Network Anomaly Detection Using Parameterized Entropy

Przemyslaw Bereziński, Marcin Szpyrka, Bartosz Jasiul, Michal Mazur

Network Anomaly Detection Using Parameterized Entropy

Przemysław Bereziński[1], Marcin Szpyrka[2], Bartosz Jasiul[1], and Michał Mazur[1]

[1] Military Communication Institute
C4I Systems' Department
ul. Warszawska 22a, 05-130 Zegrze, Poland
{p.berezinski, b.jasiul, m.mazur}@wil.waw.pl
[2] AGH University of Science and Technology
Department of Applied Computer Science
al. Mickiewicza 30, 30-059 Kraków, Poland
mszpyrka@agh.edu.pl

Abstract. Entropy-based anomaly detection has recently been extensively studied in order to overcome weaknesses of traditional volume and rule based approaches to network flows analysis. From many entropy measures only Shannon, Titchener and parameterized Renyi and Tsallis entropies have been applied to network anomaly detection. In the paper, our method based on parameterized entropy and supervised learning is presented. With this method we are able to detect a broad spectrum of anomalies with low false positive rate. In addition, we provide information revealing the anomaly type. The experimental results suggest that our method performs better than Shannon-based and volume-based approach.

Keywords: anomaly detection, entropy, netflow, network traffic measurement

1 Introduction

The number of anomalies in IP networks caused by wormlike activities is growing [2]. Widely used security solutions based on signatures or rules like firewalls, antiviruses and intrusion detection systems do not provide sufficient protection because they do not cope with evasion techniques and not known yet (0-day) attacks [12], [13]. Therefore, network anomaly detection as one of possible solutions is becoming an essential area of research. Anomaly detection is an identification of observations which do not conform to an expected behavior. In a supervised anomaly detection a labeled data set that involves training a classifier is required.

There are many problems with anomaly detectors which have to be addressed. The main challenge is setting up a precise boundary between normal and anomalous behavior to avoid high false positive error rate or low detection rate. Another problems are long computation time, anomaly details extraction and root-cause identification [7]. In our previous work [4], some generalizations of entropy were described in details and preliminary results of using parameterized entropies were presented. In this paper, we make two major contributions. Firstly, we present our method and results in comparison with Shannon-based and volume-based approach. Secondly, we describe data set as well as the method we used to generate anomalies.

2 Related work

Entropy-based network anomaly detection has been a hot research topic recently. This approach relies on traffic feature distributions [16]. In the past, anomalies were treated as deviations in the traffic volume [11]. The problem is that not all anomalous network activities result in substantial traffic volume change. Moreover, Brauckhoff [6] proved that entropy-based approach with traffic feature distributions performs better then volume-based where sampling of flows is used. Several traffic feature distributions, i.e. header-based (addresses, ports, flags), volume-based (host or service specific percentage of flows, packets and bytes) and behavior-based (in/out connections for particular host) have been suggested in the past [17], [21]. However, it is unclear which feature distributions perform best. Nychis in [17], based on his results of pairwise correlation reported dependencies between addresses and ports and recommended the use of volume and behavior-based feature distributions. In opposite, Tellenbach in [21] reported no correlation among header-based features. Parameterized entropy-based approach for network anomaly detection is promising, what is confirmed by Tellenbach [21], who employed Tsallis entropy in his Traffic Entropy Telescope prototype capable to detect a broad spectrum of anomalies, Yang [23], who applied Renyi entropy to early detection of low-rate DDoS attacks detection, and Kopylova [15], who reported positive results of using Renyi conditional entropy in detection of selected fast spreading or aggressive worms. There are some limitations of entropy based detection especially when it comes to detecting small or slow attacks. This is especially true for Shannon entropy which has a limited descriptive capability [21]. Apart from entropy, some other feature distributions summarization techniques are successfully used in the context of network anomaly detection, namely sketches [10] and histograms [14]. As the main disadvantage of this methods is the proper tuning, we decided not to include them in this work.

3 Entropy

In this section, we present some not commonly known theory regarding entropies used in our experiments. Definition of entropy as a measure of disorder comes from thermodynamic and was proposed in the early 1850s by Clausius. In 1948 Shannon adopted entropy to information theory. In information theory, entropy is a measure of the uncertainty associated with a random variable. The more random the variable, the bigger the entropy and in contrast, the greater certainty of the variable, the smaller the entropy. For a probability distribution $p(X = x_i)$ of a discrete random variable X, the Shannon entropy is defined as:

$$H_s(X) = \sum_{i=1}^{n} p(x_i) \log_a \frac{1}{p(x_i)} \tag{1}$$

X is the feature that can take values $\{x_1, \ldots, x_n\}$ and $p(x_i)$ is the probability mass function of outcome x_i. Depending on the base of the logarithm, different units can be used: bits ($a = 2$), nats ($a = e$) or hurtleys ($a = 10$). For the purpose of anomaly detection, sampled probabilities estimated from a number of occurrences of x_i in a time window t are typically used. The value of entropy depends on randomness (it

attains maximum when probability $p(x_i)$ for every x_i is equal) but also on the value of n. In order to measure randomness only, normalized forms have to be employed. For example, an entropy value can be divided by n or by maximum entropy defined as $\log_a(n)$. If not only the degree of uncertainty is important but also the extent of changes between assumed and observed distributions, denoted as q and p respectively, a relative entropy, also known as the Kullback-Leibler divergence can be used:

$$D_{KL}(p\|q) = \sum_{i=1}^{n} p(i) \log_a \frac{p(i)}{q(i)} \tag{2}$$

To measure how much uncertainty is eliminated in X by observing Y the conditional entropy may be employed:

$$H_S(X|Y) = \sum_{i=1,j=1}^{m,n} p(x_i, y_j) \log_a p(x_i|y_j) \tag{3}$$

The Shannon entropy assumes a tradeoff between contributions from the main mass of the distribution and the tail. To control this tradeoff, two parameterized Shannon entropy generalizations were proposed, by Renyi (1970s) and Tsallis (late 1980s) respectively [18], [22]. If the parameter denoted as α has a positive value, it exposes the main mass (the concentration of events that occur often), if the value is negative – it refers to the tail (the dispersion caused by seldom events). Both parameterized entropies (Renyi and Tsallis) derive from the Kolmogorov-Nagumo generalization of an average:

$$\langle X \rangle_\phi = \phi^{-1} \left(\sum_{i=1}^{n} p(x_i) \phi(x_i) \right), \tag{4}$$

where ϕ is a function which satisfies the postulate of additivity (only affine or exponential functions satisfy this) and ϕ^{-1} is the inverse function. Renyi proposed the following function ϕ:

$$\phi(x_i) = 2^{(1-\alpha)x_i} \tag{5}$$

After transformations, Renyi entropy may be given in the following form:

$$H_{R\alpha}(X) = \frac{1}{1-\alpha} \log_a (\sum_{i=1}^{n} p(x_i)^\alpha) \tag{6}$$

Tsallis extended the Renyi entropy with the following function ϕ:

$$\phi(x_i) = \frac{2^{(1-\alpha)x_i} - 1}{1-\alpha} \tag{7}$$

After transformations, the Tsallis entropy will be given by:

$$H_{T\alpha}(X) = \frac{1}{1-\alpha} \left(\sum_{i=1}^{n} p(x_i)^\alpha - 1 \right) \tag{8}$$

Both parameterized (Renyi and Tsallis) entropies:

– expose concentration for $\alpha > 1$ and dispersion for $\alpha < 1$,
– converge to the Shannon entropy for $\alpha \to 1$,
– correspond to cardinality of X for $\alpha = 0$.

4 Flow monitoring

There are two approaches to network traffic monitoring, namely, packet-based and flow-based. In our work we focus on flow-based network monitoring since it is more scalable in the context of network speed. This approach is based on the ability of network devices to aggregate packets in flows. Each flow is cached by device and when it is finished or a timeout is exceed it is exported to an element called collector. Modern approach assumes the use of dedicated probes transparently connected as a passive appliance via span ports or network taps rather than the usage of routers to export flows. This approach (presented in Fig. 1) can overcome some performance limitations of routers.

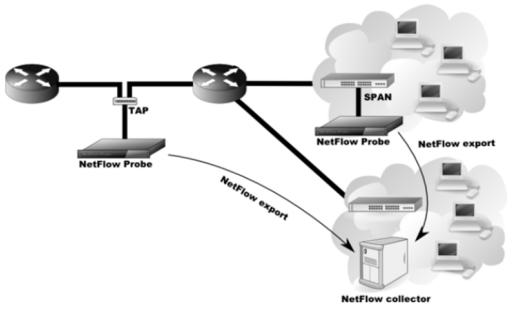

Fig. 1. Modern approach to flow exporting

The concept of network flows was introduced by Cisco and is currently standardized by the Internet Engineering Task Force (IETF). According to the IETF IPFIX working group [1], *"A flow is defined as a set of IP packets passing an observation point in the network during a certain time interval. All packets belonging to a particular flow have a set of common properties"*. In the simplest form, these properties are source and destination addresses and ports. A flow is typically defined as a unidirectional sequence of packets, which means that there are two flows for each connection between two endpoints – one from the server to client and one from the client to server. Recently, bidirectional flows (one record for each session between two endpoints) are also supported by vendors.

5 Data set

For the purpose of this work, we created the data set containing labeled flows. Firstly, we captured two-day (Tuesday, Wednesday) legitimate traffic from a medium size corporation network connected to the Internet. This was accomplished using open source software – *softflowd* and *nfsen*. The profile of this traffic is depicted in Fig. 2.

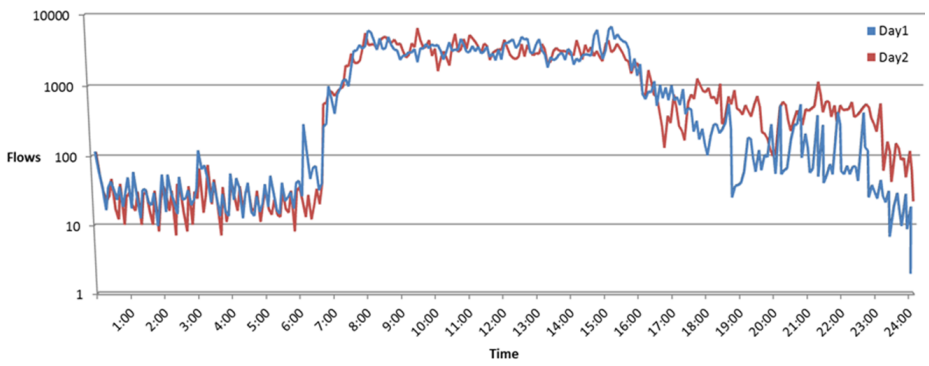

Fig. 2. Legitimate traffic profile by number of flows

We can see time t on x axis (5 minute fixed time window) and the number of flows on y (log scale) axis. Working day starts around 7 am. and finishes around 4 pm. The volume of traffic (expressed in number of flows) for both days is similar, but looking at the number of packets (Fig. 3), this similarity is a bit lower.

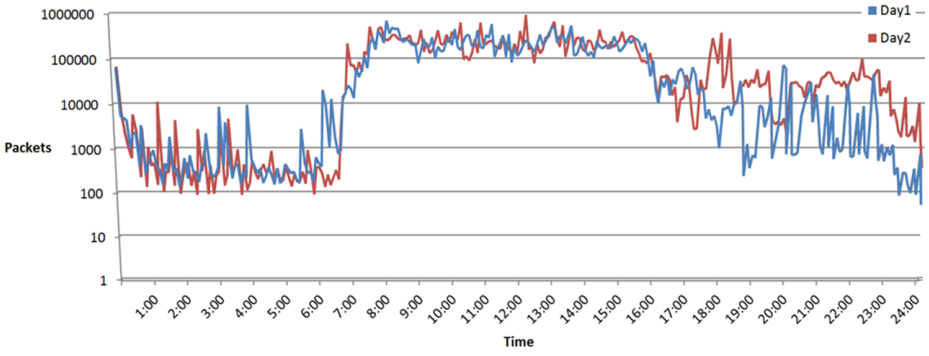

Fig. 3. Legitimate traffic profile by number of packets

In the next step, we generated *brute force*, *port scan*, *network scan* and *ddos* anomalies in different variants. More details concerning anomaly generation process is presented in the next section. Main characteristics of generated anomalies are presented in Table 1.

In the last step we mixed generated anomalies with the legitimate traffic from day2 (Wednesday) in the way presented in Fig. 4. We did not inject anomalies into the traffic from day1 (Tuesday) as it is used to build the profile of legitimate traffic in our approach.

As one can see, each anomaly is injected every 15 minutes mainly during working time. After injection only a few anomalies are visible in the volume expressed by number of flows or number of packets as depicted respectively in Fig. 5 and Fig. 6.

Table 1. Characteristics of generated anomalies

Type/kind	No. of flows	Duration [sec]	No. of victims	No. of attackers
SSH brute force (bf)				
1	1K	300	1	1
2	1K	100	1	1
3	2K	300	1	1
DDoS (dd)				
1	2K	200	1	50
2	2K	200	1	250
3	3K	300	1	50
4	3K	300	1	250
5	4K	400	1	50
6	4K	400	1	250
Network scan (ns)				
1	6K	60	6K	1
2	6K	300	6K	1
3	8K	80	8K	1
4	8K	400	8K	1
Port scan (ps)				
1	1K	50	1	1
2	1K	100	1	1
3	2K	100	1	1
4	2K	200	1	1

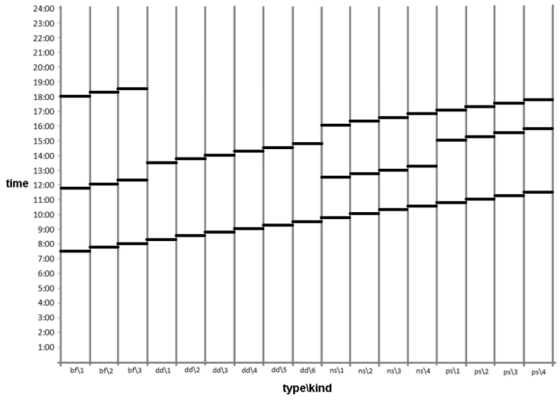

Fig. 4. Distribution of anomalies in time

6 Anomaly generator

In order to produce flows that can mimic an anomalous behavior dedicated tool in Python language was developed. With this tool we can generate flows according to the predefined policy. The policy assigns a certain type of generation method to each field of flow record. In result we obtain a set of flows which meets given statistical profile.

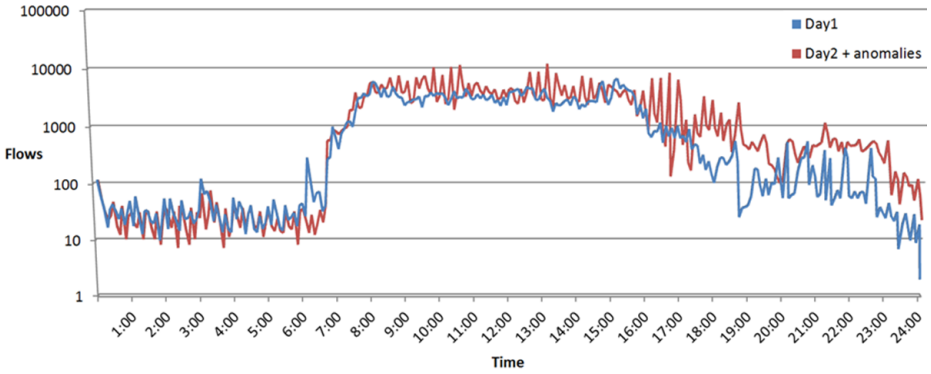

Fig. 5. Legitimate and anomalous traffic by number of flows

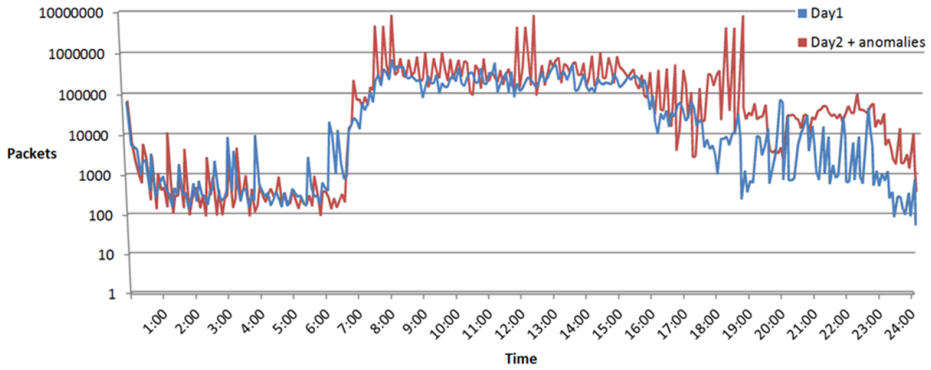

Fig. 6. Legitimate and anomalous traffic by number of packets

Internally, our tool operates on integer values which are manipulated by generation methods (introduced in [5]). They are as follows: *con* (constant), *ran* (random) and *per* (periodical). *Con* generator is straightforward and does not need further explanation, others are described below. *Ran* generators are used to obtain random values. There are two types of such generators: absolute (e.g. *srcPort* in Listing 1.1) or relative (e.g. *dstIP* in Listing 1.1). The value produced with the relative generator is summed with previously generated one. This feature can be used to sweep across certain range of values. Both generators can be initiated with either uniform or arbitrary distribution. Arbitrary distribution consists of two list: values and probabilities of these values. Relative generator additionally needs a start value and a range. *Per* generators are used to match a certain generating method with the sequence number of the currently generated flow. They are initiated with a list of key-value pairs out of which the first one represents the flow number and the second – the generator definition. On the last position, the default generator is placed. For example, *iar* definition in Listing 1.1 means that every 300th flow a uniform(10,50) generator will be applied and respectively every 800th flow generator returns 5000. In other cases, default generator will be applied. The set

Listing 1.1. Default generator group

```
[testgroup]
protocol = con[TCP]
srcIP = con[10.5.0.77]
dstIP = ran[10.1.0.1; (["0.0.0.1", "0.0.0.2", "-0.0.0.1"],
[0.97,0.15,0.15]); (10.1.0.1, 10.1.0.253)]
srcPort = ran[uniform(300, 500)]
dstPort = con[22]
fromSrcPkts = con[1]
fromSrcOctets = con[60]
fromDstPkts = con[1]
fromDstOctets = con[60]
#duration
dur = con[1]
#inter arrival time
iar = per[300:ran[uniform(10, 50)]; 800:con[500];
ran[ ([10, 11, 12, 13], [0.20, 0.30, 0.40, 0.10])]]
flags = con[SYN|ACK|RST]
```

of generators shown on Listing 1.1 is called the generator group. A policy may consist of multiple groups. In such a case probability of using a certain generator group must be defined. Only one generator group (considered as default) in a policy has a generator for each field of the flow. The additional groups may override all or selected definitions of the default one. A concept of a generator group was introduced to ensure that fields of the flow will be consistent with each other. For example, to disallow flows which are too short when compared with the amount of bytes of the flow. There are phenomena on the network that can only be modeled with sequences of flows. Our tool provides such a functionality which is available through indexing of group names. In such indexed groups, one can use mechanisms which allow sharing state between subsequent flows. For example, in Listing 1.2, we enforced value of *dstIP* not to be changed through the whole sequence.

Listing 1.2. Sequence modelling

```
[testgroup.1]
dstIP = args[usePrevValue]
dur = con[100]
[testgroup.2]
dstIP = args[usePrevValue]
dur = con[1000]
```

An example of a similar generator is Flame [5]. However, there are some significant differences. Flame comes with very basic support for generating flows, forcing users to implement all the generation logic by themselves, while our tool supports policy files. On the other hand it has fairly sophisticated functionality of inserting generated flows

into the base traffic which our tool does not support at all. Another interesting concept was introduced in [19]. Authors proposed to describe network traffic (not only flows) by a set of so-called α- and β-profiles which can subsequently be used to generate a data set. α-profiles consist of actions which should be executed to generate a given event in the network (such as attack) while β-profiles are more similar to our policy files where behavior of certain entities (packet sizes, number of packets per flow) are represented by statistical model. On the whole this concept is similar to ours but far more complex.

7 Network anomaly detector

In this section we present entropy-based network anomaly detection module which is a component of the anomaly detection and security event data correlation system currently developed it the Secor project [8]. The goal of this module is to detect network anomalies with acceptable false positive error rate and high detection rate, classify anomalies and report some details (timestamps, related addresses and ports) to the correlation engine (output) which correlates events coming from different modules and external sensors in order to improve detection and limit false positive rate. The architecture of our network anomaly detection module is presented in Fig. 7.

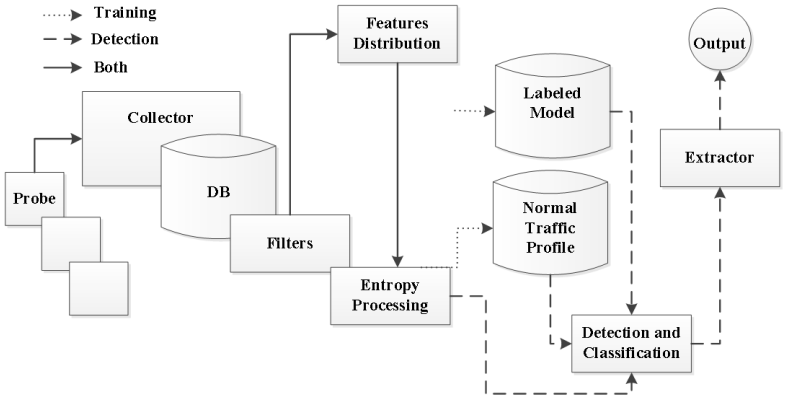

Fig. 7. Entropy-based network anomaly detection module

IP traffic is captured by NetFlow [1] probes. We decided to use bidirectional flows since, according to some works (e.g. [17]), unidirectional flows may entail biased results. In order to limit the area of search, filters per direction, protocol and subnet are used. Collected flows are analyzed within constant-length time intervals (every 5 min. by default). Next, depends on the version, Tsallis or Renyi entropy of positive and negative α values are calculated for traffic feature distributions presented in Table 2. Note: the Shannon version of our method use internally Renyi entropy with α set to 1.

Initially, during the training phase, a dynamic profile is built using $\min$ and $\max$ entropy values within a sliding time window for every $\langle feature, \alpha \rangle$ pair. Thus, we can

Table 2. Selected traffic feature distributions

Feature	Probability mass function
src(dst)address(port)	$\frac{number\ of\ x_i\ as\ src(dst)address(port)}{total\ number\ of\ src(dst)addresses(ports)}$
flows duration	$\frac{number\ of\ flows\ with\ x_i\ as\ duration}{total\ number\ of\ flows}$
packets, bytes	$\frac{number\ of\ pkts(bytes)\ with\ x_i\ as\ src(dst)\ addr(port)}{total\ number\ of\ pkts(bytes)}$
in(out)-degree	$\frac{number\ of\ hosts\ with\ x_i\ as\ in(out)-degree}{total\ number\ of\ hosts}$

reflect traffic changes during the day. In the detection phase, the observed entropy is compared with the $\min$ and $\max$ values stored in the profile according to the following rule:

$$r_\alpha(x_i) = \frac{H_\alpha(x_i) - k * \min_\alpha}{k * (\max_a - \min_\alpha)}, \qquad k \in \langle 1..2 \rangle \tag{9}$$

With this rule, anomaly threshold is defined. Values $r_\alpha(x_i) < 0$ or $r_\alpha(x_i) > 1$ indicate abnormal concentration or dispersion respectively. This abnormal dispersion or concentration for different feature distributions is characteristic for anomalies. For example, during a port scan, a high dispersion in port numbers and high concentration in addresses should be observed. Detection is based on the relative value of entropy with respect to the distance between $\min$ and $\max$. Coefficient k in the formula determines a margin for $\min$ and $\max$ boundaries and may be used for tuning purposes. A high value of k, e.g. $k = 2$, limits the number of false positives while a low value ($k = 1$) increases detection rate. We also take into consideration other approaches to thresholding based on standard deviation and quantiles. The detection is based on the results from all feature distributions. Classification is based on classifiers (decision trees, Bayes nets [20], rules and functions) employed in Weka software [3]. Extraction of anomaly details is also assumed – related ports and addresses are obtained by looking into the top contributors to the entropy value.

8 Results

Experiments were performed for Tsallis, Renyi and Shannon version of our method as well as traditional volume-based approach with flow, packet and byte counters. Final evaluation was performed with Weka tool. Some examplary results of entropies for a selected singular feature distributions are presented below. Abnormally high dispersion in destination addresses distribution for network scan anomalies exposed by negative value of α parameters is depicted in Fig. 8. One can see time t on x axis (5 minute time windows), result r on y axis and α values on z axis. Anomalies are marked with (A) on the time axis. Values of Shannon entropy are denoted as S.

Abnormal concentration of flows duration for network scans is depicted in Fig. 9. This concentration is typical for anomalies with fixed data stream.

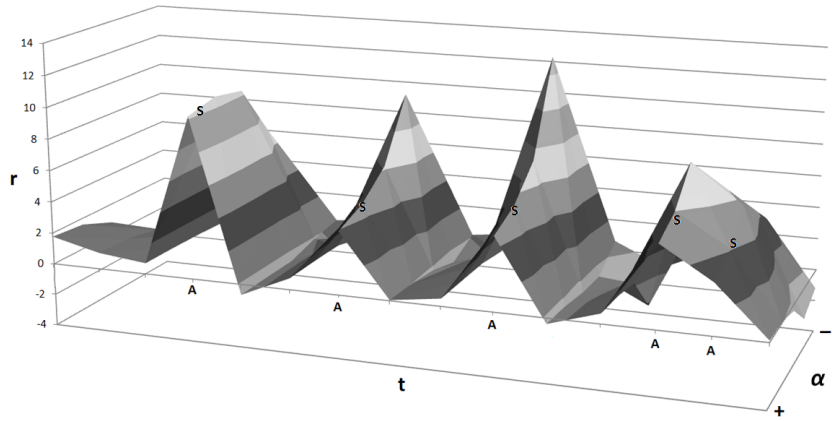

Fig. 8. Abnormally high dispersion in destination addresses for network scan anomalies (Renyi/Shannon)

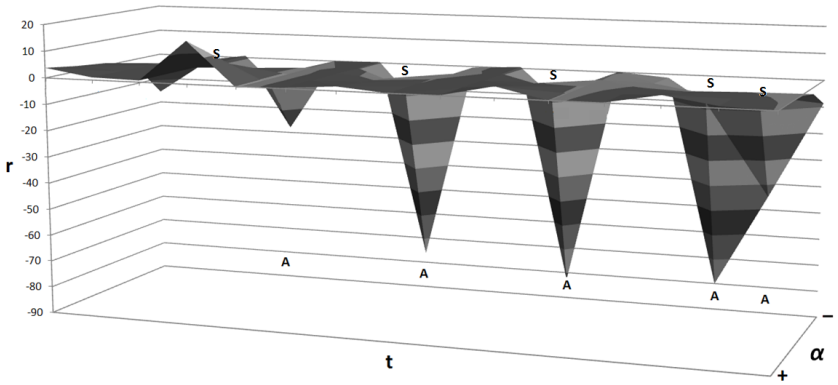

Fig. 9. Abnormally high concentration in flows duration for network scan anomalies (Tsallis/Shannon)

Fig. 10 shows ambiguous detection (no significant excess of $0-1$ threshold) of port scan anomaly with volume-based approach with flow, packet and byte counters. R on y axis corresponds to normalization applied in our method [equation (9)].

We noticed that measurements for all feature distributions as a group work better than single ones or subsets. The best results were obtained for addresses, ports and duration feature distributions, although we believe that the proper set of feature distributions is specific for particular anomalies.

Overall (whole data set, all feature distributions) multi-class classification was performed with Weka tool. We defined 4 classes for each anomaly type + 1 class for legitimate traffic. To properly asses predictive performance ten-fold cross-validation method was used. An ideal classifier should not produce false positive and false negative statistical errors. To evaluate non-ideal classifiers, one could measure proportion of correct

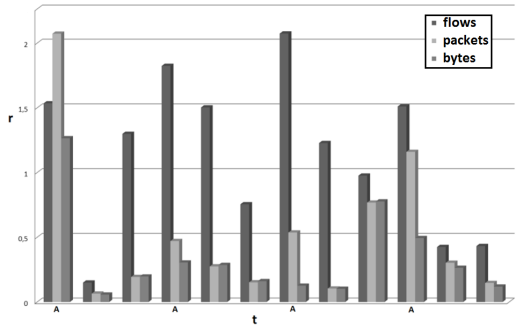

Fig. 10. Ambiguous detection of port scan anomaly with a volume-based approach

assessments to all assessments (Accuracy), the share of benign activities reported as anomalous (False Positive Rate) and the share of anomalies missed by the detector (False Negative Rate). Usage of Precision (proportion of correctly reported anomalies) and Recall (share of correctly reported anomalies compared to the total number of anomalies) is another option. Based on these measures some tools like ROC (Receiver Operating Characteristics) and PR (Precision vs Recall) are typically used [9]. Averaged performance of classification for different classifiers is presented in Table 3.

ZeroR is a trivial classifier which classifies the whole traffic as normal. We included it here as a reference to other results. We believe that an accuracy is not the correct

Table 3. Averaged performance of classification

	ZeroR|	Bayes Netw.|	Decision Tree J48|	Rand. Forest|	Simple Logistic Regress.
Accuracy					
Tsallis	0.66	0.89	0.90	0.92	0.93
Renyi	0.66	0.89	0.89	0.90	0.93
Shannon	0.66	0.84	0.86	0.90	0.92
Volume-based	0.66	0.71	0.77	0.78	0.80
ROC area					
Tsallis	0.44	0.97	0.94	0.99	0.98
Renyi	0.44	0.96	0.91	0.97	0.97
Shannon	0.44	0.95	0.88	0.97	0.98
Volume-based	0.44	0.80	0. 81	0.90	0.94
PR area					
Tsallis	0.45	0.96	0.90	0.97	0.96
Renyi	0.45	0.94	0.85	0.93	0.94
Shannon	0.45	0.93	0.83	0.94	0.96
Volume-based	0.45	0.75	0.72	0.79	0.85

choice to measure the performance of classification if data set is unbalanced – more normal than anomalous in our case. We suggest to look at ROC area and PR area instead. From the whole spectrum of tested methods the best performance was obtained by applying Logistic Regression, Bayesian Network, Decision Tree and Random Forest classifiers.

9 Conclusions and future work

Concluding the results of our studies, we can observe that, for our dataset: i) the Tsallis entropy performed best; ii) the Renyi entropy was slightly weaker; iii) the Shannon entropy was a bit worse than the Renyi (except from the Random Forest classifier); iv) the volume-based method performed poorly; v) among a large set of network traffic feature distributions, addresses, ports, and flows durations proved to be the best choices; vi) the most successful classifiers were Linear Regression, Bayes Network, Decision Tree and Random Forest. In general we believe that a broad spectrum of feature distributions provides a better flexibility to detect different types of anomalies.

While we admit that our experiments were limited to few number of cases, we also believe that these cases were representative. Our data set contains traces of network malicious activities which are typical for worm propagation, communication and attacks performed by group of machines infected by worms. Although, only one day legitimate traffic profile was built in our experiments, we have observed that this profile suits to each regular working day in the network we monitored. While more research work is necessary to validate the efficiency of the parameterized entropies, the poor performance of the Shannon entropy and volume-based methods allows to question whether they are the right approach to anomaly detection. In our method the precise traffic profile is the key, so future work will include optimization and experiments with more fluctuative legitimate traffic. We are also planning to model new anomalies and inject them into our data set to perform evaluation on a larger scale. We hope to retain good performance. Ś

Acknowledgements

This work has been partially supported by the National Centre for Research and Development project no. PBS1/A3/14/2012 "Sensor data correlation module for detection of unauthorized actions and support of decision process" and the European Regional Development Fund the Innovative Economy Operational Programme, under the INSIGMA project no. 01.01.02-00-062/09.

References

1. IETF IPFIX Working Group. http://datatracker.ietf.org/wg/ipfix/charter
2. Verizon. 2014 Data Breach Investigations Report. http://www.verizonenterprise.com/DBIR/2014/
3. Weka project homepage. http://www.cs.waikato.ac.nz/ml/weka

4. Bereziński, P., Pawelec, J., Małowidzki, M., Piotrowski, R.: Entropy-based internet traffic anomaly detection: A case study. In: Proceedings of the Ninth International Conference on Dependability and Complex Systems DepCoS-RELCOMEX, *Advances in Intelligent Systems and Computing*, vol. 286, pp. 47–58. Springer (2014)
5. Brauckhoff, D. (ed.): Network traffic anomaly detection and evaluation. ETH, Zurich (2010)
6. Brauckhoff, D., Tellenbach, B., Wagner, A., May, M., Lakhina, A.: Impact of packet sampling on anomaly detection metrics. In: Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement, IMC '06, pp. 159–164. ACM (2006)
7. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: A survey. ACM Computing Surveys **41**(3), 15:1–15:58 (2009)
8. Chorań, M., Kozik, R., Piotrowski, R., Brzostek, J., Hołubowicz, W.: Network events correlation for federated networks protection system. In: Towards a Service-Based Internet, *Lecture Notes in Computer Science*, vol. 6994, pp. 100–111. Springer (2011)
9. Davis, J., Goadrich, M.: The relationship between precision-recall and roc curves. In: Proc. of the 23rd Int. Conference on Machine Learning, ICML'06, pp. 233–240. ACM (2006)
10. Dimitropoulos, X., Stoecklin, M., Hurley, P., Kind, A.: The eternal sunshine of the sketch data structure. Computer Networks **52**(17), 3248–3257 (2008)
11. Fillatre, L., Nikiforov, I., Casas, P., Vaton, S.: Optimal volume anomaly detection in network traffic flows. In: Proceedings of the 16th European Signal Processing Conference, EURASIPCO '08. EURASIP (2008)
12. Jasiul, B., Śliwa, J., Gleba, K., Szpyrka, M.: Identification of malware activities with rules. In: Proceedings of the Federated Conference on Computer Science and Information Systems. Warsaw, Poland (2014)
13. Jasiul, B., Szpyrka, M., Śliwa, J.: Malware behavior modeling with Colored Petri nets. In: Computer Information Systems and Industrial Management Proceedings of the 13th IFIP TC8 International Conference CISIM 2014, LNCS. Springer-Verlag (2014)
14. Kind, A., Stoecklin, M.P., Dimitropoulos, X.: Histogram-based traffic anomaly detection. IEEE Trans. on Netw. and Serv. Manag. **6**(2), 110–121 (2009)
15. Kopylova, Y., Buell, D., Huang, C.T., Janies, J.: Mutual information applied to anomaly detection. Journal of Communications and Networks **10**(1), 89–97 (2008)
16. Lakhina, A., Crovella, M., Diot, C.: Mining anomalies using traffic feature distributions. In: Proceedings of the 2005 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, SIGCOMM '05, pp. 217–228. ACM (2005)
17. Nychis, G., Sekar, V., Andersen, D.G., Kim, H., Zhang, H.: An empirical evaluation of entropy-based traffic anomaly detection. In: Proceedings of the 8th ACM SIGCOMM Conference on Internet Measurement, IMC '08, pp. 151–156. ACM (2008)
18. Renyi, A.: Probability Theory. Dover Books on Mathematics Series. Dover Publ. Inc. (1973)
19. Shiravi, A., Shiravi, H., Tavallaee, M., Ghorbani, A.: Toward developing a systematic approach to generate benchmark datasets for intrusion detection. Computers and Security **31**(3), 357 – 374 (2012)
20. Szpyrka, M., Jasiul, B., Wrona, K., Dziedzic, F.: Telecommunications networks risk assessment with Bayesian networks. In: Computer Information Systems and Industrial Management Proceedings of the 12th IFIP TC8 International Conference CISIM 2013, *Lecture Notes in Computer Science*, vol. 8104, pp. 277–288. Springer-Verlag (2013)
21. Tellenbach, B., Burkhart, M., Schatzmann, D., Gugelmann, D., Sornette, D.: Accurate network anomaly classification with generalized entropy metrics. Computer Networks **55**(15), 3485–3502 (2011)
22. Tsallis, C., de Pesquisas Físicas, C.B.: Possible Generalization of Boltzmann-Gibbs Statistics. Notas de física. Centro Brasileiro de Pesquisas Físicas (1987)
23. Xiang, Y., Li, K., Zhou, W.: Low-rate ddos attacks detection and traceback by using new information metrics. Trans. Info. For. Sec. **6**(2), 426–437 (2011)