A Data Plane Approach for Detecting Control Plane Anomalies in Mobile Networks

Omer H. Abdelrahman(✉) and Erol Gelenbe

Department of Electrical and Electronic Engineering,
Imperial College, London SW7 2BT, UK
{o.abd06,e.gelenbe}@imperial.ac.uk

Abstract. This paper proposes an anomaly detection framework that utilizes key performance indicators (KPIs) and traffic measurements to identify in real-time misbehaving mobile devices that contribute to signaling overloads in cellular networks. The detection algorithm selects the devices to monitor and adjusts its own parameters based on KPIs, then computes various features from Internet traffic that capture both sudden and long term changes in behavior, and finally combines the information gathered from the individual features using a random neural network in order to detect anomalous users. The approach is validated using data generated by a detailed mobile network simulator.

Keywords: Mobile security · Random neural network · M2M · IoT · Signaling overload · Radio resource control · Key performance indicators

1 Introduction

The number of smart mobile devices which require constant access to the Internet has grown exponentially in recent years, placing significant pressure on the data and signaling infrastructures of service providers. While mobile network operators are able to cope with the growth in user traffic by increasing capacity, overloads in the signaling plane are often unpredictable and lead to performance degradation and even outages. Thus, there has been considerable interest from standardization bodies, operators and equipment vendors in addressing mobile *signaling storms*, particularly with the advent of machine to machine (M2M) and Internet of Things (IoT) whose traffic profiles can be resource-inefficient. Initial attempts have focused on developing new standards for M2M communications [1], and promoting best practices for developing network-friendly applications or optimizing network configurations [5,8,16,18].

This paper presents a random neural network (RNN) [11,12] based approach for detecting mobile devices that generate excessive radio resource control (RRC) signaling, without directly monitoring the control plane itself. In contrast to signaling based techniques [7,13], which can be more effective but require modification to cellular network equipment and/or protocols, the present approach captures packets at the edge of the mobile core network using standard switch

B. Mandler et al. (Eds.) IoT 360° 2015, Part I, LNICST 169, pp. 210–221, 2016.
DOI: 10.1007/978-3-319-47063-4_19

technologies (e.g. port mirroring, fibre taps, etc.). This offers the advantages of not requiring to decode lower radio related layers, lack of network encryption, and fewer number of nodes to monitor [23]. Moreover, the detector relies mainly on timestamps and packet header information to classify users, and does not require knowledge of the application generating a packet nor its service type, eliminating the need to use a commercial deep packet inspection tool which results in considerable overhead for real-time detection. It also interacts with existing network management and monitoring systems to reduce computational overhead, storage requirements and false alarm rate. We use supervised learning to distinguish between normal and anomalous signaling behaviors, which is well suited for classifying known patterns such as signaling storms whose characteristics and root causes are well understood [3,4,9,15].

The rest of the paper is structured as follows. Sections 1.1 and 1.2 discuss related work and the RNN as applied to our problem. Section 2 presents the detection system, along with a description of its input features and the parameters that can influence its performance. In Sect. 3, we evaluate our detection mechanism using simulations. Finally, we summarize our findings in Sect. 4.

1.1 Related Work

Mobile networks are subject to RRC-based signaling storms, which occur when a large number of mobiles make successive connection requests that time-out because of inactivity, overloading the control plane of the network [3,15] and draining the mobile devices' batteries [9]. This type of misbehavior can be triggered by poorly designed applications and M2M systems, outages in mobile cloud services or malicious activities, and it is difficult to recognize using traditional DDoS detection systems because of the low traffic volume nature of the attack.

Online detection of deliberate signaling attacks was first considered in [19], where connection inter-setup times for each mobile are estimated from IP metrics in order to detect the intention of a remote host to launch an attack. A general framework for anomaly detection was proposed in [6] based on time-series analysis of one dimensional feature distributions. While [19] and [6] aim to identify large scale events by aggregating and analyzing statistics from all hosts and mobile users, respectively, we aim to identify in real-time users that are contributing to a problem (i.e. signaling overload) rather than detect the problem itself. A supervised learning approach was used in [17] to detect mobile-initiated signaling attacks, by monitoring transmissions that trigger a radio access bearer setup procedure, and extracting various features from the corresponding packets relating to destination IP and port numbers, packet size, and response-request ratio. Although we utilize similar attributes in our approach, we do not assume knowledge of the effect that a packet has on the control plane (i.e. whether it has triggered a connection setup procedure), thus simplifying the deployment of our solution in operational networks.

A number of commercial solutions also started to appear in response to recent incidents of signaling storms, and can be classified into three groups:

(i) Anomaly detection and mitigation tools [7] similar to the approach we suggested in [13]. (ii) Air interface optimization which aims to increase the number of simultaneously connected devices in the access network; such technologies are constantly evolving with new standards, specifications and proprietary admission/congestion control and scheduling algorithms added all the time, and our solution operates on top of and is complimentary to them. (iii) Dedicated signaling infrastructure solutions to handle the expected growth in core network signaling pertaining to policies, charging, mobility management and other new services offered for the first time in LTE networks; however, it is expected that congestion management and load balancing in the core network will be less of an issue, with the trend towards network functions virtualization that will enable dynamic resource scaling as required by network load.

1.2 The Random Neural Network

The RNN is a biologically inspired computational model, introduced by Gelenbe [11], in which neurons exchange signals in the form of spikes of unit amplitude. In RNN, positive and negative signals represent excitation and inhibition respectively, and are accumulated in neurons. Positive signals are canceled by negative signals, and neurons may fire if their potential is positive. A signal may leave neuron i for neuron j as a positive signal with probability p^+_{ij}, as a negative signal with probability p^-_{ij}, or may depart from the network with probability d_i, where $\sum_j [p^+_{ij} + p^-_{ij}] + d_i = 1$. Thus, when neuron i is excited, it fires excitatory and inhibitory signals to neuron j with rates:

$$w^+_{ij} = r_i p^+_{ij} \geq 0, \quad w^-_{ij} = r_i p^-_{ij} \geq 0,$$

where $r_i = (1 - d_i)^{-1} \sum_j [w^+_{ij} + w^-_{ij}]$. The steady-state probability that neuron i is excited is given by:

$$q_i = \frac{\Lambda_i + \sum_j q_j w^+_{ji}}{\lambda_i + r_i + \sum_j q_j w^-_{ji}},$$

where Λ_i and λ_i denote the rates of exogenous excitatory and inhibitory signal inputs into neuron i, respectively.

A gradient descent supervised learning algorithm for the recurrent RNN has been developed in [12]. For a RNN with n neurons, the learning algorithm estimates the $n \times n$ weight matrices $\mathbf{W}^+ = \{w^+_{ij}\}$ and $\mathbf{W}^= \{w^-_{ij}\}$ from a training set comprising K input-output pairs $(\mathbf{X}, \mathbf{Y})$. The set of successive inputs to the algorithm is $\mathbf{X} = (\mathbf{x}^{(1)}, \cdots, \mathbf{x}^{(K)})$, where $\mathbf{x}^{(k)} = (\mathbf{\Lambda}^{(k)}, \boldsymbol{\lambda}^{(k)})$ are the pairs of exogenous excitatory and inhibitory signals entering each neuron from outside the network:

$$\mathbf{\Lambda}^{(k)} = (\Lambda^{(k)}_1, \cdots, \Lambda^{(k)}_n), \quad \boldsymbol{\lambda}^{(k)} = (\lambda^{(k)}_1, \cdots, \lambda^{(k)}_n).$$

The successive desired outputs are $\mathbf{Y} = (\mathbf{y}^{(1)}, \cdots, \mathbf{y}^{(K)})$, with the k-th vector $\mathbf{y}^{(k)} = (y^{(k)}_1, \cdots, y^{(k)}_n)$ whose elements $y^{(k)}_i \in [0, 1]$ correspond to the desired output values for each neuron. The training examples are presented to the network

sequentially, and the weights are updated according to the gradient descent rule to minimize an error function:

$$E^{(k)} = \frac{1}{2}\sum_{i=1}^{n} a_i[q_i^{(k)} - y_i^{(k)}]^2, a_i \geq 0.$$

The update procedure requires a matrix inversion operation for each neuron pairs (i, j) and input k which can be done in time complexity $O(n^3)$, or $O(mn^2)$ if m-step relaxation method is used, and $O(n^2)$ for feed-forward networks. We use the RNN because it has been successfully applied to several engineering problems [24] including pattern recognition, classification and DoS attack detection [14,20], but our detection system can work with other machine learning algorithms.

2 The Detection System

Figure 1 shows the basic architecture of the packet-switched domain of a mobile network, along with the detection system which intercepts packets directed to/from the gateway (i.e. Gn/S5 interface in UMTS/LTE). The user data transported over this interface are encapsulated in GTP-U (a simple IP based tunneling protocol) packets. The detector utilizes also information from the operation support system to reduce search space and optimize performance, and produces in real-time a list of anomalous mobiles for root cause analysis and mitigation. The three data processing stages of the algorithm are: (i) user filtering and parameter selection based on network configuration settings and KPIs related to signaling load on various network components, (ii) feature generation, and (iii) user classification with a trained RNN model. For reasons that should become apparent, we describe these processing steps in a logical order rather than the order in which they happen during run time.

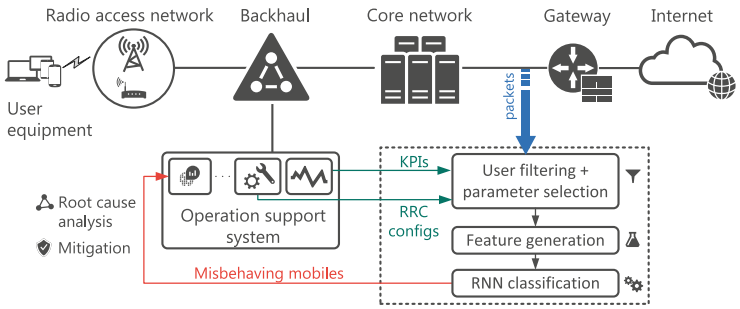

Fig. 1. The detection system and its interactions with the elements of a mobile network.

2.1 Online RNN Classification

The RNN-based algorithm monitors the activity of a set of mobile devices, specified by the data filter, and calculates expressive features that describe various characteristics of the users' behavior. Time is divided into slots, each of duration Δ seconds, in which summary statistics of several quantities related to IP traffic of each user are collected. The algorithm stores the most recent w set of measurements, and uses them to compute the current values of the input features, i.e. the features for time slot τ are computed from measurements obtained for time slots $\tau, \tau-1, \cdots, \tau-(w-1)$ so that the observation window of the algorithm is $W = w\Delta$. Let $z^{(\tau)}$ denote a measured or calculated quantity for time slot τ, then the i-th input feature $x_i^{(\tau)}$ is obtained by applying a statistical function ϕ_i of the following form:

$$x_i^{(\tau)} = \phi_i\big(z^{(\tau)}, z^{(\tau-1)}, \cdots, z^{(\tau-w-1)}\big).$$

Hence, by employing different operators ϕ_i on different statistics z stored for the observation window of w slots, it is possible to capture both instantaneous and long-term changes in the traffic profile of a user. In our experiments, we have applied a number of simple statistical functions including:

- The mean and standard deviation of z across the entire window.
- An exponential moving average filter in which the current feature is computed as $x_i^{(\tau)} = \alpha x_i^{(\tau-1)} + (1-\alpha)z^{(\tau)}$, where α is some constant $0 < \alpha < 1$ typically close to 1, with higher values discounting older observations faster.
- Shannon entropy which measures the uncertainty or unpredictability in the data, and is defined as $x_i^{(\tau)} = -\sum_{t=\tau-w-1}^{\tau} p_{z^{(t)}} \log p_{z^{(t)}}$, where $p_{z^{(t)}}$ is the probability of observing data item $z^{(t)}$ within the window, which can be estimated from the histogram of the data. A small entropy indicates deterministic behavior which is often associated with signaling anomalies [10,21].
- Anomaly score based on how close the measured quantities are to a range of values considered to be suspicious.

Once the input features for a slot have been computed, they are fused using a trained feed-forward RNN architecture such as the one presented in Fig. 2 to yield the final decision; the input neurons receive the features computed for the current time slot as exogenous excitatory signals, while all exogenous inhibitory signals are set to zero, and the output nodes correspond to the probabilities of the input pattern belonging to any of two traffic classes (i.e. attack or normal). The final decision about the traffic observed in the time slot is determined by the ratio of the two output nodes, which is q_{14}/q_{15} in the figure: it is classified as attack if the ratio is greater than 1 and normal otherwise. We have used an implementation of the RNN provided in [2].

2.2 Feature Selection

Selecting highly informative features for any classification problem is one of the most important parts of the solution. The features that we wish to use should

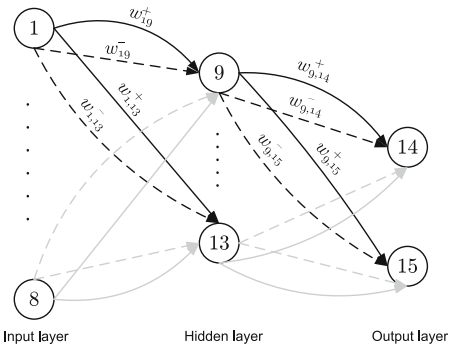

Fig. 2. The feed-forward RNN structure used for detection, with 8 input nodes, 5 hidden neurons and 2 output nodes corresponding to attack and normal traffic. The learning algorithm processes the input training patterns in sequence and updates the weights. The k-th training set consists of a feature vector $\mathbf{x}^{(k)} = (\Lambda_1^{(k)}, \cdots, \Lambda_8^{(k)})$ and its classification $\mathbf{y}^{(k)} = (y_{14}^{(k)}, y_{15}^{(k)})$ which is set to $(1, \epsilon)$ for attack and $(\epsilon, 1)$ for normal samples, where $\epsilon \simeq 0$. All other exogenous signals are set to zero.

capture the RRC signaling dynamics of users, be easy to measure or calculate without high computational or storage cost, and reflect both the instantaneous and long term trends of the traffic. We extract for each mobile under observation information related to inter-arrival times, lengths and destination IP addresses of packets, which have been suggested previously [6,17,22] as good indicators of signaling misbehavior:

Inter-arrival Times: RRC signaling occurs whenever the user equipment (UE) sends or receives packets following an inactivity period that exceeds an RRC timer. Thus, the volume of traffic exchanged by a UE does not map directly into signaling load which is more influenced by the frequency of intermittent transmissions. To capture this coupling between the data and RRC signaling planes, we define a *burst* as a collection of packets whose inter-arrival times are less than δ seconds, where δ is smaller than the RRC timers, typically in the order of few seconds. Thus, for a sequence of packets whose arrival instants are $\{t_1, t_2, \cdots\}$, we group all packets up to the n-th arrival into a single burst, where $n = \inf\{i : t_i - t_{i-1} > \delta\}$, and then proceed in a similar manner starting from the $(n+1)$-th packet arrival. Note that a burst may not necessarily generate signaling, even if it arrives after the time-out, due to possible network delays that may modify inter-arrival times of packets. However, packets within a single burst are likely not to trigger any signaling, while inter-arrival times of bursts will be correlated with the actual signaling load generated by the UE. In this manner, we remove any bias regarding the volume of traffic and focus on the frequency of potentially signaling-intensive communications.

The features based on the times between bursts are then calculated as follows. The algorithm stores the mean and standard deviation of the inter-burst times in each slot then, using the most recent w values, it computes (i) entropy of these

average values, (ii) moving average of the standard deviations, and (iii) moving average of an anomaly score for the average values computed based on the RRC timer T in the high bandwidth state. In particular, the anomaly score $a(z^{(t)})$ of the average inter-burst time in slot t is set to zero when $z^{(t)} < T$, reflecting the fact that such shortly spaced bursts may not have generated many RRC transitions; it is high when $z^{(t)}$ is slightly larger than T, indicating potentially resource-inefficient bursts; and it drops quickly when $z^{(t)}$ is few seconds larger than T. We obtain this effect using for example a Pareto or gamma density functions that assert $z^{(t)}$ must be greater than $T - \epsilon$ but not too much greater (controlled by adjusting the parameters of the density function).

Packet Size: The packet size distribution for a normal device can be markedly different from that of a device that runs a misbehaving application. For example, it is well-known that signaling storms can be triggered by failures in mobile cloud services [21] or peer-to-peer networks used by VoIP applications [6]. In such cases, the client application attempts to reconnect to its servers more frequently, causing significant increase in the number of TCP SYN packets sent by the user. This in turn changes the randomness associated with the size of packets, and can be used to identify misbehaving mobiles in the event of a signaling storm. Our algorithm computes the average size of packets sent by a UE within each slot, and evaluates a feature based on the entropy of the most recent w measurements.

Burst Rate: Another obvious characteristic of signaling storms is the sudden sustained rate acceleration of potentially harmful bursts generated by a misbehaving user. Moving average of the burst rate per slot and entropy of the rates across the observation window are used as features in order to capture, respectively, the frequent and repetitive nature of nuisance transmissions. Furthermore, a misbehaving application may change the traffic profile of a user in terms of the ratio of received and sent bursts (known as response ratio), as in the case of the outage induced storm described above where many SYN packets will not generate acknowledgments. Hence, we also use as a feature the mean of the response ratios within the window of w slots.

Destination Address: The number of destination IP addresses for a normally functioning mobile device can be very different from that of an attacker [17], whether the attack originates from the mobile network due to a misbehaving application, or from the Internet as in the case of unwanted traffic (e.g. scanning probes, spam, etc.) reaching the mobile network [22]. In the former, the number of destination IP addresses will be very small *compared to* the frequency of bursts, while in the latter this number is high. Thus we calculate the percentage of *unique* destination IP addresses contacted within each time slot, and use the average of the most recent w values as a feature.

2.3 User Filtering and Parameter Selection

Information about the "health" of network servers is typically available to mobile carriers in the form of KPIs, which can be fed to the algorithm to determine

the users that should be monitored (e.g. those attached to overloaded parts of a network). Also, using KPIs the detector can be switched off when signaling loads are below a certain threshold, effectively eliminating the need to continuously analyze users' traffic. Next we summarize the parameters of the RNN algorithm and discuss how they should be selected adaptively, based on both KPIs and RRC settings, and how the choice of each parameter influences the performance of the detector:

- Slot size Δ: This defines the resolution of the algorithm and the frequency at which classification decisions are made. It should be long enough for the measured statistical information to be significant, but not too long to make the algorithm react slowly to attacks. In our experiments we set $\Delta = 1\,\text{min}$.
- Window size $W = w\Delta$: This determines the amount of historical information to be included in a classification decision. The choice of the window size presents a trade-off between speed of detection and false alarm rate, since a small window makes the algorithm more sensitive to sudden changes in the traffic profile of a user, which in turn increases both detection and false alarm rates. This trade-off can be optimized by adjusting W according to the level of congestion in the control plane, with shorter windows for higher signaling loads to enable the algorithm to quickly identify misbehaving UEs. The value of w used in our experimental results is 5, but we also experimented with other values which confirmed the aforementioned observations.
- Maximum packet inter-arrival time within a burst δ: This should be selected based on the RRC timers, so that potentially resource-inefficient transmissions can be tracked. In our simulations of a UMTS network, the timers in DCH and FACH states are set to $T_1 = 6\,\text{s}$ and $T_2 = 12\,\text{s}$, respectively. We have evaluated different values of δ in $\frac{1}{2}\min(T_1, T_2) < \delta < \min(T_1, T_2)$, which all led to similar detection performance, but training time drops as δ is increased within this range.

3 Simulation Experiments

In this section, we evaluate the *detection performance* of our algorithm using the mobile network simulator described in [15]. Since the impact of signaling storms on mobile networks has been studied extensively in [3,9,15], we consider here a small scenario with 200 UMTS UEs in an area of $2 \times 2\,\text{km}^2$ which is covered by 7 Node Bs connected to a single radio network controller (RNC). The core network consists of the SGSN and the GGSN which is connected to 37 Internet hosts acting as application servers, including 5 for instant messaging, 30 web servers and 2 are contacted by 100 misbehaving UEs. Half of these 100 UEs are deliberate signaling attackers that know when RRC transitions occur, and they are used for training the RNN; the second half, used for testing, run a malfunctioning application or operating system that sends periodic messages whenever the user is inactive, with the transmission period set to be slightly larger than the DCH timer in order to increase the chances of triggering state transitions.

The RNN provides at the end of a time slot the odds of the input features belonging to attack behavior. Figure 3 shows the classifier output (top) and the actual RRC state transitions (bottom) of a *misbehaving* UE as captured during a simulation run. It can be observed that when the malfunctioning application is active, the number of state transitions significantly increases, with most transitions occurring between the FACH and DCH states. This alternating behavior causes excessive signaling load in the network, while predominantly generating normal traffic volume, rendering traditional DoS defense techniques ineffective. However, our detection mechanism is able to track very accurately the RRC state transitions of the UE, and to quickly identify when excessive signaling is being generated, although it does not directly monitor these transitions but rather infers them from the features that we have described. One can also see that the classifier's output sometimes drops close to 1 during an attack epoch, which is attributed to other normal applications generating traffic in those time instants, thus reducing the severity of the attack. As mentioned earlier, the detection speed and tolerance to signaling misbehavior can be adjusted by modifying the size of the observation window.

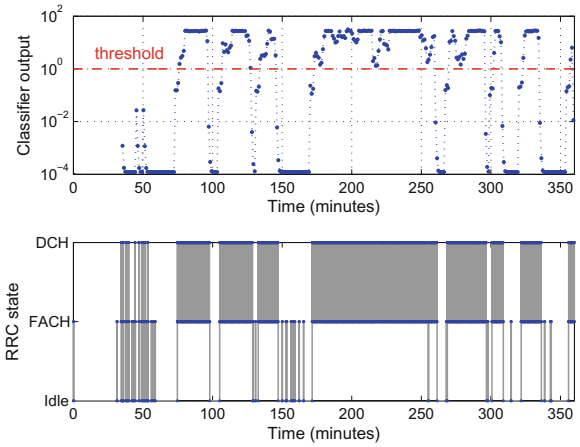

Fig. 3. Classifier output (top) and state transitions (bottom) for a misbehaving UE.

Next we examine in Fig. 4 how our algorithm performs when presented with a normal user that generates moderately more state transitions than the average normal user in our simulations. Interestingly enough, the classifier outputs a single alarm (out of 360 samples) when the corresponding state transitions are indeed excessive. Since the anomaly detection algorithm is supposed to be activate only when there is a signaling overload condition, such classification decisions may not always be considered as false alarms, as the goal would be to identify users that are causing congestion, regardless of whether they are attacking deliberately or not.

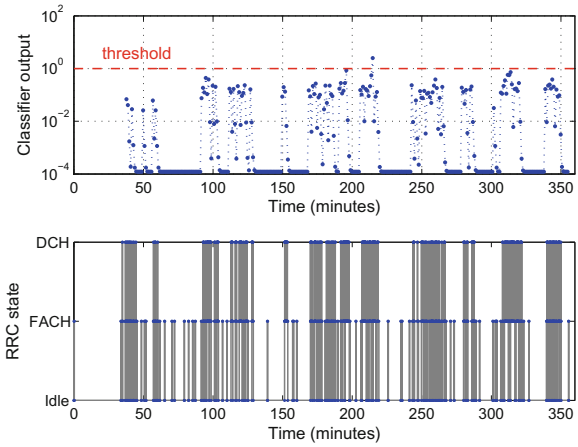

Fig. 4. Classifier output (top) and state transitions (bottom) for a heavy normal UE.

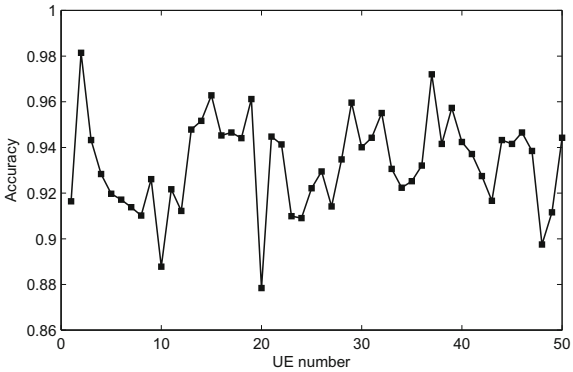

Fig. 5. The accuracy of the RNN algorithm, measured as the fraction of correct decisions over the activity period of 6 hours, for 50 misbehaving UEs.

Finally, Fig. 5 illustrates the accuracy of our classifier, namely the proportion of correct decisions (both true positives and true negatives) out of all test samples. The figure shows results for 50 UEs, where each data point represents the average of 360 classification decisions taken during the simulation experiment. For each UE, we assume that if it generates at least 1 attack packet within a time slot, then the corresponding output of the classifier should be larger than 1, otherwise a false decision is declared. The results indicate an accuracy between 88 % and 98 % with an average of 93 % over the 50 test cases. This fluctuation can be attributed to the fact that our algorithm does not classify an attack as such until few time slots have passed, and therefore misbehaving UEs with many silent periods will produce higher false positives; fortunately, these less aggressive UEs will generate lower signaling load.

4 Conclusions

This paper proposed an approach for real-time detection of signaling-intensive mobiles based on the random neural network (RNN) [11,12]. The algorithm relies on the analysis of IP packets at the edge of the mobile network to infer the signaling behavior of users; therefore, it does not require changes to network components and protocols, and can be used with standard traffic monitoring tools. In the algorithm, summary statistics about the behavior of a mobile user are collected and stored in a moving window at fixed time intervals (slots) and used in order to calculate expressive features that capture both sudden and long term changes in the user's behavior. The features for the most recent time slot are subsequently combined using a trained RNN to produce the final classification decision. Through simulations, we have demonstrated the effectiveness of the method in detecting quickly users that are causing signaling overloads in the network. The proposed approach is flexible, providing a number of parameters to optimize the trade-off between detection speed, accuracy and overhead. For instance, the size of the moving window and the frequency of statistical measurements (i.e. number of slots within the window) could be adjusted in real-time to respond to network conditions or to reflect the capacity of the network to tolerate a specific misbehavior. Future work will investigate the scalability of the approach both in terms of the processing power required to capture and process packets from a central location as well as the performance implications on the network equipment that mirrors the traffic.

Acknowledgments. This work was supported in part by the EU FP7 project NEMESYS under grant agreement no. 317888.

References

1. 3GPP TR 23.887: Machine-type and other mobile data applications communications enhancements (release 12) Technical report (2013). http://www.3gpp.org/DynaReport/23887.htm
2. Abdelbaki, H.: Random neural network simulator (RNNSIM v.2). Technical report,University of Central Florida (1999). http://www.cs.ucf.edu/~ahossam/rnnsimv2/rnnsimv2.pdf
3. Abdelrahman, O.H., Gelenbe, E.: Signalling storms in 3G mobile networks. In: Proceedings of IEEE International Conference on Communications (ICC), Sydney, pp. 1017–1022 (2014). doi:10.1109/ICC.2014.6883453
4. Abdelrahman, O.H., Gelenbe, E., Gorbil, G., Oklander, B.: Mobile network anomaly detection and mitigation: the NEMESYS approach. In: Gelenbe, E., Lent, R. (eds.) Information Sciences and Systems 2013. Lecture Notes in Electrical Engineering, vol. 264, pp. 429–438. Springer, Switzerland (2013). doi:10.1007/978-3-319-01604-7_42
5. AT&T: Best practices for 3G and 4G app development. Whitepaper (2012). http://developer.att.com/static-assets/documents/library/best-practices-3g-4g-app-development.pdf

6. Coluccia, A., D'Alconzo, A., Ricciato, F.: Distribution-based anomaly detection via generalized likelihood ratio test: a general maximum entropy approach. Comput. Netw. **57**(17), 3446–3462 (2013). doi:10.1016/j.comnet.2013.07.028
7. Ericsson: High availability is more than five nines (2014). http://www.ericsson.com/real-performance/wp-content/uploads/sites/3/2014/07/high-avaialbility.pdf
8. Ericsson: A smartphone app developers guide: Optimizing for mobile networks. Whitepaper (2014). http://www.ericsson.com/res/docs/2014/smartphone-app-dev-guide.pdf
9. Francois, F., Abdelrahman, O.H., Gelenbe, E.: Impact of signaling storms on energy consumption and latency of LTE user equipment. In: Proceedings of 7th IEEE International Symposium on Cyberspace safety and security (CSS), New York, 1248–1255 (2015). doi:10.1109/HPCC-CSS-ICESS.2015.84
10. Gabriel, C.: DoCoMo demands Google's help with signalling storm (2012). http://www.rethink-wireless.com/2012/01/30/docomo-demands-googles-signalling-storm.htm
11. Gelenbe, E.: Random neural networks with negative and positive signals and product form solution. Neural Comput. **1**(4), 502–510 (1989)
12. Gelenbe, E.: Learning in the recurrent random neural network. Neural Comput. **5**(1), 154–164 (1993)
13. Gelenbe, E., Abdelrahman, O.H.: Countering mobile signaling storms with counters. In: Mandler, B., et al. (eds.) IoT 360° 2015, Part I. LNICST, vol. 169, pp. 199–209. Springer, Heidelberg (2016)
14. Gelenbe, E., Loukas, G.: A self-aware approach to denial of service defence. Comput. Netw. **51**(5), 1299–1314 (2007)
15. Gorbil, G., Abdelrahman, O.H., Pavloski, M., Gelenbe, E.: Modeling and analysis of RRC-based signalling storms in 3G networks. IEEE Trans. Emerg. Topics Comput. **4**(1), 113–127 (2016). doi:10.1109/TETC.2015.2389662
16. GSMA: Smarter apps for smarter phones, version 4.0 (2014). http://www.gsma.com/newsroom/wp-content/uploads//TS-20-v4-0.pdf
17. Gupta, A., Verma, T., Bali, S., Kaul, S.: Detecting MS initiated signaling DDoS attacks in 3G/4G wireless networks. In: Proceeedings of 5th International Conference on Communication Systems and Networks (COMSNETS), Bangalore, pp. 1–6 (2013). doi:10.1109/COMSNETS.2013.6465568
18. Jiantao, S.: Analyzing the network friendliness of mobile applications. Technical report, Huawei (2012). http://www.huawei.com/ilink/en/download/HW_146595
19. Lee, P.P., Bu, T., Woo, T.: On the detection of signaling DoS attacks on 3G wireless networks. In: Proceedings of 26th IEEE International Conference on Computer Communications (INFOCOM), pp. 1289–1297 (2007). doi:10.1109/INFCOM.2007.153
20. Oke, G., Loukas, G., Gelenbe, E.: Detecting denial of service attacks with Bayesian classifiers and the random neural network. In: Proceedings of Fuzzy Systems Conference (Fuzz-IEEE), London, pp. 1964–1969 (2007)
21. Redding, G.: OTT service blackouts trigger signaling overload in mobile networks (2013). https://blog.networks.nokia.com/mobile-networks/2013/09/16/ott-service-blackouts-trigger-signaling-overload-in-mobile-networks/
22. Ricciato, F.: Unwanted traffic in 3G networks. ACM SIGCOMM Comput. Commun. Rev. **36**(2), 53–56 (2006). doi:10.1145/1129582.1129596
23. Telesoft Technologies: Mobile data monitoring. White paper (2012)
24. Timotheou, S.: The random neural network: a survey. Comput. J. **53**(3), 251–267 (2010). doi:10.1093/comjnl/bxp032