks as a service, might be used [7].

* Corresponding author.
E-mail address: bariskurt@gmail.com (B. Kurt).

https://doi.org/10.1016/j.dsp.2017.10.009
1051-2004/

A recent survey reports an increase in DDoS attacks, arguing that it might be a possible result of the proliferation of cheap and easy-to-launch attack tools [8]. According to another report [9], the number of attacks decreases while the average peak attack size increases. For attacks targeting SIP based VoIP systems, there has been an upward trend as well [10]. Defense strategies for these common DDoS attacks have been studied extensively [11,12].

Many network security systems have been developed for the detection of SIP-based DDoS attacks [13]. The majority of these systems uses supervised methods, such as thresholding [14] and rule-based pattern matching, as in [15,16]. These supervised methods require a training phase for learning patterns for each type of attack and for building a dictionary of known attacks. Attack detection is based on finding matching patterns between the current network state with one of the known attack patterns in the training set. However, when an unprecedented attack occurs, a supervised system can easily fail to detect it, since the pattern of the new attack will possibly be different than all the learned attack patterns. It becomes imperative then to re-train the system by an extended training data set which includes samples from the new attack traffic.

In this paper, we focus on the detection of SIP-specific DDoS flooding attacks [3,17]. We aim therefore to develop a more robust and generalizable DDoS monitor based on anomaly detection principles. Anomaly detection [18] is an unsupervised methodology where the system is programmed to recognize significant deviations from its learned data patterns, and mark them as anomalous events. In our case, anomalous events are interpreted and marked, subject to further analysis, as security threats. We assume that the SIP server state has a stationary behavior under the so-called normal, "non-attack" SIP traffic, but that these statistics will change noticeably under a DDoS flooding attack. To sense these attacks, we have designed our feature vectors as consisting of a combination of incoming and outgoing SIP message counts plus the vector of resource usage measurements of the SIP proxy software. Our DDoS monitor is based on the Bayesian change point model [19] which models the normal SIP server behavior and infers changes that are possibly due to the DDoS attacks.

Collecting real-world VoIP network traces and annotating them without violating the privacy of the users is a tedious task. Therefore, for the proof of our concept, we conducted our experiments in a simulated environment. We developed a real-time SIP network simulator system, which models a social network for a group of users. The simulator generates actual voice conversation calls by setting up SIP sessions between users through a SIP proxy server. Our DDoS detection mechanism is deployed next to the SIP proxy server, so that it does not track RTP traffic between users. Therefore, the simulated SIP sessions are silent communications, i.e., actual data transfer via RTP is not generated. We generate DDoS attacks with the help of a commercial network vulnerability scanning tool Nova-VSpy [20], simultaneously with the VoIP simulation.

The contributions of our work can be listed as:

- We develop a Bayesian change point model for detecting SIP-oriented DDoS attacks. The proposed framework extends and generalizes the previous change-point based detection methods. Our change point-based DDoS monitor can be customized with different server parameters and different probabilistic observation models.
- A real-time SIP network traffic simulator based on social network modeling is developed and the software made publicly available. The proposed framework is tested with real-time data generated by the simulator, interleaved with DDoS attack data generated by a commercial network vulnerability scanning tool.

1.1. Paper structure

The remainder of this paper is organized as follows. Section 2 presents previous studies on SIP DDoS detection and change point models. Section 3 presents a SIP network terminology and details of the protocol. Section 4 presents our change point model in details. Section 5 describes the experimental setup we used to evaluate our methodology. The experimental results are given in Section 6. Finally, Section 7, evaluates results of the experiments, and draws conclusions.

2. Related work

There are comprehensive literature surveys on vulnerabilities of the SIP protocol [2], VoIP security research [3], DoS attacks targeting SIP networks [17], and security systems to counter SIP-based DoS attacks [13]. One of the earliest and simplest attempts to prevent single-source DoS flooding attacks in SIP systems was proposed by Iancu [14] where a rate limiter is deployed at the server to limit per-host SIP traffic. More elaborate methods were proposed to detect both single and distributed DoS attacks employing rule-based schemes, statistical methods, anomaly detection approaches, and machine learning tools.

Rule-based methods maintain a list of rules, or protocol finite state machines, and check the current server state against consistent patterns described in the rule set [15,21–23]. Ormazabal et al. [24] propose a large scale SIP firewall solution by combining several rule-based filters and attack mitigation mechanisms. While such rule-based systems are useful in detecting DoS attacks, they require carefully designed and perpetually updated rule books and fine-tuned thresholds. Since these systems can easily miss a novel attack whose descriptive rule has not yet been learned, they need to be reinforced with additional tools based on statistical approaches.

Machine learning methods were proposed as an alternative to rule-based and statistical methods for DDoS flooding detection, including support vector machines [25], evolutionary algorithms [26], naive Bayes, and decision trees [27]. Tsiatsikas et al. [28,29] give a comparison of 5 supervised classifiers and conclude that these methods provide good results on low-rate DoS attacks with little classification time overhead. Inherently, the success of supervised algorithms depends on the quality of the data set used during their training. For example in [29], authors employ different training sets for different basic scenarios. Obtaining such high quality training data can be difficult in a real world implementation of a supervised system. In contrast, we propose an unsupervised system, with an optional training phase to optimize its parameters. We show that setting those parameters empirically with the help of domain expertise is sufficient.

Reynolds and Ghosal [30] were the first to propose applying change point detection in SIP networks. They present a cumulative sum (CUMSUM) algorithm in order to detect INVITE flooding. Later, Rebahi and Sisalem [31] have developed a parametric version of this algorithm. Zhang et al. [32] proposed to use additional features to enhance the accuracy of CUMSUM. Geneiatakis et al. [33] propose bloom filters to efficiently track incomplete SIP sessions and to raise an alarm if these exceed a certain threshold. The major disadvantage of these algorithms is that one needs to engineer different sets of features in order to detect different types of flooding attacks.

The works closest to our approach are the distance-based anomaly detection methods [34,35], where a distance metric is used to measure the dissimilarity between the distributions of normal and observed traffic features. If the distance between the normal and observed distributions is above a threshold, an alarm is generated. Similar to our approach, these methods can be used

Please cite this article in press as: B. Kurt et al., A Bayesian change point model for detecting SIP-based DDoS attacks, Digit. Signal Process. (2017), https://doi.org/10.1016/j.dsp.2017.10.009

to detect any type of network attack provided that full SIP message histogram is included in the feature set. Our method extends and generalizes these anomaly detection methods by introducing Bayesian framework, which models the SIP server state with a set of features that incorporates both network traffic and SIP server resource usage data. In our method, the attack decision relies on a robust posterior probability calculation rather than simple thresholding. To our best knowledge, this work presents the first Bayesian framework tailored specifically to model a SIP server in order to detect SIP anomalies, hence fills an important gap in the literature.

3. SIP network traffic

3.1. SIP terminology

SIP is designed to initiate, modify and terminate communication sessions among agents. Four general types of SIP entities are defined in RFC 3261 [1]: user agents, proxy servers, redirect servers and registrars. A user agent (UA) is the endpoint entity that generate and receive SIP messages. In a typical SIP session, UA's communicate by sending request and response messages to each other. The registrar is responsible for registering the UA's, and storing their location information. The registered UA's communicate with each other via the intermediary of proxy servers. The proxy servers deliver the request and response messages between UA's. Finally, the redirect servers allow proxy servers to communicate with other servers from external domains.

The SIP messages are divided into two basic categories: SIP requests and SIP responses. Each SIP request sent by an UA is answered by a corresponding SIP response. For examples, a UA can make a REGISTER request to the registrar in order to get online, make an INVITE request to another UA to start a call, or make a BYE request to terminate an ongoing conversation. A SIP response message generated for a request can be from one of the 6 SIP response categories: 1xx-provisional, 2xx-success, 3xx-redirection, 4xx-client, 5xx-server error or 6xx-global failure. For example, a UA may response with a 200-OK message for accepting an incoming request.

3.2. An example message flow

An illustrative example of SIP message communication is given in Fig. 1. It shows the flow of exchanged messages between a server and two users during a normal call. In this scenario, Alice initiates a call to Bob by sending an *INVITE* packet to the SIP Server. After authentication, the SIP server forwards this request to Bob. Similarly, the response of Bob, in this case *ACK* packet showing that the call is accepted, is transmitted to Alice through the SIP server. At the end, *BYE* messages terminate the conversation between Alice and Bob.

Once a SIP session is established, two endpoints start exchanging multimedia data such as audio conversations, video streams, etc. Recall that SIP, being a signaling protocol, is not involved in the multimedia data exchange between agents. Handshake on the kind and encoding type of data, on the address and ports to be used for transfer, and other details regarding the data exchange is usually achieved using Session Description Protocol (SDP) [36]. Additionally, real-time media delivery relies on RTP [37].

Real-world SIP packet exchange scenarios usually involve more than two servers and two agents. The above call setup case is illustrative but simplistic. For example, it does not specify how the server reaches out to the caller. A setup in which Alice and Bob are not registered to the same server would require a location server and the re-transmission of the INVITE message. Similarly, other features supported by SIP – such as call transfer, call park, conference – lead to distinct call flows. In summary, SIP message traffic data can be quite complex.

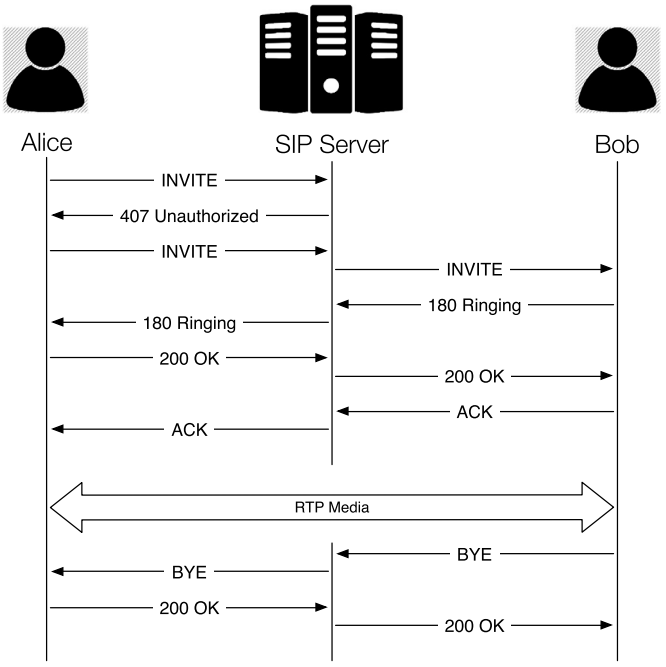

Fig. 1. A call scenario in SIP network.

3.3. DDoS attacks in SIP networks

DDoS flooding attacks could rapidly affect network traffic characteristics and cause service degradations. Their impact is contingent on the attack parameters and differs substantially from one attack to another. A DDoS detection method is expected to signal an attack, in principle practically independent of its configuration. Therefore, DDoS detector must be robust and highly sensitive, that is, with high detection rate and low probability of false alarm under a wide range of realistic network conditions.

Mirkovic et al. [4] classified DDoS attack mechanisms on the basis of its impact on the victim. First, one would expect an increase in the incoming network traffic – even beyond the server's bandwidth – as a result of a flooding attack. The severity of this increase, however, is directly related to the resources the attacker possesses and cannot be forecast beforehand. Second, the flooding rate does not necessarily stay the same throughout the attack. Mirkovic et al. also noted that slow rate boosts, i.e., creeping attacks typically result in detection latency.

Another significant parameter is the SIP packet type used in the flooding. Typical DDoS scenarios consider a SIP server being flooded by one type of packet such as INVITE, SUBSCRIBE, or BYE. Nevertheless, DDoS attacks can also be performed using a judiciously selected mixture of SIP requests. Thus, intelligently mounted schemes such as slow boost attacks, multi-SIP packet attacks, multi-agent attacks that try to obfuscate their synchronism by time jittering require more advanced defense mechanisms and concomitantly more computation time and power. Despite these variabilities, the DDoS shield is expected to have low latency in order to timely initiate attack prevention.

4. Methodology

In this work, we use a Bayesian approach for the detection of abrupt changes in SIP traffic that could possibly correspond to a DDoS attack. This approach is based on a hierarchical probability model, more precisely a hidden Markov model, that relates

Please cite this article in press as: B. Kurt et al., A Bayesian change point model for detecting SIP-based DDoS attacks, Digit. Signal Process. (2017), https://doi.org/10.1016/j.dsp.2017.10.009

observed features from network packet traffic and server load measurements (such as CPU usage and system calls) to hidden variables. These hidden variables indicate the state of the system, e.g., change and no-change, along with other hidden dynamical quantities. Formally, we will refer to the features or observations as v (visible) and the state change indicators as s. Other variables that are not of direct relevance for our problem, but are needed for describing the state of the dynamical system will be referred as h (hidden). Once the model is specified, the inferential goal is to calculate the posterior probability

$$p(s|v) = \frac{p(v|s)p(s)}{p(v)} \propto \int p(v|h,s)p(h|s)p(s)dh$$

Apart from specific special cases, the above integral is intractable. Fortunately for the change-point problem, we can describe a model where exact calculation becomes possible for relatively short time sequences. We show that, with a rather simple approximation heuristics such as pruning, we can get a constant-space algorithm that can be feasibly implemented in real time.

In the sequel, we first present the way the observations (v) are composed from the features collected from the SIP server; then we describe the probability model which is an instance of a Bayesian change point model, and the way this model can be used for online estimation of DDOS attacks. Finally, we give details of our approximation and provide a complexity analysis.

4.1. SIP server features as observations

The first step in a Bayesian approach is to provide a probabilistic generative model for the observations collected from the system. However, before going into the mathematical details of our generative model, let us first give a clear definition of the observations. As we continuously monitor a SIP server, we collect real-time statistics for a period of Δt and form an observation vector v_t as a summary of the statistics collected during that period. v_t is an N dimensional vector composed of the number of SIP request and response messages, server log messages, server statistics such as number of TCP connections, together with the CPU and memory usage measured at the end. The complete list of the features collected from the system is given in Table 3 and explained in further detail in Section 5.

4.2. Multiple change point model

The multiple change point model is a special form of hierarchical Markov models [19], where the observations conditionally depend on latent states, and the states either follow the previous regime or jump randomly to a new one. As far as network monitoring is concerned, these regime changes imply anomalous events, and which may be related to some security threats. The generative equations of the multiple change point model can be given as

$$h_0 \sim \Omega(h_0; w) \tag{1}$$

$$s_t \sim [s_t = 0]\pi + [s_t = 1](1-\pi) \tag{2}$$

$$h_t|s_t, h_{t-1} \sim [s_t = 0]\delta(h_t - h_{t-1}) + [s_t = 1]\Omega(h_t; w) \tag{3}$$

$$v_t|h_t \sim \Theta(v_t; h_t) \tag{4}$$

where δ is Dirac delta function.

The observation v_t, at time t, is assumed to be a random variable sampled from a $\Theta(v; h)$ distribution with an unknown parameter h_t. Initially, h_0 is drawn from a $\Omega(h; w)$ distribution. Afterwards, at each time instance t, h_t is either re-drawn from the same initial distribution or set to the previous value h_{t-1}. The decision for change is given by a Bernoulli random variable s_t. The model allows h_t to change as many times as required during the run of the algorithm. The graphical representation of the multiple change point model is given in Fig. 2.

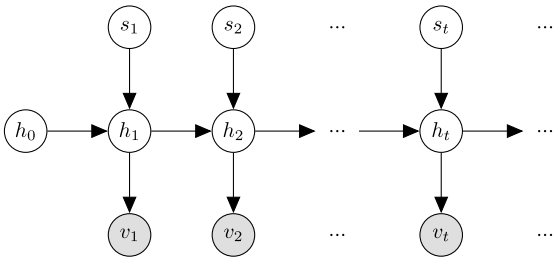

Fig. 2. Bayesian change point graphical model.

Our DDoS detection system includes a monitoring unit for observing and collecting network traffic data as well as SIP server activities. The monitoring unit collects and compiles network and server statistics into an observation vector, i.e., a feature vector, v_t at each $\sim \Delta t$ (1 second) time interval, as the resume of events that have occurred in the SIP server during that last observation interval. For each such feature vector, the model infers whether the observation vector is generated by the previous regime, that is $s_t = 0$ and $h_t = h_{t-1}$, or whether the server state has jumped to a new regime, which means $s_t = 1$ and $h_t \sim \Omega(h_t; w)$. The observation model Θ and its prior distribution Ω are selected according to the features collected from the server. The details of the data features and the distributions used in the change point model are given in Section 4.3.

The prior probability of change, π, and the parameters w of the prior distribution $\Omega(h; w)$ are the hyperparameters of our model. Provided that these hyperparameters are known, and the system is fully observable, meaning that the change points $s_{1:T}$, hidden states $h_{1:T}$ and observations $v_{1:T}$ are known, we can calculate the full joint likelihood as follows:

$$p(s_{1:T}, h_{0:T}, v_{1:T}) = p(h_0)\prod_{t=1}^{T} p(s_t)p(h_t|h_{t-1}, s_t)p(v_t|h_t) \tag{5}$$

In reality, the change point events $s_{1:T}$ and the hidden states $h_{1:T}$ are not observed, and the problem of detecting a change point event at time t is formulated as calculating the posterior probability $p(s_t = 1|v_{1:T})$. From the Bayes rule, we can write

$$p(s_t|v_{1:T}) = \frac{p(v_{1:T}, s_t)}{p(v_{1:T})} \propto p(v_{1:T}, s_t) \tag{6}$$

The probability of change at time t can be inferred online by calculating the filtering distribution $p(s_t|v_{1:t})$, or in an offline manner by the smoothing distribution $p(s_t|v_{1:T})$. The calculations can be done efficiently via the recursive *Forward–Backward* algorithm [38]. The filtering density is calculated by the forward recursion of the α messages:

$$\alpha(s_t, h_t) \equiv p(s_t, h_t, v_{1:t}) \tag{7}$$

$$= \sum_{s_{t-1}} \int_{h_{t-1}} p(h_t|h_{t-1}, s_t)\alpha(s_{t-1}, h_{t-1}) \times p(v_t|h_t) \times p(s_t) \tag{8}$$

Then, the change probability is calculated as

$$p(s_t|v_{1:t}) \propto p(s_t, v_{1:t}) = \int_{h_t} \alpha(s_t, h_t) \tag{9}$$

In an offline setting, where we can calculate decisions using the full observations of the time series $v_{1:T}$, we can smooth the

Please cite this article in press as: B. Kurt et al., A Bayesian change point model for detecting SIP-based DDoS attacks, Digit. Signal Process. (2017), https://doi.org/10.1016/j.dsp.2017.10.009

filtering distribution with backward recursions to get a stronger estimate $p(s_t|v_{1:T})$. The backward recursion can be written as

$$\beta(s_t, h_t) \equiv p(v_{t+1:T}|s_t, h_t) \tag{10}$$

$$= \sum_{s_{t+1}} \int_{h_{t+1}} p(h_{t+1}|h_t, s_{t+1})\beta(s_{t+1}, h_{t+1}) \times p(v_t|h_t) \times p(s_t) \tag{11}$$

The smoothed density is calculated as

$$p(s_t|v_{1:T}) \propto \int_{h_t} p(s_t, h_t, v_{1:t})p(v_{t+1:T}|s_t, h_t) \tag{12}$$

$$= \int_{h_t} \alpha(s_t, h_t)\beta(s_t, h_t) \tag{13}$$

Real-time anomaly detection tracks streaming data, so that processing the $v_{1:T}$ observation sequence is not feasible in practice since T is not bounded. Furthermore, anomaly detection is a time-critical task, which implies that the change points must be recognized as soon as possible. Therefore, calculating a smoothing distribution is feasible only if the system is allowed to make change point decisions deferred by a fixed amount of time L, which is called the *lag*. In such a case, the process is called *fixed-lag smoothing*, where the change point inference for s_t is done at time $t + L$ by calculating the density $p(s_t|v_{1:t+L})$ in lieu of $p(s_t|v_{1:T})$. It is important to note that this process requires calculating a backward recursion for L steps starting at each time point $t + L$, and this increases the processing complexity.

4.3. DDoS detection via multiple change point model

We have described a multiple change point model with arbitrary hidden state distribution Ω and observation model Θ, with the assumption that Ω is the conjugate prior of Θ for computational simplicity. Now we assign actual probability distributions for the hidden state and observation models.

We let the observation model Θ be a coupled distribution of multinomial and Poisson distributions. Multinomial distribution is used to model the ratios of the magnitudes of the signals ratios in an observed vectors. On the other hand, Poisson distribution models the magnitudes of the tracked signals. Without loss of generality, we assume that the features whose ratios will be modeled are stored in the first M positions of the observation vector v, denoted as $v_{1:M}$ and the remaining $N - M$ positions are filled with the features whose magnitudes are modeled. Then, we can write the observation model as

$$\Theta(v) = \mathcal{M}(v_{1:M}; p) \times \prod_{i=M+1}^{N} \mathcal{P}(v_i; \lambda_i) \tag{14}$$

where the multinomial and Poisson distributions are defined as

$$\mathcal{M}(x; p) = \frac{\Gamma(\sum_i x_i + 1)}{\prod_i \Gamma(x_i + 1)} \prod_i p_i^{x_i} \tag{15}$$

$$\mathcal{P}(x; \lambda) = \frac{\lambda^x e^{-\lambda}}{\Gamma(x + 1)} \tag{16}$$

In this setup, the hidden state vectors $h = (p; \lambda)$ are the respective parameters of the multinomial and Poisson distributions. Since the prior distribution of multinomial is Dirichlet distribution and that of the Poisson is the Gamma distribution, the prior of our state vector h becomes the product of these conjugate priors, namely Dirichlet and Gamma, and can be written as:

Table 1
Model variables and parameters.

Variable	Description
$s_{1:T}$	Reset switches
$h_{1:T}$	Hidden state vectors
$v_{1:T}$	Observation vectors
π	Reset probability
Ω	Prior distribution of hidden states
w	Parameters of the Ω distribution
Θ	Observation model
α	Dirichlet distribution parameter
a, b	Gamma distribution parameters
$\Gamma()$	Gamma function

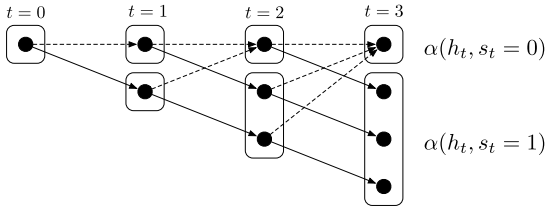

Fig. 3. Expansion in the forward variable messages.

$$\Omega(p, \lambda) = \mathcal{D}ir(p; \alpha) \times \prod_{i=M+1}^{N} \mathcal{G}(\lambda_i; a_i, b_i) \tag{17}$$

The Dirichlet and Gamma distributions are given as

$$\mathcal{D}ir(p; \alpha) = \frac{\Gamma\left(\sum_{i=1}^{M} \alpha_i\right)}{\sum_{i=1}^{M} \Gamma(\alpha_i)} \prod_{i=1}^{M} p_i^{\alpha_i - 1} \tag{18}$$

$$\mathcal{G}(\lambda; a, b) = \frac{b^a}{\Gamma(a)} \lambda^{a-1} e^{-b\lambda} \tag{19}$$

where α is an M dimensional vector and a and b are $N - M$ dimensional vectors such that each $\{a_i, b_i\}$ is a hyper-parameter for Gamma distribution. Hence, the hyper-parameters w of the model is the set $w = (\alpha, a, b)$. The complete set of model variables and parameters are given in Table 1.

4.4. Implementation details and complexity analysis

The inference for the change point model requires calculating the $\alpha(s_t, h_t)$ and $\beta(s_t, h_t)$ messages at the end of each observation period. We simply need to store a table of probabilities for each $\alpha(s_t = i, h_t = j)$, such that $i \in \{0, 1\}$ and $j \in Dom(\Omega)$ and update this table according to the equations in (8). The same is true for the β messages. When the hidden state distribution has continuous domain, we have to express the α and β messages as mixtures of Ω potentials. An Ω potential is described as

$$\phi(p, \lambda) = \exp(l)\Omega(p, \lambda; \alpha, a, b) \tag{20}$$

where l is the logarithm of the normalizing constant, and α, a and b are the parameters of the reset and observation distributions, respectively.

At each time step, the switching variable attains one of the two values, therefore, an additional Ω potential is added to the α and β messages to indicate the change potential. $\alpha(h_t, s_t = 1)$ is a single potential and $\alpha(h_t, s_t = 0)$ is a mixture of t potentials transferred from the previous α message. This linear growth of the α messages for the forward recursion is illustrated in Fig. 3. Details of the operations required to implement forward and backward recursions are presented in Appendix A.

This linear growth is not sustainable for online continuous tracking of the server state. One has to limit, then, the number

Please cite this article in press as: B. Kurt et al., A Bayesian change point model for detecting SIP-based DDoS attacks, Digit. Signal Process. (2017), https://doi.org/10.1016/j.dsp.2017.10.009

Algorithm 1 Bayesian Change Point Detection.

```
function BCPM(π, w, v_{1:T}, LAG, THRESHOLD)
    alpha ← [ ]
    for t = 1...T do
        alpha_p = PREDICT(alpha, π, w)
        alpha = UPDATE(alpha_p, v_t)
        if LAG > 0 then
            beta = BACKWARDFILTER(π, w, v_{t+1−LAG:t})
            gamma = SMOOTH(alpha, beta)
            cpp = COMPUTECPP(gamma, len(beta))
        else
            cpp = COMPUTECPP(alpha, 1)
        end if
        if cpp > THRESHOLD then
            ALARM()
        end if
    end for
end function
```

Table 2
Average run time of the algorithm.

Routine	Time (μs)
PREDICT	35
UPDATE	648
BACKWARD_FILTER	17
SMOOTH	1400
COMPUTE_CPP	7
Total	2107

of mixture potentials in the forward message by K, indicating the maximum number of components. Once a message reaches the maximum number of allowed components, at each subsequent step, the component with the minimum normalizing constant is pruned. Therefore, in the worst case, an α message has K components and a β message L components, since we had decided to run the backward-recursion for only L steps. It follows then that, during filtering, at most K observation updates are required and during the smoothing operations, where we multiply an α message with a β message, $K \times L$ multiplications are performed. Therefore, the number of operations at any time instance is $O(KL)$. We empirically set $K = 100$, and the lag parameter $L = 5$. In our experiment setup, using more than 100 potentials had no significant contribution, and while a lag value of 5 significantly improved the accuracy of the system, bigger lag values did not yield much of an improvement.

4.5. Real time analysis

Algorithm 1 presents the offline version of the main detection loop of our algorithm. Here, offline is in the sense that the whole data set is available at the beginning of the algorithm. In the online version, the single loop in the algorithm will be run exactly once after each observation period. We time the individual functions of the algorithm, whose descriptions are also given in Appendix D. The actual run time of the algorithm depends on the number of features used, the value of the lag L, and maximum number of components K. In this measurement we set $L = 5$, $K = 100$ and used all available features. The algorithm is coded with C++ and experiments are run offline, on an INTEL i7 CPU @2.7 GHz on a data sequence of 2000 observations. We can see from Table 2 that one iteration of the main loop executes in 2 ms (2107 μs) on the average, which allows online deployment of our algorithm.

4.6. Parameter learning

During the inference stage, we had assumed that the hyper-parameters of our multiple change point model, namely the reset probability π and the latent state prior parameters w were given. In practice, these parameters must be set to appropriate values for accurate change point estimation. For a small number of parameters, a grid search method can give good parameter estimates; however for large models, i.e., for large dimensional w, the search method is not applicable. Thus, we use a maximum likelihood approach to find the best hyper-parameters as a function of observations. Given observations $v_{1:T}$, we would like to find the parameters that maximize the log likelihood

$$\mathcal{L}_{\pi,w}(v_{1:T}) \equiv \log p(v_{1:T}|\pi, w) \tag{21}$$

$$= \log \sum_{s_{1:T}} \int_{h_{0:T}} p(s_{1:T}, h_{0:T}, v_{1:T}, |\pi, w) \tag{22}$$

Since this log likelihood expression is intractable due to the summation over latent parameters, we employ an iterative Expectation-Maximization (EM) scheme to find the $\{\pi, w\}$ estimates. By Jensen's inequality, the log likelihood is lower bounded as

$$\mathcal{L}_{\pi,w}(v_{1:T}) \geq \langle \log p(s_{1:T}, h_{0:T}, v_{1:T}, |\pi, w)\rangle_{q(z)} - \langle \log q(z)\rangle_{q(z)} \tag{23}$$

This bound is tight for $q(z) = p(s_{1:T}, h_{0:T}|v_{1:T}, \pi, w)$. The log-likelihood can then be maximized iteratively as follows:

E-Step:

$$q(z)^{new} = p(s_{1:T}, h_{0:T}|v_{1:T}, \pi^{old}, w^{old}) \tag{24}$$

M-Step:

$$(\pi^{new}, w^{new}) = \arg\max_{\pi,w} \left\langle p(s_{1:T}, h_{0:T}|v_{1:T}, \pi^{old}, w^{old})\right\rangle_{q(z)^{new}} \tag{25}$$

The detailed derivations of the EM algorithm for Dirichlet-Multinomial and Gamma-Poisson change point potentials are given in Appendix B.

5. Experimental setup

5.1. Data generation

Our data generator, detailed in [39], is made up of four distinct modules: (1) a SIP server, (2) a traffic simulator, (3) a DDoS attack generator and (4) a network traffic monitor. As a registrar and SIP proxy server, we have used an Asterisk-based PBX software named *Trixbox* [40]. To mimic the normal message traffic on a SIP server, we have built *Boun-Sim* [39], a probabilistic SIP network simulation tool that generates calls between a number of SIP endpoint entities in real time. Concurrently with the normal traffic simulation, a rich variety of DDoS attacks were generated by a commercial vulnerability scanning tool, called *NOVA V-Spy* [20]. The fourth and final component in the setup is the network monitor, a module that tracks the server, extracts and delivers features to the change point monitor.

Our simulation tool is driven by a probabilistic generative model to recreate typical user behaviors, such as making calls, answering, rejecting or ignoring an incoming call, and holding on an ongoing call. User actions generated by the Simulator are realized as actual SIP communications, where SIP messages are exchanged between UA's and Trixbox. Simulator omits the *RTP* messages carrying the actual conversational data between users, since these do not pass through the SIP server, and are not relevant to the outcome of the simulation. Details of the Simulator parameters are presented in Appendix C.

Note that the simulator parameters should be set according to the capacity of *Trixbox* as number of calls per second may prevent

Please cite this article in press as: B. Kurt et al., A Bayesian change point model for detecting SIP-based DDoS attacks, Digit. Signal Process. (2017), https://doi.org/10.1016/j.dsp.2017.10.009

server from handling the traffic generated by the simulator *BounSim*. A detailed performance analysis of Asterisk server of version 1.6 can be found in [41] where the performance is degraded after the number of simultaneous calls exceeds 600.

5.2. Data traces

We conduct experiments on four different simulated data sets. Our data set generating mechanism is controlled by two bi-level variables, one for network traffic intensity, the other for attack intensity, and each can be either set as *low* or *high*. To set the network traffic intensity, we tune the call rate parameters of the users since phone calls constitute the main source of the network traffic. The average number of SIP packets passing through the SIP server per second in low and high data sets are 75 and 90, respectively. To set the flood rate, we change the number of network packets delivered from V-Spy to the server in each second. In a low attack setting, the server is flooded by 100 packets per second, whereas 500 packets are used in high attack settings. In the sequel, we refer to these data sets as *LOW–LOW*, *LOW–HIGH*, *HIGH–LOW* and *HIGH–HIGH*, where the two adjectives qualify, in order, the network traffic intensity and the flood rate. All simulations are realized by 500 active users registered to the server.

In order to demonstrate the robustness of our change point model, we tested it with a dataset consisting of 40 different DDoS attacks. These attacks are generated by tuning the following options of V-Spy:

- **Attack Type:** We flood the server with five different SIP request packets, randomly chosen from among *REGISTER*, *INVITE*, *OPTIONS*, *CANCEL* and *BYE* requests. Each type of attack generates different types of changes in SIP server state.
- **Transport Protocol:** Since SIP operates independently of the transport protocol, we generated attacks over both TCP and User Datagram Protocol (UDP).
- **Fluctuation:** Nova V-Spy can generate floods with both constant and fluctuating rates. In half of our experiments, we generated floods with fluctuating rates.
- **Content Size:** Nova V-Spy can optionally insert dummy strings to the end of SIP messages, which must also be within the capability of the attack detector since this manipulation affects the bandwidth consumption.

An example attack could be an "*INVITE* attack through UDP port without any fluctuation and with large content size". The attacks last for about 20 seconds and we have left an interval of at least 25 seconds between two consecutive attacks; this results in a simulation sequence of around half an hour duration in order for the 40 attacks to occur. IP addresses and the user id's of attackers were randomly chosen and the attacker terminals were unregistered throughout the simulation.

5.3. Data features

Table 3 shows the features monitored for DDoS attack detection. We can divide the feature space roughly into five categories, the first two categories collected from server's network connection side, and the last three categories collected from server's resource management side. The first two categories consist of a variety of packet types in a SIP network: *SIP Requests* and *SIP Responses* (Fig. 4); these packet types have different well-defined roles [1]. Notice that the actual features used in the detection model are the count statistics or histogram of these message type occurrences within an observation interval (e.g., 1 sec). The underlying assumption here is that a significant change in the pattern of SIP message histograms is a direct reflection of messaging traffic behavior, indicating possibly an anomaly, i.e., an attack.

The other three feature categories are entitled as *Resource Usage*, *Asterisk Stats* and *Asterisk Logs*. The first of these consists of the pair of CPU usage and memory usage of the virtual machine in which Trixbox is installed. The second one is made up of features that reflect the load created by Asterisk. The last category counts the keywords in the log files generated by Asterisk. We conjecture that all these features would diverge from their average values in the case of an attack and hence potentially qualify as anomaly indicators.

5.4. Evaluation

We measure the performance of our DDoS monitor on the basis of the F-score, which is defined as the harmonic mean of the precision (P) and recall (R) measures. The F-score gets closer to 1 when both precision and recall are close to 1, and corresponding to good performance; conversely, the F-score diminishes to 0, when the system performs poorly either due to low precision or low recall or both.

$$\text{F-score} = 2 \times \frac{P \times R}{P + R} \tag{26}$$

$$\text{Precision (P)} = \frac{\#\text{ true alarms}}{\#\text{ alarms}} = \frac{T_a}{T_a + F_a} \tag{27}$$

$$\text{Recall (R)} = \frac{\#\text{ true alarms}}{\#\text{ ground truth}} = \frac{T_a}{G_a} \tag{28}$$

where T_a and F_a are the true alarms (true positive) and false alarms (false positive), and G_a is the true number of change points. An alarm g_t means signaling of a change event, and it is triggered whenever the change point probability in Eq. (14) at time t exceeds a certain threshold λ_a.

$$g_t = \begin{cases} 0 & \text{if } p(s_t | v_{1:t+L}) < \lambda_a \\ 1 & \text{otherwise} \end{cases} \tag{29}$$

The true and false alarms are calculated by matching the alarms $g_{1:T}$, with the ground truth of change events $\hat{g}_{1:T}$. In the design of our experiment, the time stamps for the beginning of attacks are manually set, but the actual effect of an attack is observed with some delay due to the combined emergent behavior of the SIP server, simulation and the vulnerability scanning tool Nova V-Spy. Therefore we set the ground truths as attack time stamps which are initially set and adjust them manually afterwards.

We declare a correct detection if the alarm g_i is within a tolerance vicinity of the corresponding ground truth event $\hat{g}_j$, that's $|i - j| < w$, and increment the number of true alarms. Alarms not matched with any ground truth are regarded as false positives.

6. Results

We evaluate the performance of our proposed DDoS monitor with model simulation data generated by various input feature combinations.We test exhaustively the 5 feature categories in various category combinations, i.e., including them or not, into Poisson and Multinomial cases, respectively. This results in a total of $3^5 - 1 = 242$ observation models, where each observation model corresponds to one particular instance of category combination.

The hyper-parameters for each observation model need to be adjusted for getting best F-scores. For this purpose, we first perform a grid search inside the parameter space. Since grid search is feasible for only a limited number of parameters, we use shared parameters for the priors of the Dirichlet and Gamma distributions. In this setup, we assign a single parameter α for the Dirichlet priors by setting $w_D = [\alpha, \alpha, \ldots, \alpha]$ and a single parameter a for all

Please cite this article in press as: B. Kurt et al., A Bayesian change point model for detecting SIP-based DDoS attacks, Digit. Signal Process. (2017), https://doi.org/10.1016/j.dsp.2017.10.009

Table 3
The five categories of features collected from the network side and resource side of the SIP server.

Category	Feature	Description
SIP Requests	REGISTER	Num. of "register" requests
	INVITE	Num. of "invite" requests
	SUBSCRIBE	Num. of "subscription" requests
	NOTIFY	Num. of "notification" requests
	OPTIONS	Num. of "options" requests
	ACK	Num. of "acknowledgment" requests
	BYE	Num. of "bye" requests
	CANCEL	Num. of "cancellation" requests
	PRACK	Num. of "provisional acknowledgement" requests
	PUBLISH	Num. of "event publish" requests
	INFO	Num. of "information update" requests
	REFER	Num. of "call transfer" requests
	MESSAGE	Num. of "instant message" requests
	UPDATE	Num. of "session state update" requests
SIP Responses	100	Num. of trying responses
	180	Num. of "ringing" responses
	183	Num. of "session progress" responses
	200	Num. of "success" responses
	400	Num. of "bad request" errors
	401	Num. of "unauthorized" errors
	403	Num. of "forbidden" errors
	404	Num. of "not found" errors
	405	Num. of "not allowed" errors
	481	Num. of "dialog does not exist" errors
	486	Num. of "busy" errors
	487	Num. of "request terminated" errors
	500	Num. of "server internal" errors
	603	Num. of "decline" errors
Resource Usage	TOT_CPU	Percentage of total CPU usage
	TOT_MEM	Percentage of total virtual memory usage
Asterisk Stats	A_CPU	Percentage of CPU used by Asterisk
	MEM	Percentage of physical memory utilized by Asterisk
	FH	Num. of Asterisk file descriptors
	THREADS	Num. of Asterisk threads
	TCP_CONN	Num. of Asterisk TCP connections
	UDP_CONN	Num. of Asterisk UDP connections
Asterisk Logs	A_WARNING	Num. of Asterisk "warning" log messages
	NOTICE	Num. of Asterisk "notice" log messages
	VERBOSE	Num. of Asterisk "verbose" log messages
	ERROR	Num. of Asterisk "error" log messages
	DEBUG	Num. of Asterisk "debug" log messages

Table 4
Grid search space.

Parameter	Search values
α	$1, 10, 100$
a	$1, 10, 100$
π	$10^{-2}, 10^{-4}, 10^{-8}$

Gamma priors. We also set the scale parameter of Gamma priors, $b = 1$. The search space is given in Table 4.

The configurations with the best average F-scores on 4 different traces after the grid search are reported in Tables 5, 6 and 7. Table 5 presents the results when change point probabilities are computed online (using only forward recursion). In order to test the conjecture that deferred change point decisions should yield better performance, we run online smoothing algorithm (see Table 6). Since the grid search may not be feasible for bigger dimensional vectors, we also develop a maximum likelihood scheme for estimating the hyper-parameters. To this effect, we employ the EM algorithm described in Section 4.6 for the case of models that has attained the highest F-scores according to the grid search. The results are given in Table 7.

From Tables 5–7 one can observe that *SIP Requests* contribute to the feature set for all cases, hence they seem to be the most important features collected from data. Second in importance, the *Resource Usage* features help improving the accuracy of our system. We also notice that the Dirichlet-Multinomial (DM) model usually gives better accuracy than the Poisson-Gamma model. Furthermore, as average F-scores are Table 6 are greater than those in Table 5, we deduce that the accuracy of change estimations increases provided we allow deferred change point decision with a lag value of $L = 5$ seconds. Increasing the lag further enhances the results only very slightly whereas the cost of latency in the attack signal may become prohibitive.

We observe that the maximization of the hyper-parameters with respect to their likelihood under the proposed models does not necessarily maximize the F-scores; in other words, the F-scores obtained after the maximum likelihood estimation of hyper-parameter values are below the scores obtained by the grid search. We conjecture that this may be due to the mismatch between the model and the actual data, and will be the subject of future research.

6.1. Comparison to a distance based method

We employed a simple distance based method for classifying the normal and attack traffic in our data set, based on the previous works [34] and [35]. In this method, we use the Hellinger distance between a normal traffic feature vector p, which is learned from the data, and traffic vectors q_t, collected at each time instance t.

Please cite this article in press as: B. Kurt et al., A Bayesian change point model for detecting SIP-based DDoS attacks, Digit. Signal Process. (2017), https://doi.org/10.1016/j.dsp.2017.10.009

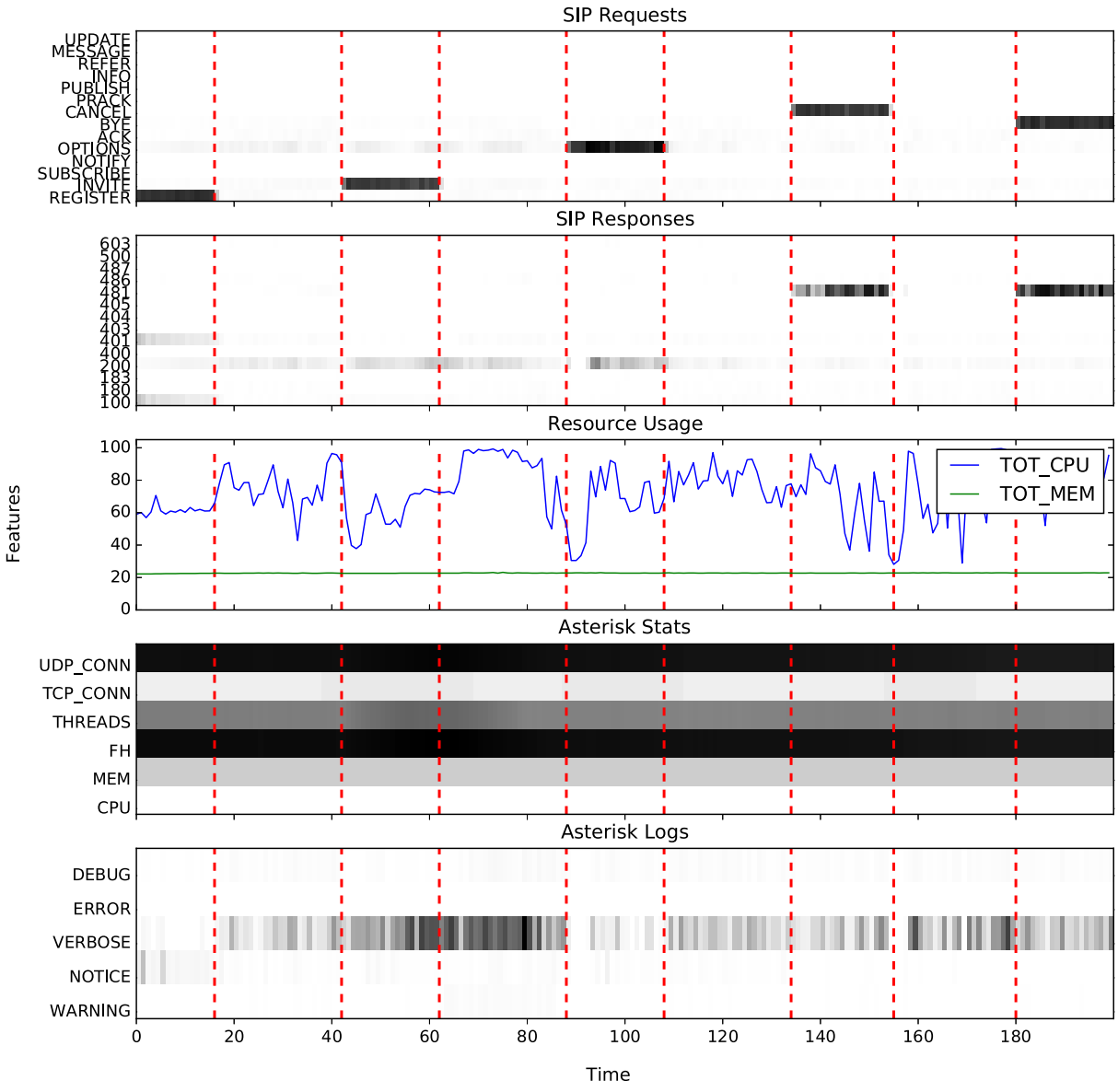

Fig. 4. Histogram of features from the HIGH–HIGH data set for 200 seconds.

Table 5
Best models after the grid search with scores collected by filtering.

Features					Traffic								F-score (Avg.)
SIP Requests	SIP Responses	Resource Usage	Asterisk Stats	Asterisk Logs	Low–Low		Low–High		High–Low		High–High		
					P	R	P	R	P	R	P	R	
DM		DM			0.86	0.83	0.93	0.95	0.85	0.94	0.99	0.94	0.91
PG		DM			0.94	0.88	0.93	0.95	0.88	0.71	0.95	1.00	0.90
PG					0.91	0.88	0.92	0.95	0.88	0.72	0.95	1.00	0.90
PG				DM	0.89	0.86	0.90	0.95	0.89	0.71	0.98	0.98	0.89
PG	DM				0.92	0.83	0.89	0.95	0.90	0.66	0.96	1.00	0.88
PG	DM	DM			0.92	0.82	0.94	0.93	0.90	0.65	0.95	1.00	0.88
DM					0.90	0.77	0.91	0.89	0.82	0.82	0.96	0.94	0.88
DM			DM		0.94	0.74	0.91	0.96	0.89	0.64	0.98	0.98	0.87
DM	DM	DM			0.76	0.88	0.94	0.89	0.85	0.80	0.88	0.98	0.87
DM	DM				0.81	0.83	0.84	0.94	0.86	0.82	0.90	0.95	0.87
Average					0.89	0.83	0.91	0.94	0.87	0.75	0.95	0.98	0.89

Please cite this article in press as: B. Kurt et al., A Bayesian change point model for detecting SIP-based DDoS attacks, Digit. Signal Process. (2017), https://doi.org/10.1016/j.dsp.2017.10.009

Table 6
Best models after the grid search with scores collected by online smoothing.

Features					Traffic								F-score (Avg.)
SIP Requests	SIP Responses	Resource Usage	Asterisk Stats	Asterisk Logs	Low–Low		Low–High		High–Low		High–High		
					P	R	P	R	P	R	P	R	
DM		PG	DM		0.91	0.96	0.93	0.96	0.93	0.94	1.00	0.98	0.95
PG			DM		0.96	0.93	0.96	0.95	0.90	0.91	1.00	0.98	0.95
PG				DM	0.93	0.95	0.92	0.98	0.92	0.91	1.00	0.98	0.95
PG		PG	DM		0.94	0.94	0.96	0.95	0.90	0.91	1.00	0.98	0.95
PG		DM		DM	0.89	0.96	0.95	0.96	0.87	0.95	1.00	0.98	0.95
PG		PG		DM	0.91	0.98	0.93	0.98	0.89	0.90	1.00	0.98	0.94
PG	DM	PG			0.91	0.96	0.94	0.95	0.88	0.93	1.00	0.98	0.94
PG		DM			0.90	0.96	0.94	0.95	0.90	0.90	1.00	0.98	0.94
PG		PG			0.92	0.95	0.94	0.95	0.93	0.85	1.00	0.98	0.94
DM		DM			0.89	0.93	0.95	0.96	0.90	0.94	0.96	0.98	0.94
Average					0.92	0.95	0.94	0.96	0.90	0.91	1.00	0.98	0.94

Table 7
Best models according to maximum likelihood parameter estimation.

Features					Traffic								F-score (Avg.)
SIP Requests	SIP Responses	Resource Usage	Asterisk Stats	Asterisk Logs	Low–Low		Low–High		High–Low		High–High		
					P	R	P	R	P	R	P	R	
PG			DM		0.85	0.94	0.78	0.98	0.80	0.94	0.85	1.00	0.88
PG		PG	DM		0.76	0.94	0.77	1.00	0.81	0.93	0.80	1.00	0.87
PG		DM			0.68	0.96	0.72	0.98	0.80	0.91	0.77	1.00	0.84
DM		PG	DM		0.68	0.98	0.66	0.98	0.76	0.97	0.68	1.00	0.81
PG				DM	0.68	0.96	0.64	0.98	0.71	0.94	0.71	1.00	0.80
PG		PG			0.53	0.95	0.69	0.98	0.75	0.90	0.71	1.00	0.78
PG		PG		DM	0.54	0.95	0.63	1.00	0.69	0.94	0.66	1.00	0.76
PG	DM	PG			0.50	0.95	0.66	1.00	0.71	0.94	0.65	1.00	0.76
PG		DM		DM	0.30	0.99	0.31	1.00	0.36	0.95	0.41	1.00	0.51
Average					0.61	0.96	0.65	0.99	0.71	0.93	0.69	1.00	0.78

During the training phase, we randomly select time instances $i \in \mathcal{I}$, from the attack-free part of the data, and calculate p as

$$p_k = \frac{\sum_{i\in\mathcal{I}} v_{i,k}}{\sum_{i\in\mathcal{I}}\sum_{k=1}^{K} v_{i,k}}$$

where K is the cardinality of the feature vector, which is made up of the *SIPRequests* and *SIPResponses* from Table 3. The Hellinger distance at each time t is calculated as

$$hd_t = \frac{1}{2}\sum_{k=1}^{K}(\sqrt{(p_k)} - \sqrt{(q_{t,k})})^2$$

We classify the traffic v_t as attack if the distance hd_t is greater than a threshold τ. Fig. 5 shows the ROC curves of this method for varying τ values, together with the true and false positive rates of our best models we found by maximum likelihood estimation (Table 7). When the Hellinger distance based methods reach our true positive rate, their false positive rates reach 15 to 20%. Also, while both methods have lower scores on low attack traffic sets, the decrease in the accuracy of the distance based methods are higher.

7. Conclusion

In this work, we have proposed a Bayesian multiple change point model for detecting DDoS flooding attacks in VoIP networks, which use SIP as their signaling mechanism. We have provided a framework that can work with different types of input features monitored at a SIP proxy server.The framework tracks the network traffic and SIP server behavior and raises an alarm whenever a change in those behaviors is detected.

Additionally, we have developed a probabilistic SIP network simulation system, which can generate real-time SIP messaging sequences to establish telephone connections in a PBX community. The simulation system provides a realistic test environment for SIP security tools, in case of the absence of real world test data. This simulation system is also made publicly available.

We tested our proposed system with traffic generated by the SIP network simulator together with DDoS attacks generated by a commercial network vulnerability scanning tool, Nova-VSpy.

Acknowledgments

This study is a Bogazici University – NETAS Nova V-Gate collaboration and funded by TEYDEB-3140701 project “Realization of Anomaly Detection and Prevention with Learning System Architectures, Quality Improvement, High Rate Service Availability and Rich Services in a VoIP Firewall Product”, by the Scientific and Technological Research Council of Turkey (TUBITAK).

Appendix A. Implementation of forward–backward algorithm

As noted earlier, in order to compute (fixed lag) smoothed density, forward and backward messages must be multiplied. The good news is multiplication of two Ω potentials is yet another Ω potential since the same applies to Dirichlet and Gamma potentials as we show below:

$$\begin{aligned} & c_1\mathcal{D}ir(p;\alpha) \times c_2\mathcal{D}ir(p;\beta) \\ &= c_1c_2\frac{\Gamma\left(\sum_{i=1}^{K}\alpha_i\right)}{\prod_{i=1}^{K}\Gamma(\alpha_i)}\prod_{i=1}^{K}p_i^{\alpha_i-1}\frac{\Gamma\left(\sum_{i=1}^{K}\beta_i\right)}{\prod_{i=1}^{K}\Gamma(\beta_i)}\prod_{i=1}^{K}p_i^{\beta_i-1} \\ &= c_1c_2\frac{\Gamma\left(\sum_{i=1}^{K}\alpha_i\right)}{\prod_{i=1}^{K}\Gamma(\alpha_i)}\frac{\Gamma\left(\sum_{i=1}^{K}\beta_i\right)}{\prod_{i=1}^{K}\Gamma(\beta_i)}\prod_{i=1}^{K}p_i^{(\alpha_i+\beta_i-1)-1} \end{aligned}$$

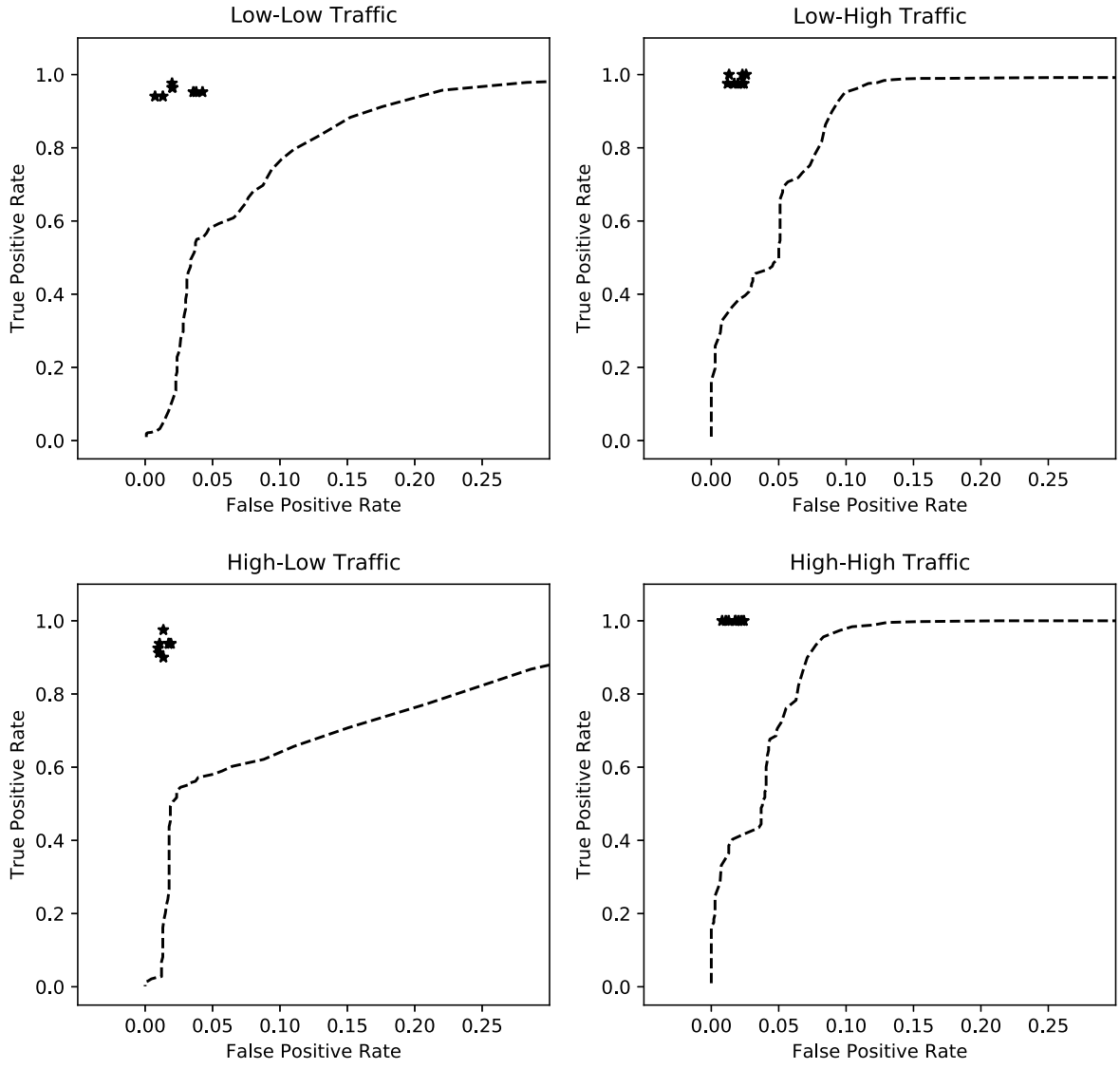

Fig. 5. Performance of Hellinger distance based anomaly detection on our data sets.

$$
\begin{aligned}
&= c_1 c_2 \frac{\Gamma\left(\sum_{i=1}^{K} \alpha_i\right)}{\prod_{i=1}^{K} \Gamma(\alpha_i)} \frac{\Gamma\left(\sum_{i=1}^{K} \beta_i\right)}{\prod_{i=1}^{K} \Gamma(\beta_i)} \prod_{i=1}^{K} p_i^{(\alpha_i+\beta_i-1)-1} \\
&\quad \times \frac{\Gamma\left(\sum_{i=1}^{K} (\alpha_i+\beta_i-1)\right)}{\prod_{i=1}^{K} \Gamma((\alpha_i+\beta_i-1))} \\
&\quad \times \frac{\prod_{i=1}^{K} \Gamma((\alpha_i+\beta_i-1))}{\Gamma\left(\sum_{i=1}^{K} (\alpha_i+\beta_i-1)\right)} \\
&= \underbrace{c_1 c_2 \frac{\Gamma\left(\sum_{i=1}^{K} \alpha_i\right)}{\prod_{i=1}^{K} \Gamma(\alpha_i)} \frac{\Gamma\left(\sum_{i=1}^{K} \beta_i\right)}{\prod_{i=1}^{K} \Gamma(\beta_i)} \frac{\prod_{i=1}^{K} \Gamma(\alpha_i+\beta_i-1)}{\Gamma\left(\sum_{i=1}^{K} (\alpha_i+\beta_i-1)\right)}}_{\text{normalizing constant}} \\
&\quad \times \mathcal{D}ir(p; \alpha+\beta-1) \qquad \text{(A.1)}
\end{aligned}
$$

$$
\begin{aligned}
&c_1 \mathcal{G}(\lambda; \alpha_1, \beta_1) \times c_2 \mathcal{G}(\lambda; \alpha_2, \beta_2) \\
&= c_1 c_2 \frac{\beta_1^{\alpha_1}}{\Gamma(\alpha_1)} \lambda^{\alpha_1-1} \exp^{-\beta_1 \lambda} \frac{\beta_2^{\alpha_2}}{\Gamma(\alpha_2)} \lambda^{\alpha_2-1} \exp^{-\beta_2 \lambda} \\
&= c_1 c_2 \frac{\beta_1^{\alpha_1}}{\Gamma(\alpha_1)} \frac{\beta_2^{\alpha_2}}{\Gamma(\alpha_2)} \lambda^{(\alpha_1+\alpha_2-1)-1} \exp^{-(\beta_1+\beta_2)\lambda} \\
&= c_1 c_2 \frac{\beta_1^{\alpha_1}}{\Gamma(\alpha_1)} \frac{\beta_2^{\alpha_2}}{\Gamma(\alpha_2)} \lambda^{\tilde{\alpha}-1} \exp^{-\tilde{\beta}\lambda} \\
&= c_1 c_2 \frac{\beta_1^{\alpha_1}}{\Gamma(\alpha_1)} \frac{\beta_2^{\alpha_2}}{\Gamma(\alpha_2)} \frac{\Gamma(\tilde{\alpha})}{\tilde{\beta}^{\tilde{\alpha}}} \frac{\tilde{\beta}^{\tilde{\alpha}}}{\Gamma(\tilde{\alpha})} \lambda^{\tilde{\alpha}-1} \exp^{-\tilde{\beta}\lambda} \\
&= \underbrace{c_1 c_2 \frac{\beta_1^{\alpha_1}}{\Gamma(\alpha_1)} \frac{\beta_2^{\alpha_2}}{\Gamma(\alpha_2)} \frac{\Gamma(\tilde{\alpha})}{\tilde{\beta}^{\tilde{\alpha}}}}_{\text{normalizing constant}} \mathcal{G}(\lambda; \alpha_1+\alpha_2-1, \beta_1+\beta_2) \qquad \text{(A.2)}
\end{aligned}
$$

What is more, update step of the forward–backward algorithm requires computing the product of observation model Θ and reset model Ω, which can be easily performed thanks to the conjugacy property. Alternatively, one may reformulate Θ distribution in terms of a normalization constant and Ω distribution and take advantage of that the multiplication of two Ω potentials is already defined.

$$
\begin{aligned}
\mathcal{M}(x; p) &= \frac{\Gamma(\sum_i x_i + 1)}{\prod_i \Gamma(x_i+1)} \prod_i p_i^{x_i} \\
&= \frac{\Gamma(\sum_i x_i + 1)}{\prod_i \Gamma(x_i+1)} \frac{\Gamma(\sum_i (x_i+1))}{\Gamma(\sum_i (x_i+1))} \prod_i p_i^{x_i}
\end{aligned}
$$

$$= \frac{\Gamma(\sum_i x_i + 1)}{\Gamma(\sum_i (x_i + 1))} \mathcal{D}ir(p; x + 1) \tag{A.3}$$

$$\mathcal{P}(x; \lambda) = \frac{\lambda^x e^{-\lambda}}{\Gamma(x+1)} = \mathcal{G}(\lambda; x + 1, 1) \tag{A.4}$$

Appendix B. Parameter estimation for the multiple change point model

Here we present the detailed equations of the Expectation-Maximization method for estimating model parameters.

B.1. E-step

At the E-step, we calculate the expectation of the complete data log-likelihood as follows

$$\mathcal{L}_{\pi,w} = \log p(s_{1:T}, h_{0:T}, v_{1:T}, |\pi, w) \tag{B.1}$$

$$= \sum_{t=1}^{T} \Big[[s_t = 0] \big(\log(1-\pi) + \log \delta(h_t - h_{t-1}) \big) + [s_t = 1] \big(\log \pi + \log \Omega(h_t; w) \big) + \log \Theta(v_t; h_t) \Big] + \log \Omega(h_0; w) \tag{B.2}$$

We can write the expectation of this complete likelihood under the auxiliary distribution $q(z) = p(s_{1:T}, h_{0:T} | v_{1:T}, \pi, w)$ as follows, by letting $p(s_0 = 1 | v_{1:T}, \pi, w) = 1$ in a compact way as

$$\langle \mathcal{L}_{\pi,w} \rangle_{q(z)} = \sum_{t=0}^{T} \Big[\langle s_t = 1 \rangle_{q(z)} \Big(\log \pi + \langle \log \Omega(h_t; w) \rangle_{q(z)} \Big) + \langle s_t = 0 \rangle_{q(z)} \log(1-\pi) + \langle \log \Theta(v_t; h_t) \rangle_{q(z)} \Big] \tag{B.3}$$

B.2. M-step

We maximize the expectation of the complete data log-likelihood with respect to the parameters π and w.

B.2.1. Maximizing for π

Maximizing the expectation of the complete data log likelihood for π is the same for all prior and observation models.

$$0 = \frac{\partial \langle \mathcal{L}_{\pi,w} \rangle_{q(z)}}{\partial \pi} \tag{B.4}$$

$$0 = \frac{\partial}{\partial \pi} \sum_{t=1}^{T} \Big(\langle s_t = 1 \rangle_{q(z)} \log \pi + \langle s_t = 0 \rangle_{q(z)} \log(1-\pi) \Big) \tag{B.5}$$

$$\pi = \frac{1}{T} \sum_{t=1}^{T} \langle s_t = 1 \rangle_{q(z)} \tag{B.6}$$

B.2.2. Maximizing for w

Maximizing the expectation of the complete data log likelihood for w depends on the observation and prior distributions. We begin by taking the derivative

$$0 = \frac{\partial \langle \mathcal{L}_{\pi,w} \rangle_{q(z)}}{\partial w} \tag{B.7}$$

$$= \frac{\partial}{\partial w} \sum_{t=1}^{T} \langle s_t = 1 \rangle_{q(z)} \langle \log \Omega(h_t; w) \rangle_{q(z)} \tag{B.8}$$

In this work, we used a coupled model where the signals are assumed to be generated by a Poisson-Gamma or Multinomial-Dirichlet models, hence in our case $w = [\alpha_{1:M}, a_{1:N}, b_{1:N}]$, where $\alpha_{1:M}$ is the parameter of the Dirichlet prior and each $\{a_i, b_i\}$ pair is the parameter for a Gamma prior. For Dirichlet priors,

$$0 = \frac{\partial}{\partial \alpha_k} \sum_{t=0}^{T} \langle s_t = 1 \rangle_{q(z)} \langle \log Dir(h_t; \alpha) \rangle_{q(z)} \tag{B.9}$$

$$= \frac{\partial}{\partial \alpha_k} \sum_{t=0}^{T} \langle s_t = 1 \rangle_{q(z)} \Bigg[\sum_k (\alpha_k - 1) \langle \log h_{t,k} \rangle_{q(z)} + \log \Gamma(\sum_k \alpha_k) - \sum_k \log \Gamma(\alpha_k) \Bigg] \tag{B.10}$$

$$= \sum_{t=0}^{T} \langle s_t = 1 \rangle_{q(z)} \Big(\langle \log h_{t,k} \rangle_{q(z)} + \psi(\sum_k \alpha_k) - \psi(\alpha_k) \Big) \tag{B.11}$$

We can solve the above equation for α_k with fixed-point iterations [42]:

$$\alpha_k^{new} = \psi^{-1} \Big(\frac{1}{C} \sum_{t=0}^{T} \langle s_t = 1 \rangle_{q(z)} \langle \log h_{t,k} \rangle_{q(z)} + \psi(\sum_k \alpha_k^{old}) \Big) \tag{B.12}$$

where $C = \sum_t \langle s_t = 1 \rangle_{q(z)}$. We calculate the $\langle \log h_{t,k} \rangle_{q(z)}$ from the mixture of Dirichlet potentials as

$$\langle \log h_{t,k} \rangle_{q(z)} = \frac{1}{\mathbf{Z}} \sum_{r=1}^{R} c^r \left(\psi(\alpha_k^r) - \psi(\alpha_0^r) \right) \tag{B.13}$$

where R is the number of potentials and $Z = \sum_r c^r$ is the sum of their normalizing constants.

For the Gamma distribution, the maximization leads to the following fixed-point iteration for a and equation for b [43]:

$$a^{new} = \psi^{-1} \Bigg(\frac{1}{C} \sum_{t=0}^{T} \langle s_t = 1 \rangle_{q(z)} \Big(\langle \log h_{t,k} \rangle_{q(z)} - \log \langle h_{t,k} \rangle_{q(z)} \Big) + \log a^{old} \Bigg) \tag{B.14}$$

$$b = \frac{1 a^{new}}{\sum_{t=0}^{T} \langle s_t = 1 \rangle_{q(z)} \langle h_t \rangle_{q(z)}} \tag{B.15}$$

The expected sufficient statistics for the Gamma parameters are calculated from mixtures of Gamma potentials by:

$$\langle h_t \rangle_{q(z)} = \frac{1}{\mathbf{Z}} \sum_{r=1}^{R} c^r (a^r b^r)$$

$$\langle \log h_t \rangle_{q(z)} = \frac{1}{\mathbf{Z}} \sum_{r=1}^{R} c^r \left(\psi(a^r) - \log(b^r) \right) \tag{B.16}$$

again, R is the number of potentials and $Z = \sum_r c^r$ is the sum of their normalizing constants.

Appendix C. SIP traffic generator model

Here we describe the generative process underlying the BounSim network simulator.

Please cite this article in press as: B. Kurt et al., A Bayesian change point model for detecting SIP-based DDoS attacks, Digit. Signal Process. (2017), https://doi.org/10.1016/j.dsp.2017.10.009

C.1. Social network of SIP users

In order to simulate a realistic network, we modeled the relationships between users by a stochastic block model [44]. In a stochastic graph model, graph nodes are distributed over groups and the probability of having a connection between two nodes is governed by inner and between group connection parameters. Similarly we distributed N SIP users over K social groups. Whenever a user decides to make a call, they pick a callee within their group or from other groups with different probabilities. As a concrete example, we can think of SIP users inside a company, divided into different departments. A user may talk to their teammates more often then people from other teams.

At the beginning of the simulation, the generative model first generates the social network. The probability that a user belongs to a certain group, π is drawn from a Dirichlet distribution with a parameter α. Relative group sizes can be adjusted by the α parameter. We represent the group assignments of users by an $N \times K$ dimensional binary G matrix, where each row g_n represents a user such that $g_{n,k} = 1$ if and only if user n belongs to group k.

$$\pi \sim \mathcal{D}ir(\pi; \alpha) \tag{C.1}$$

$$p(\pi|\alpha) = \frac{\Gamma(\sum_k \alpha_k)}{\prod_k \Gamma(\alpha_k)} \prod_k \pi_k^{\alpha_k - 1} \tag{C.2}$$

Then each user is assigned to a group from a categorical distribution.

$$g_n \sim \mathcal{C}at(g_n; \pi) \quad \text{for each } n \in [1, N] \tag{C.3}$$

$$p(g_{n,k}) = \prod_k \pi_k^{g_{n,k}} \tag{C.4}$$

Typically, every stochastic block model has a $K \times K$ parameter matrix B, such that $B_{i,j}$ is the probability of having an edge between a node from group i to some other node in group j. We created the B matrix such that inner group communications are more probable than between group communications by setting the Beta distribution parameters as $a > b$:

$$B_{i,i} \sim \mathcal{B}eta(B_{i,i}; a, b) \quad \forall i \in [1, K] \tag{C.5}$$

$$B_{i,j} \sim \mathcal{B}eta(B_{i,j}; b, a) \quad \forall i \neq j \tag{C.6}$$

C.2. Phone book of SIP users

After the social network is constructed, the simulator creates a phone book for each user, according to the users social behavior. Let P be an $N \times N$ phone book matrix, such that $P_{m,n}$ denotes the probability that user m calls user n whenever m decides to call someone.

$$P_{m,n} \propto \prod_{i,j} B_{i,j}^{G_{m,i} G_{n,j}} \tag{C.7}$$

or, in compact matrix notation,

$$P \propto GBG^T \tag{C.8}$$

C.3. Registration of users

The simulation starts with all users offline. A user becomes online by sending a *REGISTER* request to the SIP server. Each user waits a random amount of time before registering to the server. Let r_n denote the amount of time user n waits offline. We generate this waiting time from a Gamma distribution:

$$r_n \sim \mathcal{G}(\beta_i; \rho, \phi) \tag{C.9}$$

$$p(r_n|\rho, \phi) = \beta_i^{\rho - 1} \frac{e^{-x/\phi}}{\phi^\rho \Gamma(\rho)} \tag{C.10}$$

C.3.1. Call rates

A users call rate is governed by the time they wait idle after communications. Whenever user n becomes idle, that is after registration or end of a call, a random waiting time t_n is sampled from an exponential distribution, and makes a random call at the end of this time period. However, they can choose to answer an incoming call during this period.

$$t_n|\beta_n \sim \mathcal{E}xp(t_n; \beta_n) \tag{C.11}$$

$$p(t_n|\beta_n) = 1/\beta_n \exp(-t_n/\beta_n) \tag{C.12}$$

Here, β_n is the call rate parameter of user N, is sampled for every user from a Gamma distribution

$$\beta_i \sim \mathcal{G}(\beta_i; k, \theta) \tag{C.13}$$

Hence, in addition to their social behavior, each user has a different personal behavior.

C.3.2. Making a call

At the end of their waiting time, a user initiates a call by randomly selecting another user from their phone book. Let c_n denote the callee that user n is about to call. We draw c_n from a categorical distribution

$$c_n|P_n \propto \mathcal{K}(c_n; P_n) \tag{C.14}$$

such that $P_n \propto \{P_{n,1}, \ldots P_{n,N}\}$ is the normalized probability vector for user n, where $P_{n,m}$ is the probability that user n calls m. When the user n selects the callee, they send *INVITE* request to start a conversation.

C.3.3. Responding to a call

Whenever a user receives a call, they can accept or reject the call, and as a third option, may not notice the call at all. Each user has three call response parameters, f_n, a_n and h_n, such that f_n is the call notice parameter, a_n call accepting parameter and h_n is the call hold parameter.

$$f_n \sim \mathcal{U}(f_n; f_{min}, f_{max}) \tag{C.15}$$

$$a_n \sim \mathcal{U}(a_n; a_{min}, a_{max}) \tag{C.16}$$

$$h_n \sim \mathcal{U}(h_n; h_{min}, h_{max}) \tag{C.17}$$

Whenever a user receives a call, they notice the call with probability f_n and, if they do notice, accepts the call with probability a_n. If the user is already on another call, they can put their ongoing call on hold with probability h_n and accepts the call.

C.3.4. Call durations

If a call is successfully established, both participants draw their own call duration, and at the end of this duration, terminate the call.

$$d_n|\delta_n \sim \mathcal{E}xp(d_n; \delta_n) \tag{C.18}$$

$$d_m|\delta_m \sim \mathcal{E}xp(d_m; \delta_m) \tag{C.19}$$

$$d_c = \min(d_n, d_m) \tag{C.20}$$

Here, d_n and d_m denotes the call durations sampled by call participants n and m and d_c is the actual call duration.

Please cite this article in press as: B. Kurt et al., A Bayesian change point model for detecting SIP-based DDoS attacks, Digit. Signal Process. (2017), https://doi.org/10.1016/j.dsp.2017.10.009

Appendix D. Pseudo code of forward–backward helper routines

Algorithm 2 Backward Filtering Loop.

```
function BackwardFilter(π, w, v_{1:T})
    beta ← [ ]
    prior ← [w, 0]
    for t = T ... 1 do
        obs_pot ← Obs2Pot(v_t)
        // Change case
        tmp_msg ← [prior*pot for pot in beta[end]]
        new_msg ← [ ][obs_pot.w, log π + LogLikelihood(tmp_msg)] ]
        // No change case
        no_change ← [obs_pot*pot for pot in beta[end]]
        for pot in no_change: pot[2]+=log(1 − π)
        // Update beta
        for pot in no_change: new_msg.append(pot)
        beta.append(new_msg)
    end for
    return beta
end function
```

Algorithm 3 Forward–Backward Recursion Functions.

```
function Predict(alpha, π, w)
    alpha_p ← [ ]
    alpha_p.append(w, LogLikelihood(alpha) + log(π))
    for pot in alpha: alpha_p.append(pot[1], pot[2]+log(1 − π))
    return alpha_p
end function

function Update(msg, v_t)
    msg_u ← [ ]
    obs_pot =← Obs2Pot(v_t)                ▷ see eq's. (A.3) and (A.4)
    msg_u = [pot*obs_pot for pot in msg]   ▷ see eq's. (A.1) and (A.2)
    return msg_u
end function

function ComputeCPP(msg, d)
    // Change case is represented by the mixture of first d potentials
    p1 = exp(LogLikelihood(msg[1 : d])
    p0 = exp(LogLikelihood(msg[d + 1 :])
    return p1 /(p0 + p1)
end function

function Smooth(alpha, beta)
    gamma ← [ ]
    for i = 1 ... len(alpha) do
        for j = 1 ... len(beta) do
            gamma.append(alpha[i] * beta[j])
        end for
    end for
    return gamma
end function
```

References

[1] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, E. Schooler, RFC 3261: SIP: Session Initiation Protocol, Technical Report, IETF, 2007.
[2] D. Geneiatakis, T. Dagiuklas, G. Kambourakis, C. Lambrinoudakis, S. Gritzalis, K.S. Ehlert, D. Sisalem, Survey of security vulnerabilities in session initiation protocol, IEEE Commun. Surv. Tutor. 8 (2006) 68–81.
[3] A.D. Keromytis, A comprehensive survey of voice over IP security research, IEEE Commun. Surv. Tutor. 14 (2012) 514–537.
[4] J. Mirkovic, P. Reiher, A taxonomy of DDoS attack and DDoS defense mechanisms, Comput. Commun. Rev. 34 (2004) 39–53.
[5] W. Eddy, RFC 4987: z, Technical Report, IETF, 2007.
[6] J. Sanders, Chinese Government Linked to Largest DDoS Attack in GitHub History, Technical Report, 2015.
[7] M. Korolov, DDoS Attack on BBC May Have Been Biggest in History, Technical Report, 2016.
[8] DDoS Trends Report, Technical Report, Corero, 2017.
[9] Verisign DDoS Trends Report, Technical Report, Verisign, 2017.
[10] M. Cooney, IBM Warns of Rising VoIP Cyber-Attacks, Technical Report, 2016.
[11] R.K. Chang, Defending against flooding-based distributed denial-of-service attacks: a tutorial, IEEE Commun. Mag. 40 (2002) 42–51.
[12] T. Peng, C. Leckie, K. Ramamohanarao, Survey of network-based defense mechanisms countering the DoS and DDoS problems, ACM Comput. Surv. 39 (2007).
[13] S. Ehlert, D. Geneiatakis, T. Magedanz, Survey of network security systems to counter SIP-based denial-of-service attacks, Comput. Secur. 29 (2010) 225–243.
[14] B. Iancu, Ser pike excessive traffic monitoring module, 2003.
[15] Y.-S. Wu, S. Bagchi, S. Garg, N. Singh, T.K. Tsai, SCIDIVE: A Stateful and Cross Protocol Intrusion Detection Architecture for Voice-Over-IP Environments, IEEE Computer Society, 2004, pp. 433–442.
[16] E.Y. Chen, Detecting DoS attacks on SIP systems, in: 1st IEEE Workshop on VoIP Management and Security, IEEE, 2006, pp. 53–58.
[17] D. Sisalem, J. Kuthan, S. Ehlert, Denial of service attacks targeting a SIP VoIP infrastructure: attack scenarios and prevention mechanisms, IEEE Netw. 20 (2006), Special Issue on Securing VOIP.
[18] V. Chandola, A. Banerjee, V. Kumar, Anomaly detection: a survey, ACM Comput. Surv. 41 (2009) 15.
[19] P. Fearnhead, Exact and efficient Bayesian inference for multiple changepoint problems, Stat. Comput. 16 (2006) 203–213.
[20] V-Spy, Nova V-SPY, http://www.netas.com.tr/en/innovation-productization/nova-cyber-security-products/. (Accessed 29 April 2016). Online.
[21] H. Sengar, D. Wijesekera, H. Wang, S. Jajodia, VoIP intrusion detection through interacting protocol state machines, in: International Conference on Dependable Systems and Networks, DSN'06, IEEE, pp. 393–402.
[22] Y. Bouzida, C. Mangin, A framework for detecting anomalies in VoIP networks, in: Third International Conference on Availability, Reliability and Security, ARES 08, IEEE, 2008, pp. 204–211.
[23] S. Ehlert, G. Zhang, D. Geneiatakis, G. Kambourakis, T. Dagiuklas, J. Markl, D. Sisalem, Two layer denial of service prevention on SIP VoIP infrastructures, Comput. Commun. 31 (2008) 2443–2456.
[24] G. Ormazabal, S. Nagpal, E. Yardeni, H. Schulzrinne, Secure SIP: a scalable prevention mechanism for DoS attacks on SIP based VoIP systems, Springer, Berlin, Heidelberg, pp. 107–132.
[25] M. Nassar, R. State, O. Festor, Monitoring SIP traffic using support vector machines, in: Proceedings of the 11th International Symposium on Recent Advances in Intrusion Detection, RAID '08, Springer-Verlag, Berlin, Heidelberg, 2008, pp. 311–330.
[26] M.A. Akbar, M. Farooq, Application of evolutionary algorithms in detection of SIP based flooding attacks, in: Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation, GECCO '09, ACM, New York, NY, USA, 2009, pp. 1419–1426.
[27] M.A. Akbar, M. Farooq, Securing SIP-based VoIP infrastructure against flooding attacks and spam over IP telephony, Knowl. Inf. Syst. 38 (2014) 491–510.
[28] Z. Tsiatsikas, A. Fakis, D. Papamartzivanos, D. Geneiatakis, G. Kambourakis, C. Kolias, Battling against DDoS in SIP – is machine learning-based detection an effective weapon?, in: Proceedings of the 12th International Conference on Security and Cryptography, vol. 1: SECRYPT, ICETE 2015, INSTICC, ScitePress, 2015, pp. 301–308.
[29] Z. Tsiatsikas, D. Geneiatakis, G. Kambourakis, S. Gritzalis, Realtime DDoS detection in SIP ecosystems: machine learning tools of the trade, Springer International Publishing, pp. 126–139.
[30] B. Reynolds, D. Ghosal, Secure IP telephony using multi-layered protection, in: NDSS.
[31] Y. Rebahi, D. Sisalem, Change-point detection for voice over IP denial of service attacks, in: KiVS 2007, 2007.
[32] H. Zhang, Z. Gu, C. Liu, T. Jie, Detecting VoIP-specific denial-of-servijordan rudessce using change-point method, in: Proceedings of the 11th International Conference on Advanced Communication Technology, vol. 2, IEEE Press, 2009, pp. 1059–1064.
[33] D. Geneiatakis, N. Vrakas, C. Lambrinoudakis, Utilizing bloom filters for detecting flooding attacks against SIP based services, Comput. Secur. 28 (2009) 578–591.
[34] H. Sengar, H. Wang, D. Wijesekera, S. Jajodia, Fast detection of denial-of-service attacks on IP telephony, in: IWQoS.
[35] Z. Tsiatsikas, D. Geneiatakis, G. Kambourakis, Exposing resource consumption attacks in internet multimedia services, in: Proceedings of 14th IEEE International Symposium on Signal Processing and Information Technology (ISSPIT), Security Track, IEEE Press, 2014, pp. 1–6.
[36] M. Handley, V. Jacobson, RFC 4566: SDP: Session Description Protocol, Technical Report, IETF, 2006.
[37] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson, RFC 3550: RTP: A Transport Protocol for Real-Time Applications, Technical Report, IETF, 2003.
[38] D. Barber, Bayesian Reasoning and Machine Learning, Cambridge University Press, New York, NY, USA, 2012.
[39] Ç. Yıldız, B. Kurt, T.Y. Ceritli, B. Sankur, A.T. Cemgil, A real-time SIP network simulation and monitoring system, SoftwareX (2017), in press, Special Issue on Reproducible Research.
[40] Trixbox, http://www.fonality.com/trixbox. (Accessed 29 April 2016). Online.
[41] M. Voznak, Evaluating the performance of sip infrastructure, Geant-Terena (2011).

Please cite this article in press as: B. Kurt et al., A Bayesian change point model for detecting SIP-based DDoS attacks, Digit. Signal Process. (2017), https://doi.org/10.1016/j.dsp.2017.10.009

[42] T. Minka, Estimating a Dirichlet Distribution, Technical Report, 2016.
[43] T. Minka, Estimating a Gamma Distribution, Technical Report, 2002.
[44] A. Goldenberg, A.X. Zheng, S.E. Fienberg, E.M. Airoldi, A survey of statistical network models, Found. Trends Mach. Learn. 2 (2010) 129–233.

Barış Kurt is a Ph.D. student at the Computer Engineering Department in Bogazici University. He received his B.S. and M.Sc. degrees from the same department. His research interests are Bayesian statistics, machine learning, approximate inference, Monte Carlo methods and network traffic analysis.

Cagatay Yıldız received the BSc and MSc degrees in computer engineering from Bogazici University, Istanbul in 2014 and 2017, respectively. He is currently a PhD candidate at the Department of Computer Science at Aalto University, Finland. His research interests mainly involve Bayesian statistical methods and inference, machine learning and stochastic optimization.

Taha Yusuf Ceritli received his B.Sc. (2015) from Electrical and Electronics Department at Bogazici University. He holds his M.Sc. (2017) from Department of Computational Science and Engineering at the same university. Currently, he is a Ph.D. student at The Alan Turing Institute/University of Edinburgh. His research interests include Bayesian inference and machine learning.

Bülent Sankur is presently at Bogazici University in the Department of Electrical-Electronic Engineering. His research interests are in the areas of digital signal processing, security and biometry, cognition and multimedia systems. He has served as a consultant in several industrial and government projects and has been involved in various European framework and/or bilateral projects. He has held visiting positions at the University of Ottawa, Technical University of Delft, and Ecole Nationale Supérieure des Télécommunications, Paris. He has published over 210 journal and conference papers and co-authored two books. He was the chairman of EUSIPCO'05: The European Conference on Signal Processing, as well as technical chairman of ICASSP'00. Dr. Sankur is presently an associate editor of Journal of Image and Video Processing, Image and Vision Computing, and Annals of Telecommunications.

Taylan Cemgil received Ph.D. (2004) from SNN, Radboud University Nijmegen, the Netherlands. Between 2004 and 2008 he worked as a postdoctoral researcher at Amsterdam University and University of Cambridge, UK. He is currently an associate professor of Computer Engineering at Bogazici University, Istanbul, Turkey. His research interests are in Bayesian statistical methods and inference, machine learning and signal processing.

Please cite this article in press as: B. Kurt et al., A Bayesian change point model for detecting SIP-based DDoS attacks, Digit. Signal Process. (2017), https://doi.org/10.1016/j.dsp.2017.10.009