Attacks detection on Sampled Netflow traffic through image analysis with Convolutional Neural Networks (CNN)

Alberto Fernández de Retana, Alberto Miranda, Ángel Manuel Guerrero, and Camino Fernández-Llamas

afernf@unileon.es, amirg@unileon.es, am.guerrero@unileon.es
camino.fernandez@unileon.es
University of Leon, Spain.

Abstract. The interest in attacks detection has increased significantly in recent years together with the internet traffic and connections. Due to the big amount of packages, it is not feasible to analyze the payload of every packet that goes through the network. In order to have a statistical solution, the NetFlow protocol was designed. The payload of the packets is not included in the information stored by this protocol, making the detection of malicious attacks more challenging. Furthermore, to alleviate the performance penalty generated by the NetFlow on the routers, the Sampled NetFlow was developed. Sampled NetFlow allows the system administrators to define the interval in which these flows are going to be gathered. In the literature, there are several approaches that make use of traditional Machine Learning methods like KNN or SVM. To the best of our knowledge, there is currently no study attempting to probe Convolutional Neural Network on Sampled NetFlow. In this paper, we present the results obtained using Convolutional Neural Network on flows of Sampled NetFlow v5 to fill this gap. Our approach was able to obtain 94.15% of accuracy on sampling rate of 500. Additionally the limitations of this technique are going to be discussed if the interval of the Sampled Netflow is greather than 500.

Keywords: Attack detection, Sampled Netflow data, Convolutional Neural Network

1 Introduction

The NetFlow technology was developed in an attempt to collect information about IP traffic in a simple way, thus being able to track the flows. NetFlow, which allows us to obtain several characteristics of the packets that pass through network devices, has become an industry standard protocol. Nowadays, it is used by the majority of routers in the world. This protocol has several versions but the most widely used standard versions are versions 5 and 9, which gather data such as the source IP, the destination IP or the source and destination ports, among others. One of the consequences of the design of this protocol is that we are unable to inspect the payload of the packets, called deep packet inspection (DPI). Additionally, in order to avoid the saturation of the routers the Sampled NetFlow is used. This allows the system administrators to define the threshold in which the flows of Netflow are gathered.

As of today, due to the impossibility of exhaustively analyzing each and every one of the network packets, the interest in gathering information based on the network flows is growing dramatically. One of the first topics to be studied was the traffic classification [7]. The accuracy obtained without sampling was 96.67% [2] and the accuracy with sampling was 51.02% [4]. Nowadays, the researchers go further trying to recognize worm attacks [1] or building generic detection systems [16].

One of the latest topics in this research area is the detection of network attacks, such as DDoS or Ports scanning. For this purpose, a large number of papers using machine learning, e.g. KNN, SVM, have been published [3,5,8,12–15]. However, there are few published research [9] working with the increasingly popular Convolutional Neural Networks, in particular, with the well known architecture called ResNet. That paper describes a methodology with better results than some traditional methods described before, reaching a 95.86% accuracy. Despite that, the paper was written ambiguously, not describing the version of NetFlow used nor the columns selected on the research, among other examples. This method of Convolutional Neural Networks applied in Netflow for detection networks attacks is a new field opened by this group of researchers. Our work continues the research done in this new field working on Sampled Neflow.

In summary, we present results applying Convolutional Neural Network method on Sampled Netflow (version 5). Achieving 94.15% of accuracy in 500 of sampling rate. Furthermore, the limitations generated by the sampling are discussed getting a low accuracy with a big value of interval in sampling. This is the main contribution due to the fact that sampling is used in real environments.

This paper is structured as follows: Section 2 explains the different datasets used in the project and describes both image generation and model training. Additionally, Section 3 describes the results obtained by our models and their time consumption. Finally, Section 4 presents our conclusions and discusses about the further work in the area and the limitations.

2 Materials and methods

In this section, the core of the research is explained. The different parts of the project are explained separately below, from the generation of images, to the training of the network. We follow a three-step methodology (Figure 1).

First, data gathering of NetFlow flows is done by using DOROTHEA, a NetFlow dataset generator explained below. Next, in the image generation phase NetFlow data (1D) is converted to images (2D). Finally, the model of CNN is trained, and later validated.

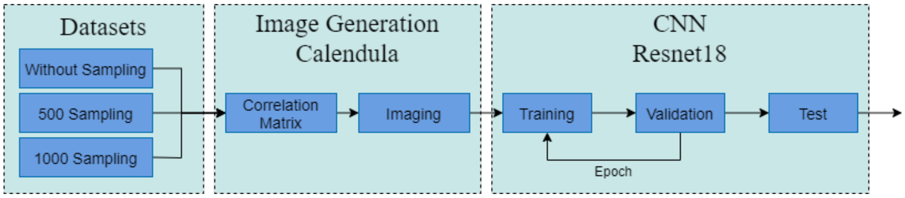

Fig. 1. Project Architecture

2.1 Data gathering

In this section, we explain the datasets used in our research. In our work, we have a different datasets for each sampling. The datasets were obtained using DOROTHEA, an open-source tool to gather netflow datasets [3]. We used three types of dataset, the first one without sampling, the next one with a sampling rate of 500 and the last one with 1000. All of them are NetFlow version 5. Datasets were taken in similar duration periods, but as a consequence of sampling, the size of datasets are smaller. The first dataset is used in order to probe that we have a similar accuracy to previous research [9], verifying that the solution of CNN on NetFlow works. We have used a dataset with a sampling rate of 1000 as it resembles a real production environment due to the fact that it is usually the most common sampling rate on big networks. Finally, we chose a sampling rate of 500 because it is a middle value between the real environment and the NetFlow without sampling. In the Table 1, the total number of images generated for the different datasets are shown. The dataset without sampling is the bigger one with a total of 1M images separated in training and validation phases. The dataset with a sampling rate of 500 has about 10k images. Finally, the dataset with a sampling rate of 1000, that imitates a real environment, has about 5000 images. All the datasets have a total of 12GB of disk usage. The dataset is publicly available with the project.

2.2 Image generation

Imaging is one of the most important aspects of this research field. For the training and testing of the network, it is very important to be able to correctly represent all the flow data, in such a way that the network can be able to catalog

Table 1. Datasets of images used to train the models.

	Training		Validation	
Sampling rate	Attack	Normal	Attack	Normal
-	241.910	236.855	303.907	303.907
500	2742	5506	2727	2727
1000	1428	1428	1292	1292

the flows. For this task, we have used the parallel computing cluster of Calendula. Calendula is the supercomputer of SCAYLE (Supercomputing Center of Castilla and León) with a total of 397 TFlops which has allowed us to generate a large number of images in a reduced time for the generation of the model. This phase is one of the most critical ones, because it is a process that requires a big amount of time. On our approach, we decided to run 36 processes simultaneously. The creation of the images from the datasets took a few weeks executing on Calendula.

To carry out the task of representing the values properly, only numerical values have been used, discarding other NetFlow features with different types of values. Regarding the eliminated characteristics, in addition to the non-numerical ones, we have also discarded the ones which are related to the exact time in which the flow was extracted, the next hop of the packet or the IP from which the flow was extracted, since they do not provide any type of relevant information for the image. It should be noted, that some columns, e.g. *src_as*, *dst_as*, *src_mask* or *dst_mask*, may include relevant information on datasets retrieved from real infrastructures. The source IP and the destination IP are converted to decimal values before the image is generated. Finally, NetFlow v5 features used in this research are: *dpkts*, *doctets*, *first*, *last*, *srcaddr*, *dstaddr*, *input*, *output*, *srcport*, *dstport*, *prot*, *tos* and *tcp_flags*. Therefore, the official NetFlow v5 columns discarded due to the lack of information are: *unix_secs*, *unix_nsecs*, *sysuptime*, *exaddr*, *engine_type*, *engine_id*, *nexthop*, *src_mask*, *dst_mask*, *src_as*, *dst_as*.

First of all, a correlation matrix is generated with all the characteristics defined before. In order to create the correlation matrix, the Pandas library [10] is used. After that, each of the columns is surrounded by their eight more correlated values making a matrix of 3x3. Finally, all the 3x3 matrices are joined on a big matrix called Surrounding Correlation matrix (SC matrix). This methodology was presented by a group of researchers [9] from Purdue University. Finally, this SC matrix values are replaced by each of the flows, generating an image similar to Figure 2.

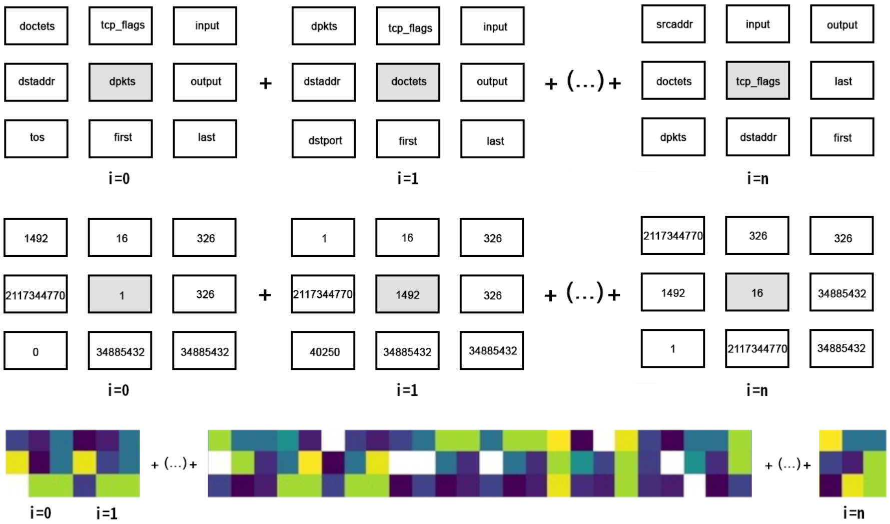

Fig. 2. Netflow Matrix Numeric Representation

2.3 Training the CNN

The training phase was done using the popular library PyTorch [11]. This tool is a machine learning library which is focused on usability and speed, allowing us to train ResNet-18. Our approach was to use the model offered by PyTorch, freezing all the layers except the final one. This methodology allowed us to train the model with a maximum performance. The Convolutional Neural Network model used was ResNet-18 [6]. This architecture is a reliable model used in the previous work in this field by Purdue's researchers. ResNet-18 is well-known to be used for classification [6].

Before feeding the model with data, the images need to be resized to 224x224 in order to fit the requirements of the network. After that, the last layer of the model is trained using a defined number of epochs in order to get the best accuracy possible. In each of the epochs there are two phases defined, training and then validation phase. This technique lets us save the best model.

3 Results

In this section, a detailed overview of the results are presented. The results are obtained using the methodology explained above. First of all, the accuracy of Netflow without sampling is 96.58%. This accuracy, that is greater than the previous work [9], demonstrates that Convolutional Neural Network (CNN) could be used on Netflow with good results. Moreover, with this dataset, we have double-checked the previous research. This dataset is the bigger one, so the training of the model was the largest in terms of time. In other words, the training with this dataset delays approximately 11 hours. The accuracy of Convolutional Neural Network (CNN) is similar to any other method used in the literature, e.g. KNN, SVM [3, 5, 8, 12–15]. In the table 2 the accuracies of other classifiers are shown.

Table 2. Accuracy from the literature.

Classifier	Accuracy
CNN	96.58%
KNN	96.41%
LR	94.83%
SGD	92.10%
OvR	93.23%
CART	90.18%
RF	90.18%
AB	90.18%
BRBM	78.22%
QDA	51.71%
LDA	51.71%
NB	51.71%
BC	51.71%

The second dataset contains the data with a sampling rate of 500. Like we explained before, this sampling rate is a middle value between the actual literature and the real environments. Using this second dataset, the accuracy was 94.15%, decreasing a 2.52% below the results without sampling. This value shows that Convolutional Neural Network (CNN) could be applied successfully on Sampled NetFlow.

The last one is the simulation of a real environment with a sampling rate of 1000. This is the sampling rate that is normally applied in real routers to avoid the degradation of the performance due to their saturation. The accuracy obtained by the model with this dataset is slightly disappointing, getting a 50.11%. This supposes a 48.12% decrease below the first dataset and a 46.78% below the second one with 500 of sampling rate. Although the results are not encouraging, they open an opportunity to research new methodologies in this way.

In summary, the accuracy decreased drastically when the rate of Sampled NetFlow is greater than 500. The reason for that is the amount of information that is lost when the NetFlow is sampled. In the Figure 3 the accuracies of the datasets are shown.

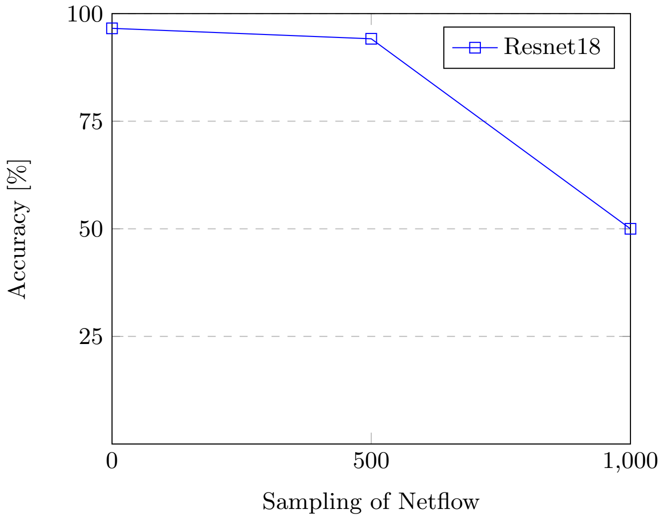

Fig. 3. Accuracy obtained by CNN.

4 Conclusions and further work

This paper presents the first work using Convolutional Neural Networks on Sampled Netflow. Moreover, in this research we describe the limitations that Convolutional Neural Networks have on Sampled NetFlow. This is an issue that also appear in other similar approaches [4]. Using Convolutional Neural Networks we obtain a 94.15% of accuracy with a sampling rate of 500. We believe these insights can be the foundation for more systematic future works in this field. Additionally, we have made the project[1] publicly available, allowing other researchers to double-check and improve our findings.

Our work shows that Convolutional Neural Networks (CNN) accuracy decreases when the interval of Sampled Netflow increases. Furthermore, with a simulated real environment dataset with a sampling rate of 1000, the accuracy is greatly decreased. However, there are smaller networks where the sampling rate is 500 or less in which this work could be applied. On a future work, the model is going to be tested with real data to compare the accuracy obtained.

[1] https://niebla.unileon.es/AlbertoMGV/cnn/

Besides, another line of research is the study of the information added to the model for each of the features of NetFlow. In this line of work, different architectures of CNN or different methods of training could be studied. Furthermore, the performance of the image creation process is critical in order to implement this methodology on a real environment, so this could be a future limitation of this new approach.

5 Acknowledgement

The research described in this article has been partially funded by Instituto Nacional de Ciberseguridad de España (INCIBE), under the grant “ADENDA 4: Detección de nuevas amenazas y patrones desconocidos (red Regional de Ciencia y Tecnología)”, addendum to the framework agreement INCIBE–Universidad de León, 2019–2021; and by the Spanish Ministry of Science, Innovation, and Universities RTI (RTI2018-100683-B-I00) grant.

References

1. Abdulla, S., Ramadass, S., Taha, A., Amer, N.: Article: Setting a worm attack warning by using machine learning to classify netflow data. International Journal of Computer Applications 36, 49–56 (10 2011)
2. Bakhshi, T., Ghita, B.: On internet traffic classification: A two-phased machine learning approach. Journal of Computer Networks and Communications 2016, 1–21 (2016)
3. Campazas-Vega, A., Crespo-Martínez, I.S., Guerrero Higueras, A.M., Fernández Llamas, C.: Flow-data gathering using netflow sensors for fitting malicious-traffic detection models. Sensors 20(24), 7294 (2020)
4. Carela-Español, V., Barlet-Ros, P., Cabellos-Aparicio, A., Solé-Pareta, J.: Analysis of the impact of sampling on netflow traffic classification. Computer Networks 55(5), 1083–1099 (2011)
5. Casas, P., Mazel, J., Owezarski, P.: Unsupervised network intrusion detection systems: Detecting the unknown without knowledge. Computer Communications 35(7), 772–783 (2012)
6. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
7. Jiang, H., Moore, A.W., Ge, Z., Jin, S., Wang, J.: Lightweight application classification for network management. Proceedings of the 2007 SIGCOMM workshop on Internet network management - INM 07 (2007)
8. Kanda, Y., Fontugne, R., Fukuda, K., Sugawara, T.: Admire: Anomaly detection method using entropy-based pca with three-step sketches. Computer Communications 36(5), 575–588 (2013)
9. Liu, X., Tang, Z., Yang, B.: Predicting network attacks with cnn by constructing images from netflow data. 2019 IEEE 5th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart

Computing, (HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS) (2019)

10. Mckinney, W.: Pandas: a foundational python library for data analysis and statistics. Python High Performance Science Computer (01 2011)
11. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library (2019)
12. Shone, N., Ngoc, T.N., Phai, V.D., Shi, Q.: A deep learning approach to network intrusion detection. IEEE Transactions on Emerging Topics in Computational Intelligence 2(1), 41–50 (2018)
13. Tran, Q.A., Jiang, F., Ha, Q.M.: Evolving block-based neural network and field programmable gate arrays for host-based intrusion detection system. 2012 Fourth International Conference on Knowledge and Systems Engineering (2012)
14. Tran, Q.A., Jiang, F., Hu, J.: A real-time netflow-based intrusion detection system with improved bbnn and high-frequency field programmable gate arrays. 2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications (2012)
15. Winter, P., Hermann, E., Zeilinger, M.: Inductive intrusion detection in flow-based network data using one-class support vector machines. 2011 4th IFIP International Conference on New Technologies, Mobility and Security (2011)
16. Zhenqi, W., Xinyu, W.: Netflow based intrusion detection system. 2008 International Conference on MultiMedia and Information Technology (2008)