Chapter 32
A Survey and Comparison of Performance Evaluation in Intrusion Detection Systems

Jason Ernst, Tarfa Hamed, and Stefan Kremer

32.1 Introduction

Intrusion detection systems (IDS) are a rapidly changing field—out of necessity since security is often a cat-and-mouse game. The detection systems must constantly evolve in order to keep up with new attacks and vulnerabilities. Due to the rapidly changing nature of attacks and the IDSs created to detect these attacks, there is not much standardization in the field. While there have been some attempts in this direction, it is typically focused on standard datasets which catalog known attacks. Increasingly, to combat the rapidly changing nature of attacks and vulnerabilities, IDS are turning towards machine learning which provides some flexibility in detecting novel attacks. With the overview and comparison of the different IDSs provided in this chapter, the reader may then compare how new and existing systems compare with each other in the context of these previously known attacks, and their ability to adapt to new threats. The aim of this chapter is to collect and present a common framework for comparison of IDSs wherever possible, and also to highlight systems which are specialized towards specific attacks, and those systems which provide widespread and general coverage. In addition, where possible (based on data provided in the surveyed approaches), we aim to present comparison metrics so that the reader may try to gauge which IDS approaches may be the most effective in particular circumstances. First, we survey popular standardized datasets, and metrics which are commonly used to evaluate and compare IDSs and then we provide an in-depth survey and overview of particular IDSs.

J. Ernst (✉) • T. Hamed • S. Kremer
University of Guelph, Guelph, ON, N1G 2W1, Canada
e-mail: jason@left.io; tyaseen@uoguelph.ca; skremer@uoguelph.ca

K. Daimi (ed.), *Computer and Network Security Essentials*,
DOI 10.1007/978-3-319-58424-9_32

32.2 Standardized Datasets for Benchmarking IDSs

One popular dataset is the KDD-99 dataset [1] which “contains a standard set of data to be audited, which includes a wide variety of intrusions simulated in a military network environment”. The KDD-99 dataset covers four types of attacks: DOS (Denial of Service), R2L (Remote to Local), U2R (User to Root) and Probe [2]. Another well-known dataset for benchmarking IDS is the Lincoln Labs dataset from 1999 [3]. It also covers four types of attacks which can be classified into the same categories as the KDD-99 dataset. While both of these datasets are over 15 years old, they are still used in recent papers, particularly because they are so commonly used. However, modern intrusions require updated datasets including attacks such as those from botnets which recently caused significant security issues with the Internet [4]. A more modern dataset called CTU-13 [5] has been recognized by some researchers as the most valuable botnet dataset [6]. There are also many active discussions about which datasets should be used to replace KDD-99 and Lincoln Labs datasets, as well as suggestions for datasets for particular types of attacks, different training techniques and other situations by researchers [7, 8].

32.3 Metrics Used to Evaluate and Compare IDSs

In the majority of the papers surveyed, there are few common metrics which are used to evaluate the performance across all of the approaches. Most of the performance metrics are illustrated in Fig. 32.1. Typically the most common are [9]:

1. *False positive rate* which is the percentage of traffic identified as anomalies or threats but are actually legitimate traffic.
2. *False negative rate* which is the percentage of traffic which is actually anomalies or threats but are missed by the system.

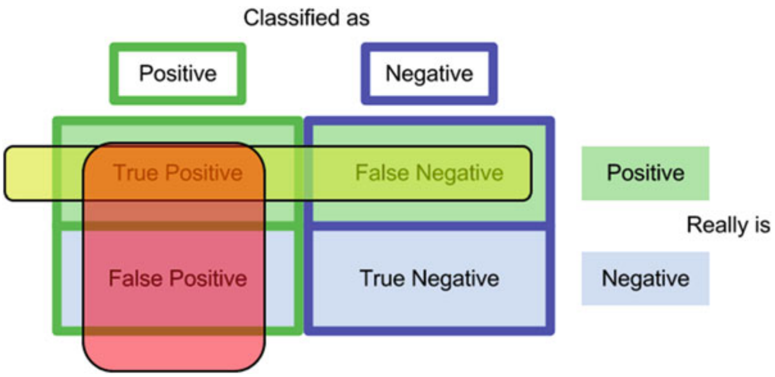

Fig. 32.1 Some common metrics illustrated (*precision red*, recall *yellow*) [8]

3. *True positive (or attack detection) rate* which is the percentage of actual attacks or anomalies which are correctly identified by the system.
4. *True negative rate* which is the percentage of traffic which is correctly identified as not an anomaly or threat.
5. *Precision*: is the number of true positives divided by the total of true positives and false positives.
6. *Sensitivity/recall*: is the number of true positives divided by the total of true positives and false negatives.
7. *Processing speed/capacity* which is the amount of time it takes an IDS to classify all of the attacks in a dataset.
8. *Training speed* which is the amount of time a system must be trained on a dataset to reach the effectiveness reported in the results.
9. *System utilization*: the amount of memory and CPU required to run the IDS [10]

32.4 Specification Method

The specification-based method is one where "manually specified program behavioral specifications are used as a basis to detect attack" [11]. Misuse detection, in contrast, is unable to detect attacks which have not been seen before, and anomaly detection often has high false positive rates. Specification methods instead focus on characterizing legitimate program or network behaviour. The biggest drawback would be every time new behaviour is required, it would need to be included into the specification method so as to not result in a false positive. For the specification-based method which was presented in [12], the experiment was directed to lower layer protocols (TCP and IP) which correspond to Denial of Service (DoS) and probing attacks in the Lincoln Labs data. The interesting points of the experiment were [12]:

1. A high rate of attack detection was achieved: The system proved its ability to detect all the attacks (100% rate of attack detection) that were supposed to be detected according to the prototype.
2. The system was able to detect IP-sweeps also which many anomaly detection systems fail to detect.
3. A low rate of false positives was achieved: The system proved to produce a very low false positive rate (about 5.5 false alarms/day).
4. The system exhibited an acceptable processing capacity: The system proved fast enough to be able to process an entire day's data within only 10 min.

For the IP machine, Table 32.1 explains the details of the detected attacks [12]. The experiment was performed on 43 attacks varied in type from DoS attacks, probing attacks and eavesdropping attacks. As shown from Table 32.1, the system was able to detect the whole number of given attacks.

Table 32.1 Attacks detected in 1999 Lincoln labs IDS evaluation data [12]

Attack name	Attacks present	Attacks detected	Description	Threat category
Apache2	2	2	DoS attack on Apache web server	DoS/comm.
Back	3	3	DoS attack on Apache web server	DoS/comm.
IP Sweep	6	6	Probe to identify potential victims	Access/IP sweep
Mailbomb	3	3	Large volume of mail to a server	DoS/comm.
Mscan	1	1	Attack tool	DoS/comm.
Neptune	3	3	SYN-flood attack	DoS/comm.
Ping-of-death	4	4	Over-sized ping packets	DoS/system
Smurf	3	3	ICMP echo-reply flood	DoS/comm.
Queso	3	3	Stealthy probe to identify victim OS	Access/ eavesdropping
Satan	2	2	Attack tool	Access/ eavesdropping
Portsweep	13	13	Probing to identify exploitable servers	DoS/comm.
Total	43	43		

32.5 Support Vector Machines

As opposed to manually coding behaviours, misuse or attacks, it is also possible to train systems to detect improper behaviour. Typically, Support Vector Machines (SVM)s, which are a type of machine learning, require a training stage along with labelled data which indicates normal traffic or attack traffic. The approach taken in [13] is a hybrid SVM which combines a trained SVM with a portion which can detect novel attacks so that it is flexible and has potentially better performance than completely untrained SVMs. The empirical evaluation phase of [13] involved three steps:

1. To verify the efficiency of the packet filtering approach, labelled datasets were used. A normal dataset consisted of collected normal packets during certain agreed times for the experiment. More than one dataset is used, the abnormal dataset #1 represents attack packets using TCP/IP covert channels, abnormal #2 represents DoS attacks and abnormal #3 contains malformed TCP/IP packets that disobey the published TCP/IP standard. The details of packet filtering using TCP/IP are given in Table 32.2, where the last column in the table represents the ratio of the number of filtered packets to the total number of packets [13].
2. Classification was performed using a soft-margin SVM, which showed high performance as expected, since the soft-margin SVM is an established, well-developed, time-proven supervised learning method. In spite of the above, it is

Table 32.2 Packet filtering results using passive TCP/IP finger printing [13]

	No. of packets	Inbound filtering	Outbound filtering	No. of filtered packets	Ratio (%)
Normal #1	1,369,134	1,249,924	39,804	1,289,728	94.20
Normal #2	1,574,274	1,435,838	46,374	1,482,212	94.20
Normal #3	1,571,598	1,439,880	44,071	1,483,951	94.40
Normal #4	1,783,444	1,589,337	65,035	1,654,372	92.80
Total	6,298,450	5,714,979	195,284	5,910,263	Ave: 93.84 STD: 0.74

not suitable for detecting novel attacks because of its supervised nature. The one-class SVM used an RBF kernel which showed excellent performance (94.65%), but the false positive rate was as high as well [13]. Since the enhanced SVM employs two kinds of machine learning features, it showed that it is very sensitive to feature mapping using a kernel function, since it incorporates two types of machine learning features. The enhanced SVM experiment with a sigmoid kernel outperformed in two areas: it produced similar detection performance as the soft-margin SVM, and it produced lower false positive and negative rates than the previous methods.

3. The third stage focused on comparing results with a real NIDS: In order to confirm the efficiency of the proposed approach, a comparison with a real NIDS was shown. Some well-known NIDSs (like Snort and Bro) are used for comparison. Those NIDSs are de facto standards used in the industry. The proposed framework is compared using the enhanced SVM method with NIDS systems based on real data.

Four kinds of test sets were used in each test experiment as follows:

1. Test #1 was an attack-free first source dataset not used during training
2. Test #2 contained ten kinds of attacks found in the data captured by the second source
3. Test #3 contained nine kinds of attack data from the DARPA IDS Test
4. Test #4 was comprised of real data collected by the second source.

The detailed results of the proposed framework with comparison to Snort and Bro are given in Table 32.3.

32.6 Machine Learning

In addition to SVMs there are also many other machine learning approaches. Several of the approaches based on classifier decision trees and decision rules are evaluated in [14]. The empirical evaluation was performed by comparing the performance of the several classification algorithms. The comparison involved calculating True

Table 32.3 Real-world test results [13]

		Test sets	Detection rate (%)	False positive rate (%)	False negative rate (%)
Enhanced SVM	TR1	Test#1—Normal	92.40	7.60	–
		Test#2—Attacks	68.70	–	31.30
		Test#3—Attacks	74.47	–	25.53
		Test#4—Real	99.80	0.20	–
	TR2	Test#1—Normal	94.53	5.46	
		Test#2—Attacks	66.90		33.10
		Test#3—Attacks	65.60		34.40
		Test#4—Real	99.93	0.07	
	TR3	Test#1—Normal	95.63	4.37	
		Test#2—Attacks	64.20		35.8
		Test#3—Attacks	57.20		42.80
		Test#4—Real	99.99	0.01	–
	Analysis	Test#1—Normal	Ave: 94.19, SD: 1.64	Ave: 5.81, SD: 1.64	Ave: 0.00, SD: 0.00
		Test#2—Attacks	Ave: 66.60, SD: 1.26	Ave: 0.00, SD: 0.00	Ave: 33.40, SD: 2.26
		Test#3—Attacks	Ave: 65.76, SD: 8.64	Ave: 0.00, SD: 0.00	Ave: 34.24, SD: 8.63
		Test#4—Real	Ave: 99.90, SD: 0.10	Ave: 0.09, SD: 0.09	Ave: 0.00, SD: 0.00
Snort		Test#1—Normal	94.77	5.23	–
		Test#2—Attacks	80.00	–	20.00
		Test#3—Attacks	88.88	–	11.12
		Test#4—Real	93.62	6.38	–
Bro		Test#1—Normal	96.56	3.44	–
		Test#2—Attacks	70.00	–	30.00
		Test#3—Attacks	77.78		22.22
		Test#4—Real	97.29	2.71	–

While TR1, TR2, TR3 mean training set 1,2 and 3, respectively

Positive (TP), False Positive (FP) and Prediction Accuracy (PA). An evaluation was made for these criteria to determine the best algorithm for the given attack. Table 32.4 shows the evaluation criteria for decision rules used in [14]. For the first column, CCI means Correctly Classified Instances and ICCI means Incorrectly Classified Instances [14]. This evaluation revealed that different algorithms that should be used to process different types of network attacks since some algorithms have an outstanding detection performance on specific types of attacks, compared with others [14].

Table 32.4 Evaluation criteria for decision rules [14]

Evaluation criteria	Classifier rules			
	JRip	Decision table	PART	OneR
CCI	63,810	63,250	63,979	56,909
ICCI	1724	2284	1555	8625
PA (%)	97.36	96.51	97.62	86.83

32.7 Behaviour-Based Approaches

Behaviour-based approaches focus on trying to capture, represent and encode the behaviour of malware, or an attack in a standardized format. An example of such a system is Malware Instruction Set (MIST) [15]. The biggest benefit of focusing on representing the behaviour is that it works well with other classification approaches. Good representation can enhance classification. Furthermore, due to the standardized and efficient encoding, it is possible to reduce the size of the reports generated on systems running IDSs. For behaviour-based analysis which was presented in [15], the empirical evaluation phase involved using 500 malware binaries. Then, the system used six independent antivirus software applications to assign the samples to five different classes. The assigned malwares were then executed under control of CWSandbox to produce 500 behaviour reports. That resulting distance matrix for the MIST showed high classification accuracy by the dark colour of the matrix diagonal (the darker the better). The results showed that MIST performed better than the XML representation especially in classifying Allaple and Looper malwares. In addition, the size of the resulting MIST report was very small as compared with XML, which leads to a reduction of analysis run-time as a result [15].

32.8 Mobile Agents

While many of the previous approaches apply on a single node, or even in a centralized manner, mobile agents allow the network to co-operatively determine if the network is being attacked. Since the task is divided, one of the benefits is often decreased load in terms of CPU and memory, however, it often comes at the cost of increased overhead in terms of network communications in order to coordinate with each other. The empirical evaluation phase of using a mobile agent with intrusion detection was a little bit different. In [10], the empirical evaluation phase consisted of studying the system resource utilization and capturing data statistics. For the resource utilization, results showed that the IDS has very low occupancy of the system resources (memory and CPU). For capturing data statistics, results showed that the IDS is very sensitive so that it can detect the “ping command”—which is a command used for checking a network connection—from the sharp change of the

ICMP package [10]. In [16], which proposes an intrusion detection model of mobile agent based on Aglets (IDMAA) for detecting intrusions, the empirical evaluation can be illustrated via two properties [16]:

1. Strength of IDMAA: The system consisted of some detection agents, and these agents were independent from each other, so that if one agent was stopped, it will affect the detection rate on that node only. That confirms that the system is fault-tolerant and robust.
2. Dynamic adaptability: If a host generated an agent and sent them to another host, there is no need to restart the system. The same applies when a host disposes of an agent; that does not affect the other detection agents. IDMAA can increase or decrease the detection agents according to the need to enhance detection performance. That is the idea of dynamic adaptability.

32.9 Genetic Network Programming

Similar to the goals of the hybrid SVM approach in [13], the Genetic Network Programming (GNP) approach in [17] tries to balance the ability to detect novel attacks with having too high of a false positive rate. The empirical evaluation phase of the approach presented in [17]—which focuses on using GNP for intrusion detection—consisted of two parts: choosing the sample space and checking the detection rates. After selecting the training sample space in the data collecting phase, the testing sample space was specified by dividing the database into two parts: 748 unlabelled normal connections and 320 unlabelled intrusions (Neptune, smurf, portsweep and nmap) connections. For checking detection rates, the results are given in Table 32.5 [17].

In order to evaluate the testing results, some criteria were calculated such as: Positive False Rate (PFR), Negative False Rate (NFR) and accuracy. The PFR corresponds to the false positive rate, the NFR corresponds to false negative rate and the accuracy corresponds to detection rate, respectively. Those criteria were calculated for the above table as follows [17]:

PFR = (4+6)/748 = 1.34%

NFR = (0+24)/320 = 7.5%

Accuracy = (738+170+55)/1068 = 90.16%

Table 32.5 Simulation results of GNP intrusion detection [17]

	Normal (T)	Known intrusion (T)	Unknown intrusion (T)	Total (T)
Normal (R)	738	4	6	748
Known intrusion(R)	0	170	70	240
Unknown intrusion (R)	24	1	55	80
Total	762	175	131	1068

Table 32.6 Detection results for testing dataset of DOS, Probe, R2L and U2R with Clustering, C4.5 Rules, RIPPER and FILMID [18]

Attack class		Clustering	C4.5 Rules	RIPPER	FILMID
DoS	DR (%)	83.64	91.89	92.34	92.37
	FPR (%)	3.13	2.41	2.36	2.38
Probe	DR (%)	71.45	81.29	81.42	81.41
	FPR (%)	0.68	0.54	0.47	0.46
R2L	DR (%)	68.26	78.61	78.65	78.71
	FPR (%)	1.52	1.12	1.14	1.13
U2R	DR (%)	11.85	17.53	17.56	17.49
	FPR (%)	0.47	0.31	0.35	0.35

32.10 Fast Inductive Learning

"Given a series of positive examples and negative examples about one concept, the task of inductive learning is to induce a common concept description from these examples". In [18], which used fast inductive learning for intrusion detection (FILMD), the empirical evaluation phase began after training FILMD, RIPPER and C4.5 rules on the training datasets (in the detection phase), the system was tested on the specified dataset for the four attack classes. Next, some of the criteria were calculated for the three algorithms and clustering-based anomaly detection against all the attack classes. These criteria include: Detection Rate (DR) and False Positive Rate (FPR). The obtained results are shown in Table 32.6 [18]. The FILMD approach is similar to other machine learning approaches [13, 14], but the focus is on fast detection time while still maintaining high performance. The RIPPER and C4.5 Rules approaches are other competing inductive learning algorithms used in [18] to evaluate the performance of FILMD

It can be concluded that the FILMID algorithm which was based on double profiles has given better results than the other three algorithms (Clustering, C4.5Rules and RIPPER). In addition, the detection time was also calculated for both the RIPPER and the FILMID algorithms using the same computer and operating system. The detection time for the FILMID algorithm was better than that of RIPPER since the FILMID algorithm generates fewer rules than the RIPPER algorithm [18].

32.11 Situational Awareness

Situation awareness tries to capture sequential patterns in attack data, rather than focusing on classifying a single event. The idea is that a chain of suspicious events makes it much more likely that an attack has occurred. Since it determines which events are likely related, the number of attack events can be greatly reduced since many anomalous traffic events may be part of the same attack. A different empirical evaluation phase is presented in [19], which used situational awareness for intrusion

Table 32.7 Comparison of the detection results between SVM and Back propagation [18]

Algorithm	Total number/the number of error categorization	Detection accuracy
SVM	350/331	94.57%
Back propagation	350/307	87.71%

detection. Sensors provide input for the events in the IDS. After reporting the records of the sensors during all the stages, the alert events generated from security sensors will be exposed to simplifying, filtering, fusing and correlating. This led to decreasing warning events from 64,481 to 6,164. A risk value was also calculated which could be used to know the route of the attack [19].

32.12 Back Propagation

To some extent, the empirical evaluation phase in [18] was similar to what was used in the previously mentioned machine learning approaches. The KDD-Cup99 dataset of MIT Lincoln Lab was used in the experiment. Four SVMs were used to generate the detection model. The training dataset was divided into 300 samples of normal data and 200 intrusion samples from DoS, U2R, R2L and Probing datasets. The testing dataset was divided into 200 normal samples of normal data and 150 intrusion samples from DoS, U2R, R2L and Probing datasets. The detection rates given by the system based on SVM and back propagation network are shown in Table 32.7 [18]. In [20], the model was evaluated by calculating two performance measures: Detection Rate and FAR after applying back-propagation algorithm on the KDD cup99 dataset.

32.13 Fuzzy Logic

Fuzzy logic is a relatively new approach in intrusion detection. The benefit of a fuzzy system is that training of the system (in a machine learning sense) is not required. On the other hand, it is not as flexible to adapt to attacks which are novel. For instance, consider an attack that is nothing like an attack that is defined within the fuzzy rules—it will likely be misclassified. Consider a second attack that is similar to one of the approaches, but not quite the same—it may be misclassified by the fuzzy rules. The empirical evaluation used in [21], which was based on fuzzy logic, was specified for calculating the overall accuracy. Calculating the overall accuracy was based on some criteria such as the False Positive (FP), False Negative (FN), True Positive (TP) and True Negative (TN) rates. After calculating all the above criteria for the system, the classification performance of the system was noticeably improved so that it gave more than 90% accuracy for the four types of attacks [21].

Table 32.8 Metrics used

Metric	Papers
True Positive (TP)	[14, 22–24]
False Positive (FP)	[2, 14, 17, 22–28]
False Negative (FN)	[2, 17, 22–24, 26, 28]
True Negative (TN)	[22–24]
Recall	[22, 23, 29]
F-measure	[22, 23, 29, 30]
Precision	[22, 23, 29]
Overall accuracy	[20, 22–24, 28–33]
Prediction Accuracy (PA) or Detection Rate(DR)	[2, 14, 17, 20, 25–27, 30, 33–37]
False Alarm Rate (FAR)	[2, 12, 20, 23, 27, 29–31, 33, 35–37]
Correctly Classified Instances (CCI)	[14]
Incorrectly Classified Instances (ICCI)	[14]

32.14 Comparisons of IDSs by Metrics Used

Generally, researchers use different kinds of empirical evaluation metrics to evaluate their IDS. To summarize, Table 32.8 above shows the most used metrics and the papers that used them.

32.15 Conclusion

In conclusion, in this chapter has provided a review of the contemporary intrusion detection systems literature structured around a metrics and performance evaluation criteria. The aim of the chapter is to provide the reader with a different and new review and taxonomy of the IDSs spanning a variety of approaches, with a particular emphasis on approaches which are flexible and perform well with respect to the metrics outlined. First, we outline the typical datasets used in intrusion detection system performance evaluation. Then, the metrics used to evaluate the performance of IDSs were also defined and explained. The chapter then gives an overview of a large variety of approaches which are used to detect anomalies, intrusions and improper behaviour on the network. Lastly, this chapter explained the empirical evaluation of the IDS by discussing multiple metrics used in these kinds of systems including the standard ones and custom ones. The value of this chapter lies not only on its treatment of the source papers discusses, but also in its novel style in presenting the information about IDSs to the reader. The chapter's concept is to make the reader flows with the stream of the data from the input through the internal processing to the final output decision. This manner gives researchers a comprehensive knowledge about ID and what has been done in this field until now in terms of pre-processing, classifier algorithms and the achieved results. In addition, this style helps the reader to find which feature can be used in detecting certain kind of intrusions and which papers have used that. Another benefit of this approach is that it can reveal which papers have used training and testing or testing data only. It is hoped that this chapter will have a significant impact on future research in the IDS area by providing readers new to this area which a “jumping-off point” into the source literature. In addition, the structure of the review should provide some perspective of how researchers can investigate specific aspects of IDS and what solutions have been previously explored within each aspect. In addition, the review conducted important comparisons and provided some critiques after each component of IDS supported by some tables to give the reader a better perspective about that particular component. Intrusion detection will remain an interesting research topic for as long as there are intruders trying to gain illicit access to a network. The discipline represents a perpetual arms-race between those attempting to gain unauthorized control and those trying to prevent them. We hope that this chapter has provided an overview of this fascinating field and a starting point for future study.

References

1. Cup, K. (1999). *Dataset.* Available at the following website http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html
2. Sharma, V., & Nema, A. (2013). Innovative genetic approach for intrusion detection by using decision tree. In *2013 international conference on communication systems and network technologies* (pp. 418–422).
3. Lippmann, R., Haines, J. W., Fried, D. J., Korba, J., & Das, K. (2000). The 1999 DARPA off-line intrusion detection evaluation. *Computer Networks, 34*(4), 579–595.
4. J. G. Elevate Communications (2016). Terabit-scale multi-vector DDoS attacks to become the new normal in 2017, Predict DDoS Experts, *Business Wire*.
5. García, S., Grill, M., Stiborek, J., & Zunino, A. (2014). An empirical comparison of botnet detection methods. *Computers & Security, 45*, 100–123.
6. Małowidzki, M., Berezinski, P., & Mazur, M. (2015). Network intrusion detection: half a kingdom for a good dataset. In *Proceedings of NATO STO SAS-139 Workshop*. Portugal.
7. Scully, P. (2016). Where can I get the latest dataset for a network intrusion detection system?. *Quora* [Online]. Available: https://www.quora.com/Where-can-I-get-the-latest-dataset-for-a-network-intrusion-detection-system. Accessed January 12, 2017.
8. ubershmekel (2012). Precision, recall, sensitivity and specificity. *Ubershmekel’s Uberpython Pythonlog* [Online]. Available: https://uberpython.wordpress.com/2012/01/01/precision-recall-sensitivity-and-specificity/. Accessed February 09, 2017.
9. Natesan, P., Balasubramanie, P., & Gowrison, G. (2012). Improving the attack detection rate in network intrusion detection using adaboost algorithm. *Journal of Computer Science, 8*(7), 1041–1048.
10. Mo, Y., Ma, Y., & Xu, L. (2008). Design and implementation of intrusion detection based on mobile agents. In *2008 IEEE international symposium on IT in medicine and education* (pp. 278–281).
11. Uppuluri, P., & Sekar, R. (2001). Experiences with specification-based intrusion detection. In *Recent advances in intrusion detection* (pp. 172–189).

12. Sekar, R. et al. (2002). Specification-based anomaly detection: A new approach for detecting network intrusions. In *Proceedings of the 9th ACM conference on computer and communications security* (pp. 265–274). Washington, DC, USA.
13. Shon, T., & Moon, J. (2007). A hybrid machine learning approach to network anomaly detection. *Information Science, 177*(18), 3799–3821.
14. MeeraGandhi, G., Appavoo, K., & Srivasta, S. (2010). Effective network intrusion detection using classifiers decision trees and decision rules. *International Journal Advanced network and Application*, *2*(3), 686–692.
15. Trinius, P., Willems, C., Holz, T., & Rieck, K. (2009). A malware instruction set for behavior-based analysis. Tech. Rep. TR-2009-07, University of Mannheim.
16. Xu, J., & Wu, S. (2010). Intrusion detection model of mobile agent based on Aglets. In *2010 international conference on computer application and system modeling (ICCASM 2010)* (Vol. 4, pp. V4-347–V4-350).
17. Gong, Y., Mabu, S., Chen, C., Wang, Y., & Hirasawa, K. (2009). Intrusion detection system combining misuse detection and anomaly detection using Genetic Network Programming. *ICCAS-SICE, 2009*.
18. Yang, W., Wan, W., Guo, L., & Zhang L. J. (2007). An efficient intrusion detection model based on fast inductive learning. In *2007 international conference on machine learning and cybernetics* (Vol. 6, pp. 3249–3254).
19. Lan, F., Chunlei, W., & Guoqing, M. (2010). A framework for network security situation awareness based on knowledge discovery. In *2nd international conference on computer engineering and technology* (Vol. 1, pp. V1-226–V1-231).
20. Jaiganesh, V., Sumathi, P., & Mangayarkarasi, S. (2013). An analysis of intrusion detection system using back propagation neural network. In *2013 international conference on information communication and embedded systems (ICICES)* (pp. 232–236).
21. Shanmugavadivu, R., & Nagarajan, N. (2011). Network intrusion detection system using fuzzy logic. *Indian Journal of Computer Science and Engineering (IJCSE), 2*(1), 101–111.
22. Sen, J. (2010). Efficient routing anomaly detection in wireless mesh networks. In *2010 first international conference on integrated intelligent computing* (pp. 302–307).
23. Aggarwal, P., & Sharma, S. K. (2015). An empirical comparison of classifiers to analyze intrusion detection. In *2015 fifth international conference on advanced computing communication technologies* (pp. 446–450).
24. Vyas, T., Prajapati, P., & Gadhwal, S. (2015). A survey and evaluation of supervised machine learning techniques for spam e-mail filtering. In *2015 IEEE international conference on electrical, computer and communication technologies (ICECCT)* (pp. 1–7).
25. Rieck, K., Schwenk, G., Limmer, T., Holz, T., & Laskov, P. (2010). Botzilla: Detecting the phoning home of malicious software. In *Proceedings of the 2010 ACM symposium on applied computing* (pp. 1978–1984).
26. Lane, T. (2006). A decision-theoretic, semi-supervised model for intrusion detection. In M. A. Maloof (Ed.), *Machine learning and data mining for computer security* (pp. 157–177). London: Springer.
27. Warrender, C., Forrest, S., & Pearlmutter, B. (1999). Detecting intrusions using system calls: alternative data models. In *Proceedings of the 1999 IEEE symposium on security and privacy (Cat. No.99CB36344)* (pp. 133–145).
28. Joo, D., Hong, T., & Han, I. (2003). The neural network models for IDS based on the asymmetric costs of false negative errors and false positive errors. *Expert System with Applications, 25*(1), 69–75.
29. Kolias, C., Kambourakis, G., Stavrou, A., & Gritzalis, S. (2016). Intrusion detection in 802.11 networks: Empirical evaluation of threats and a public dataset. *IEEE Communications Surveys Tutorials, 18*(1), 184–208.
30. Subramanian, U., & Ong, H. S. (2014). Analysis of the effect of clustering the training data in Naive Bayes classifier for anomaly network intrusion detection. *Journal of Advances in Computer Networks, 2*(1), 91–94.

31. Casas, P., Mazel, J., & Owezarski, P. (2012). Unsupervised network intrusion detection systems: Detecting the unknown without knowledge. *Computer Communications, 35*(7), 772–783.
32. Muzammil, M. J., Qazi, S., & Ali, T. (2013). Comparative analysis of classification algorithms performance for statistical based intrusion detection system. In *3rd IEEE international conference on computer, control and communication (IC4)* (pp. 1–6).
33. Tan, Z., Jamdagni, A., He, X., Nanda, P., Liu, R. P., & Hu, J. (2015). Detection of denial-of-service attacks based on computer vision techniques. *IEEE Transactions Computers, 64*(9), 2519–2533.
34. Bhuse, V., & Gupta, A. (2006). Anomaly intrusion detection in wireless sensor networks. *Journal of High Speed Networks, 15*(1), 33–51.
35. Zhao, Y. J., Wei, M. J., & Wang, J. (2013). Realization of intrusion detection system based on the improved data mining technology. In *8th international conference on Computer Science and Education*. Colombo, Sri Lanka.
36. Mahoney, M. V., & Chan, P. K. (2001). *PHAD: Packet header anomaly detection for identifying hostile network traffic* (Tech. Rep. CS-2001-4). Melbourne, FL: Florida Institute of Technology.
37. Sedjelmaci, H., & Senouci, S. M. (2015). An accurate and efficient collaborative intrusion detection framework to secure vehicular networks. *Computers and Electrical Engineering, 43*, 33–47.