Impact of Generative Adversarial Networks on NetFlow-Based Traffic Classification

Maximilian Wolf(✉), Markus Ring, and Dieter Landes

Coburg University of Applied Sciences and Arts, 96450 Coburg, Germany
maximilian.wolf@stud.hs-coburg.de,
{markus.ring,dieter.landes}@hs-coburg.de

Abstract. Long-Short-Term Memory (LSTM) networks can process sequential information and are a promising approach towards self-learning intrusion detection methods. Yet, this approach requires huge amounts of barely available labeled training data with recent and realistic behavior. This paper analyzes if the use of Generative Adversarial Networks (GANs) can improve the quality of LSTM classifiers on flow-based network data. GANs provide an opportunity to generate synthetic, but realistic data without creating exact copies. The classification objective is to separate flow-based network data into normal behavior and anomalies. To that end, we build a transformation process of the underlying data and develop a baseline LSTM classifier and a GAN-based model called LSTM-WGAN-GP. We investigate the effect of training the LSTM classifier only on real world data and training the LSTM-WGAN-GP on real and synthesized data. An experimental evaluation using the CIDDS-001 and ISCX Botnet data sets shows a general improvement in terms of Accuracy and F1-Score, while maintaining identical low False Positive Rates.

Keywords: GANs · NetFlow · Intrusion detection · Synthetic data

1 Introduction

A recent study [1] interviewed 254 companies from different countries. As a result, annual average cyber-security costs add up to $11.7 millions, with a predicted increase of 22.7% per year. Further, 98% of the companies experienced malware attacks and 63% of them were attacked by botnets [1].

Problem. Network Intrusion Detection Systems (NIDS) are often used to detect malicious activities in company networks. Simultaneously, normal user behavior is subject to concept drift, new attack scenarios appear over time and malware signatures alter continuously. This situation complicates the detection of malicious activities for NIDS. Anomaly-based NIDS try to solve this challenge by modeling normal user behavior and highlighting deviations from known behavior

Á. Herrero et al. (Eds.): CISIS 2020, AISC 1267, pp. 393–404, 2021.
https://doi.org/10.1007/978-3-030-57805-3_37

as malicious. These systems require representative labeled training data which are often unavailable or only available to a limited extent.

Objective. This work's primary objective is the improvement of anomaly-based intrusion detection methods. To be precise, our research focuses on LSTM-based neural networks which classify network traffic in two classes, *normal* and *anomaly*. Based on the fact that representative labeled network traffic is limited, this work aims to generate realistic synthetic network data in order to increase the amount and variance of training data.

Approach and Contributions. NIDS analyze network traffic either on packet-based or flow-based level [18]. While packet-based approaches analyze payloads and exhibit good identification accuracy, flow-based approaches inspect only the metadata of a network communication to identify suspicious communication patterns within a network. Since processing flow-based data requires less resources, tolerates encrypted connections, and considers data privacy restrictions, this work focuses on flow-based network traffic in unidirectional NetFlow [4] format.

This work uses Generative Adversarial Networks (GANs) [6] to enrich existing data sets by generating flow-based network traffic and evaluates if the generated network traffic is able to improve subsequent intrusion detection methods. GANs consist of two neural networks, a Generator network G and a Discriminator network D. The Generator network G tries to create realistic data while the Discriminator network D tries to distinguish real from synthetically created data. Both networks are trained iteratively such that they get better and better until the Generator is able to create realistic data. In particular, this work uses Improved Wasserstein GANs (WGAN-GP) [8] which have been shown to be effective for flow-based network traffic generation [12]. Since flow-based network traffic consists of heterogeneous data and neural networks can only process continuous data, this work first designs a semantics-preserving transformation of the underlying data. Then, we propose a new GAN-based model (called LSTM-WGAN-GP classifier) where the Generator network G creates synthetic network traffic and the Discriminator network D (called Critic) classifies flow-based network traffic into two classes, *normal* and *anomaly*. We evaluate our approach experimentally using two public data sets, CIDDS-001 [13] and ISCX Botnet [3], and show that it achieves superior results compared to a baseline LSTM classifier. Our main contributions encompass the semantic-preserving transformation of flow-based network traffic such that it can be processed by neural networks, and the design of a new LSTM-WGAN-GP classifier.

Structure. In the following, we discuss related work regarding IDS and GANs. Then, the required foundations including NetFlows, LSTM networks and GANs are reviewed in Sect. 3. Section 4 presents the transformation of the NetFlow data and our new model. Finally, the experimental setup and the results are presented in Sect. 5 and discussed in Sect. 6. The last section concludes the paper.

2 Related Work

This section reviews related work in flow-based intrusion detection in general, and specifically on using GANs in this area.

Umer et al. [16] provide a comprehensive survey of flow-based intrusion detection. They categorize IDS as statistical IDS, machine learning IDS, and other techniques. They point out that most of the flow-based IDS are based on statistics and machine learning IDS need further attention. Additionally, many techniques are specialized to certain attacks, which limits their real-world integration and practical application. Moreover, several studies use non-representative data sets for validation, so that the real-world performance is questionable [16]. A practical application is presented by Qin et al. [10] who investigate the suitability of recurrent neural networks and convolutional neural networks for flow-based intrusion detection and achieve high detection accuracy.

Ring et al. [12] use Improved Wasserstein GANs to create synthetic flow-based network traffic. The authors considered all attributes of a flow and evaluated three different approaches to process categorical attributes. Ring et al. are able to create flows with high quality, but no sequential relationships between sequences of flows are considered in [12].

Rigaki and Garcia [11] use GANs to modify the communication patterns of malware in order to prevent detection through an Intrusion Prevention System (IPS). The IPS is based on Markov models and evaluates the attributes *bytes*, *duration* and *time-delta* of flow-based network traffic. The GAN is trained to imitate Facebook chat traffic. Then, the authors adapt the malware to match these traffic patterns and are able to trick the IPS. Another approach called MalGAN is presented by Hu and Tan [9]. MalGAN is able to create malware examples which are represented as 160-dimensional binary attributes. These samples are able to bypass anomaly-based intrusion detection methods.

Yin et al. [17] present the most notable work. They evaluate the performance of GANs as IDS and show that the performance of a classifier can be improved when GAN-generated data are used to extend the training data for a classifier. They use a multi-layer LSTM network trained on flows consisting of 16 extracted features. Their selected features are focused on 15 numerical features like the duration and the number of exchanged packets, while the transport protocol is the only categorical feature used. The authors train and evaluate their GAN model on the ISCX Botnet data set [3] and achieve an accuracy of 71% and a false positive rate of 16%.

While Rigaki and Garcia [11] and Hu and Tan [9] use GANs to adapt malware in order to trick IDS, we focus on the improvement of intrusion detection methods. In contrast to Yin et al. [17], this work includes additional categorical attributes of NetFlow and uses the evolved WGAN-GP from [8].

3 Foundations

3.1 NetFlow

The NetFlow file format [4] describes the exchanged data in a session between source and destination IP in an aggregated format. Therefore, the meta information of the connection is aggregated in time periods. NetFlow itself contains at least the timestamp of the first packet, the transmissions' duration, the transport protocol, the ports of source and destination, the bytes sent and the amount of packets sent [4]. Other than packet-based traffic captures, NetFlow does not contain any payload and requires less storage capacity.

NetFlows are unidirectional or bidirectional. Unidirectional NetFlows describe the connection always in one direction and contain the information about the data sent from a source to a destination. If the destination sends back data to the source, this connection is aggregated in a separate NetFlow. Bidirectional NetFlows aggregate data sent between source and destination into a single NetFlow. The data used in the experiments are unidirectional NetFlows.

3.2 Long- and Short-Term Memory Networks

The Long- and Short-Term Memory (LSTM) cell contains a memory block which is able to store information. This allows LSTM cells to process sequential data, since the additional memory cell adds information of the previous inputs to the current input. The memory block is connected to the input and output gate and regulates the information flow to and from the memory cell via activation functions. Additionally, the forget-gate is able to reset the memory block. This mechanism creates outputs based on current and previously seen data [5].

Similar to vanilla neural networks, LSTM cells can also be arranged as stackable layers to create multi-layer LSTM networks. The ability of processing sequential data can be improved by stacking LSTM-layers because this architecture can handle long-term dependencies in sequences better than single layer architectures, which is experimentally shown on acoustic sequence data in [14]. Since network data and especially attacks like slow Port Scans in the CIDDS-001 data set also contain long-term dependencies, which a classifier needs to handle properly, this architecture is beneficial.

3.3 Wasserstein Generative Adversarial Network

GANs encompass two competitive neural networks. A Generator network gets random inputs and outputs fake data samples. A Discriminator network attempts to distinguish real data from fake data samples. Based on the Discriminator's classification both networks are trained. The Generator's objective is to deceive the Discriminator, while the Discriminator tries to expose fake samples. This adversarial game causes constant adaption of the networks and improves their performance in the long run [6].

One drawback of vanilla GANs [6] is their instability during training which often leads to mode collapse [2]. Wasserstein GANs (WGAN) [2] use the continuous and differentiable Earth Movers (EM) distance and replace the Discriminator network with a Critic network. The weights of the neural networks need to be constrained to a compact space, to guarantee differentiability.

As suggested by Arjovsky et al. [2] weights can be constrained by clipping. In their experiments, this simple approach achieved good results, but can easily lead to vanishing gradients if the neural network consists of many layers or if no batch normalization is used. On the positive side, in the empirical evaluation mode collapse did not appear [2].

Nevertheless, WGANs suffer from unstable gradients which leads to exploding and vanishing gradients [8]. In order to avoid this, Gulrajani et al. proposed to constrain the gradient directly based on its input. This gradient penalty does not show gradient exploding or vanishing in the experimental evaluation of Gulrajani et al. [8]. In general the improved training of WGAN called WGAN-GP outperforms on various network architectures [8].

In consideration of the advantages, we build a LSTM-WGAN-GP in this paper which is based on the improved WGAN training method [8].

4 Approach

4.1 Data Transformation

Table 1 gives an overview of the flow encoding rules, while more specific information is provided subsequently.

Table 1. Encoding rules, "—" means that this field is not used

Attribute description			Example	
Field	Encoding	Vectors	Raw	Encoded
Date	—	0	—	—
Time	Normalize	1	17:10:29.057	0.7156134259259259
Duration	One-hot	15	1.064	0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0
Protocol	One-hot	3	TCP	0, 1, 0
Source IP	—	0	—	—
Source port	Binarize	16	4370	0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0
Destination IP	—	0	—	—
Destination port	Binarize	16	6667	0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1
Packets	Binarize	32	10	0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0
Bytes	Binarize	32	1052	0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0
Flags	Categorize	6	.AP.SF	0, 1, 1, 1, 0, 1
Labels	One-hot	2	attacker	1, 0

Date and Time. Due to the limited recording time of available data sets, the concrete date is discarded and only the daytime is extracted, since it may contain information about typical working hours or special activities at night. Regarding real-world applications, the concrete daytime contains important behavioral patterns and should not be discarded.

Time is normalized by seconds of day according to: $\frac{\text{time in seconds}}{\text{seconds of day}}$.

Duration. Due to the value distribution of the attribute duration (many very small values and few very large values), we decided to categorize it to discrete intervals in steps of 2^n. Based on an empirical analysis, we chose the following 15 intervals $[0, 2^{-5}]$, $(2^{-5}, 2^{-4}]$, $(2^{-4}, 2^{-3}]$, ..., $(2^6, 2^7]$ and $(2^7, \infty)$.

Transport Protocol. The transport protocol is one-hot encoded by considering the most popular protocols, i.e. *TCP*, *UDP*, and *ICMP*. The resulting vector consists of the components [isUDP, isTCP, isICMP]. If a NetFlow encompasses a different protocol, the vector is set to [0,0,0].

IP Addresses. IP addresses are not encoded into the data sets since an IP address that exhibited normal user behavior might be infected and act maliciously from then on. If the IP is used, the classification of a model could be biased when the IP information is used in the decision. This training behavior would negatively affect the generalization of the classification model according to the classification of different data with other IPs.

Ports. Source and destination ports are categorical attributes. Since a simple one-hot encoding would create too many values (2^{16}), we encode the source and destination port numbers by using their 16 Bit binary value represented in vectors, where each component is a digit of the binary number.

Packets and Bytes. Both attributes are encoded by using their 32 Bit binary value represented in vectors, where each component is a digit of the binary number. We preferred this representation since it achieved better results compared to a straightforward normalization in Ring et al. [12].

Flags. Only TCP-Flags [U, A, S, P, R, F] are extracted and encoded as a 6 component vector. Once a flag is set, it is encoded as 1, otherwise as 0.

Labels. The original labels of the CIDDS-001 [13] data set. i.e. [normal, attacker, victim], are transformed into normal and anomalous traffic where [normal, victim] are encoded as normal and [attacker] is encoded as anomaly. The ISCX Botnet data set [3] is labeled based on malicious (anomaly) and normal IPs, where the flows are encoded in accordance to their labels.

4.2 The LSTM-WGAN-GP Model

The data transformation process of the previous section leads to a 123-dimensional vector representation for each flow. Due to the long term dependencies in our underlying flow-based network data, we stack several LSTM-layers in our models. Further, we decided to use Wasserstein GANs [2] with the improved training method from [8]. The resulting architectures of the baseline LSTM-Classifier and our new LSTM-WGAN-GP model are shown in Fig. 1.

The baseline model and our model are trained on the same training data. While the baseline has to classify between anomaly and normal data, the Critic of LSTM-WGAN-GP has to classify between anomaly, normal and fake.

The fake classification is necessary due to the LSTM-WGAN-GP setup where the Generator attempts to create fake data which can not be distinguished from real data by the Critic. The Critic shall distinguish between fake samples from the Generator and real data. In order to train the Generator and the Critic, the classification error of the Critic is used to optimize the Critic and the Generator.

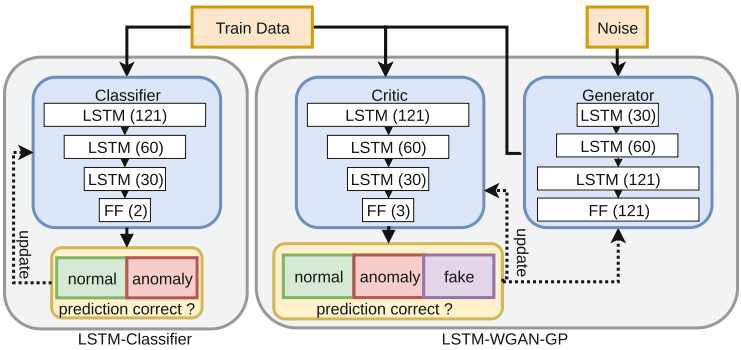

Fig. 1. The experimental setup for the training. Types of neurons are defined by LSTM or FF and the number of neurons per layer is given in brackets.

Based on the assumption that the Generator is able to create data similar to real data, these additional synthetic data can be used to train the Critic with training data which are close to real samples but not exact copies. By using this data for the training of the Critic, there is the possibility of increasing the classification quality of normal and anomaly traffic with synthetic generated data.

After the training phase, both models can be used for intrusion detection. According to the LSTM-WGAN-GP's Critic, only the classification between normal and anomaly is evaluated, because the fake classification is not interesting in terms of normal anomaly classification in this setup.

5 Experiments

5.1 Experimental Setup

The model's composition used in the experiments is shown in Fig. 1.

Each network consists of three LSTM layers and one Feed Forward (FF) layer. In prior tests, architectures with a single LSTM layer showed unsuitable classification results. This behavior is similar to the findings of Sak et al. [14] who recommend multiple stacked layers in LSTM architectures for the classification of long-term dependencies in sequential data.

Based on the findings of Greff et al. [7] who showed that the learning rate impacts the training behavior heavily, the learning rates [0.1, 0.05, 0.01, 0.001, 0.0001] are tested in a small scale study with a sequence length and a batch size of 64. The Generator learning rate was set to half of the Critic's. In this study, each model was trained three times for 50 epochs and the best model was selected based on its accuracy. The study shows that the learning rate of 0.05 for the Classifier and Critic and 0.025 for the Generator provided the best results for both data sets.

For the final evaluation, the models were trained for 50 epochs on the CIDDS-001 data set [13] and 200 epochs on the smaller, but more diverse ISCX Botnet data set [3]. For training on CIDDS-001, week one (around 8.5 million flows) is used, while the evaluation is done on week two (around 10 million flows). Both weeks encompass clients that perform attacks like Ping-Scans, Port-Scans, Brute-Force and Denial of Service attacks and are composed of 91% normal and 9% malicious traffic.

The ISCX Botnet data set consists of a training and a test set, where the test set includes a wider range of Botnet activity. The training set contains seven botnet types and the test set contains 16 types of botnets. ISCX Botnet is originally packet based and is converted to approx. 500.000 flows in the training and test split. The flows then need to be labeled based on malicious IPs and connections[1] which results in non-balanced subsets.

5.2 Evaluation

The models are evaluated by the standard data mining evaluation metrics Accuracy, Precision, Recall and F1-Score. Further, we consider the number of false alarms and the number of detected attacks which are important metrics for an intrusion detection method. When a sequence of NetFlows contains a type of attack and is labeled as anomaly by a model, the attack type is considered as discovered. Due to the randomly initialized weights of the neural networks, which is useful for error back-propagation, the classification performance differs slightly in equal hyper-parameter configurations. Given this effect, each model is trained ten times which allows calculating mean values and standard deviations.

[1] The malicious IPs are listed at: https://www.unb.ca/cic/datasets/botnet.html.

5.3 Results

Table 2 shows the results of our experiments.

On both data sets, the LSTM-WGAN-GP model achieves higher scores for Accuracy, Recall and F1-Score. The scores for the metrics Precision and False Positive Rate are similar. The LSTM-WGAN-GP increased the number of detected attacks on the ISCX Botnet data set, which means that one more type of botnet is detected. Both models discover all attacks in the CIDDS-001 data set.

Table 2. Evaluation metrics for the CIDDS-001 and ISCX Botnet data set. The cells contain the mean value and standard deviation over ten runs.

Metric	CIDDS-001		ISCX Botnet	
	LSTM-Classifier	LSTM-WGAN-GP	LSTM-Classifier	LSTM-WGAN-GP
Precision	0.9988 ± 0.0005	0.9987 ± 0.0006	0.9834 ± 0.0203	0.9807 ± 0.0274
Accuracy	0.8122 ± 0.0360	0.8190 ± 0.0348	0.6884 ± 0.0438	0.6900 ± 0.0324
Recall	0.3543 ± 0.1241	0.3779 ± 0.1200	0.2828 ± 0.1032	0.2866 ± 0.0746
F1-Score	0.5112 ± 0.1434	0.5375 ± 0.1382	0.4286 ± 0.1433	0.4387 ± 0.0962
FPR	0.0002 ± 0.0001	0.0002 ± 0.0001	0.0041 ± 0.0047	0.0041 ± 0.0067
False Alarms	21.5 ± 13.6076	25.2 ± 13.8788	18.5 ± 21.345	18.7 ± 30.4779
Detected Attacks	4/4	4/4	7.1/16	8.1/16

6 Discussion

Data set specific. Table 2 indicates that the LSTM-Classifier and LSTM-WGAN-GP achieve acceptable results for CIDDS-001, but not for the ISCX Botnet data set.

This may be explained by the structure of the data sets. CIDDS-001 contains equal attack types with equal behavior in the training and the test set. The ISCX Botnet data set contains novel botnets and therefore novel behavior in the test set. The network architecture used in the experiments could not generalize well enough to detect novel attacks or attacker behaviors. Overall, the training subset of the ISCX Botnet data set is not representative enough that the used network architectures could generalize well enough to detect novel attacks.

Another reason could be the integrity of the data sets. CIDDS-001 has a high integrity because it is recorded in one consistent network architecture and consistent normal user behavior. ISCX Botnet data set is a mixture of multiple data sets that are synthetically combined, which increases the diversity of the attacks, but demolish the network infrastructure in terms of integrity and consistency.

Model specific. Based on the results of Table 2, the models demonstrate a high precision score (Prec), but low recall scores (Rec). Taking the number of detected attacks also into account, reveals that the models detect the attacks, but not all partial sequences of an attack.

Considering the standard data mining evaluation measures, the comparison of the LSTM-Classifier and LSTM-WGAN-GP performance shows a marked improvement. By additionally taking the application-specific metric of detected attacks into account, an improvement can certainly be detected.

The LSTM-WGAN-GP models achieve a higher Accuracy and F1-Score on both data sets. On the CIDDS-001 data set all attacks are discovered by both models with similar False Positive Rates (FPR) and false alarms. The LSTM-WGAN-GP increased the number of Botnet detection on the ISCX Botnet data set and retained the number of false alarms. The FPR is the most important metric and demonstrates very low scores on both models and data sets. The increased variance in the data makes the classification more resistant towards outliers, which could be miss-classified otherwise.

Yin et al. [17] achieved an accuracy of 71% and a False Positive Rate of 16% on the ISCX Botnet data set. Compared to their results, our model reduces the False Positive Rates enormously to 0.4%. Simultaneously, the accuracy decreased only from 71% to 69%. According to Sommer and Paxson [15], decreasing the number of false alarms is one of the most challenging aspects of anomaly-based intrusion detection.

Overall. A general challenge is the qualitative evaluation of the generated data by LSTM-WGAN-GP. The generated NetFlow data are sequential data, which have a high dimension of freedom in terms of combinatorial arrangement of NetFlows in order to create sequences or correct NetFlows itself. This means that there are several correct combinations and it is hard to determine which NetFlows are correct, given the training data.

All data sets are recorded in different networks with different setups and participants which makes it more difficult to tell if a generated NetFlow is likely to appear in that network setup and can be considered real or not. This question requires deeper research in the future.

7 Conclusion

NIDS are used to discover attacks in company networks. Neural networks are able to handle novel attacks by learning normal behavior and need not be updated with new signatures continuously.

This work investigates the effect of the classification performance of multi-layer LSTM-Classifiers which are trained exclusively with real or with a combination of real and synthetically generated data. The synthetic training data are generated using Improved Wasserstein Generative Adversarial Networks, which are successfully used for syntactically correct NetFlow creation.

In order to feed the heterogeneous NetFlow data to neural networks, they have been transformed into a continuous representation. The models have been tested on two different data sets: CIDDS-001 and ISCX Botnet. While the CIDDS-001 incorporates a consistent behavior of normal user actions, it contains a low diversity of attacks. On the contrary, the ISCX Botnet data set

contains diverse botnets and their attack behavior, but lacks on normal behavior consistency, since it is a combination of multiple different botnet data sets with different network structures and user activity. In the experimental setup, a baseline Classifier consisting of a multi-layer LSTM is compared with our new multi-layer LSTM-WGAN-GP.

The experiments revealed that the classification performance is improved in general. The important False Positive Rate and the number of false alarms constant remain very low. Based on CIDDS-001 all types of attacks are detected. On the more diverse ISCX Botnet data set the LSTM-WGAN-GP increases the number of detected botnets.

In the future, we want to evaluate another GAN architecture which creates flow sequences and the traffic investigation is done in an independent LSTM network afterwards. Further, we want to extend the evaluation of our models.

Acknowledgements. This work is funded by the Bavarian Ministry for Economic affairs through the OBLEISK project. Further, we gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

References

1. Accenture, Institute, P.: 2017 Cost of Cyber Crime Study. https://www.accenture.com/t20170926T072837Z__w__/us-en/_acnmedia/PDF-61/Accenture-2017-CostCyberCrimeStudy.pdf (2017). Accessed 16 Jul 2019
2. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein GAN. ArXiv **abs/1701.07875** (2017)
3. Beigi, E.B., Jazi, H.H., Stakhanova, N., Ghorbani, A.A.: Towards effective feature selection in machine learning-based botnet detection approaches. In: IEEE Conference on Communications and Network Security (CNS), pp. 247–255. IEEE (2014)
4. Claise, B.: Cisco Systems NetFlow Services Export Version 9. RFC 3954, Internet Engineering Task Force (2004). https://tools.ietf.org/html/rfc3954
5. Gers, F.A., Schmidhuber, J.: Recurrent Nets that Time and Count. In: IEEE Int. Joint Conference on Neural Networks (IJCNN), pp. 189–194 vol.3 (2000)
6. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Advances in Neural Information Processing Systems (NIPS), pp. 2672–2680 (2014)
7. Greff, K., Srivastava, R.K., Koutník, J., Steunebrink, B.R., Schmidhuber, J.: LSTM: A Search Space Odyssey. IEEE Trans. Neural Netw. Learn. Syst. **28**(10), 2222–2232 (2016)
8. Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., Courville, A.C.: Improved training of Wasserstein GAN. In: Advances in Neural Information Processing Systems (NIPS), pp. 5769–5779 (2017)
9. Hu, W., Tan, Y.: Generating adversarial malware examples for black-box attacks based on GAN (2017). arXiv preprint arXiv:1702.05983
10. Qin, Y., Wei, J., Yang, W.: Deep learning based anomaly detection scheme in software-defined networking. In: Asia-Pacific Network Operations and Management Symposium (APNOMS), pp. 1–4. IEEE (2019)

11. Rigaki, M., Garcia, S.: Bringing a GAN to a knife-fight: Adapting malware communication to avoid detection. In: Deep Learning and Security Workshop, IEEE Security & Privacy Workshops (SPW), pp. 70–75 (2018)
12. Ring, M., Schlör, D., Landes, D., Hotho, A.: Flow-based network traffic generation using generative adversarial networks. Comput. Secur. **82**, 156–172 (2019)
13. Ring, M., Wunderlich, S., Grdül, D., Landes, D., Hotho, A.: Flow-based benchmark data sets for intrusion detection. In: European Conference on Cyber Warfare and Security (ECCWS), pp. 361–369. ACPI (2017)
14. Sak, H., Senior, A.W., Beaufays, F.: Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In: Conference of the International Speech Communication Association (INTERSPEECH), pp. 338–342 (2014)
15. Sommer, R., Paxson, V.: Outside the closed world: On using machine learning for network intrusion detection. In: IEEE Symposium on Security & Privacy, pp. 305–316. IEEE (2010)
16. Umer, M.F., Sher, M., Bi, Y.: Flow-based intrusion detection: Techniques and challenges. Comput. Secur. **70**, 238–254 (2017)
17. Yin, C., Zhu, Y., Liu, S., Fei, J., Zhang, H.: An enhancing framework for botnet detection using generative adversarial networks. In: International Conference on Artificial Intelligence and Big Data (ICAIBD), pp. 228–234 (2018)
18. Zhao, D., Traore, I., Sayed, B., Lu, W., Saad, S., Ghorbani, A., Garant, D.: Botnet detection based on traffic behavior analysis and flow intervals. Comput. Secur. **39**, 2–16 (2013)