Weighted Vote Algorithm Combination Technique for Anomaly Based Smart Grid Intrusion Detection Systems

Joris Lueckenga and Dominik Engel
Josef Ressel Center for User-Centric
Smart Grid Privacy, Security and Control
Salzburg University of Applied Sciences
Salzburg, Austria
Email: joris.lueckenga@en-trust.at
Email: dominik.engel@en-trust.at

Robert Green
Department of Computer Science
Bowling Green State University
Bowling Green, OH 43403
Email: greenr@bgsu.edu

***Abstract*—Intrusion Detection systems (IDS) are a crucial and necessary aspect of the smart grid, particularly when considering the possible attack vectors and their consequences. While there are many different approaches on IDS for Smart Grid, the benefits of an anomaly detection technique is still in discussion, due to its capability of detecting zero-day attacks and misuse. This paper proposes a weighted vote classification approach and a general weight calculation function to improve the detection performances of anomaly IDS systems. Initial results show that a combination technique is able to improve classifier performance by several percent.**

I. Introduction

Since the beginning of electrical power distribution, many ways have been found to compromise the metering system in order to gain financial benefits. A collection of attacks on the grid components and other issues regarding power grid security are given in [1]. These examples show that even with digital technology and securely acting implementations, attackers can and will try to manipulate the power grid infrastructure. One specific Smart Grid scenario, in which an attacker tries to lower the billing rates is discussed in [2]. This scenario describes the procedure of packet manipulation to inject false data into the Smart Grid network. As a result, billing rates are compromised and other clients have to pay more. Another problem is the intention to weaken a company or country by invading communication networks to limit or stop critical services. Recent events show that hacking activity is used for espionage and can be used as an instrument to threaten or damage countries. Viruses or attacks, like Stuxnet or countless examples of exploits available in the Internet are menacing threats for the Smart Grid security. In a survey of the anti-virus company Symantec, attacks on the energy sector were gathered and presented. It states that in the second half of 2012, the energy sector was the second most targeted asset, and attackers tended to go after valuable information. In addition to that, the sector is also a major target for sabotage attacks [3]. With the possibility of controlling electrical devices and cutting power supply, the Smart Grid functionality must be ensured and secured. Even though anomaly detection does have downsides (e.g.false positives, etc.), the flexibility of its implementation and the possibility to detect zero day attacks encourage the use of this technique. As it has previously been shown that the use of ensemble techniques may outperform single classifiers [13], this work focuses on extending the current research in the area of smart grid IDSs by proposing a novel, ensemble classification methodology using weighted voting to identify potential threats.

II. Literature Review

The area of intrusion detection, both in general and in the area of smart grid, is a well researched area [4], [5]. Some concepts have the potential to be adapted and integrated into a Smart Grid security concept including improvements in the Mobile WAN Connection, Rule-based IDS, Domain Knowledge for IDS, general Smart Grid Intrusion Detection systems, and ensemble classification.

A. Mobile WAN Connection Concept

In the work of [6], the Smart Meter is the core element of all functionality. Due to the mobile Internet connection, which is used to communicate with the power provider, a network infrastructure besides the Home Area Network (HAN) is not required. Also, the Smart Meter is considered a multi-functional device which contains a firewall and provides Power Line Connection, Zig-Bee or WLAN for HAN communication. To ensure the security and integrity of the connection, a tamper-resistant cryptoprocessor is used. The device will also have cryptographic algorithms and functions to support a public-private key infrastructure. The concept also suggests that, apart from the sender and receiver part of the transmitted data, packets should be fully encrypted. Using encryption would make this system safe, if the methods are applied and implemented correctly. It also would make IDS rather difficult, because the package content could not be read and checked against attacks. Another issue, which Anderson states in [7], is the problem of key compromising. If the private key of a provider is acquired, an attacker could remotely control a high number of meters and would be able to cause large electricity blackouts. Nevertheless, the concept provides good security measures and does not require the construction of a NAN and WAN infrastructure.

B. Rule-Based IDS

In the work of [8], a Behavior-Rule-Based Intrusion Detection System is suggested. The argumentation is that anomaly detection methods cannot avoid false classifications and still have too many false positives in the current research. Therefore it is argued, that they are not suitable for Smart Grid implementations. Instead, the work suggests a derivation of a specification-based IDS technique. In order to specify constraints for the possible actions of network nodes, a table of behavior rules is created. A rule defines valid behavior, for example the increase of billing rates on high demand. Any derivations from this rule would be considered as an attack. Every node has a monitor and a trustee to be able to control the behavior of each other. This approach can be very effective, since known as well as unknown attacks would trigger invalid behavior, like lowering the price, even if the demand is high. The downside of this is similar to the specification based concepts. If complex implementations already exist or if there is not enough effort done to be able to implement this rule based concept, the creation of this type of IDS might become too complex

C. Domain Knowledge for IDS

One method of enhancing anomaly-based intrusion detection is presented in the work of [8]. This concept suggests using Fuzzy Logic systems in order to represent human domain knowledge. To achieve this, several rules have to be defined. One given example is the increase of protocols during an attack. This can be the case if a system only uses a small variety of protocols and it is rather unlikely that a large protocol variety occurs. This knowledge can be mapped to a fuzzy rule, which controls the sensitivity of an anomaly detection algorithm. In case the amount of protocols increases, the detection sensitivity is high and packets are more likely to be considered as an attack. This triggers more findings. It will also increase the likelihood of false positives, but since an attack might occur, it would be useful to find every malicious packet. The other way round would happen when the number of protocols is low. In this case, less detections are triggered. This concept could be used very efficiently, when good human domain knowledge rules can be found for a system. It also could be combined with an already existing implementation, in order to enhance detection accuracy.

D. Smart Grid Intrusion Detection System

One promising and popular concept for detecting anomalous traffic in the Smart Grid is the Smart Grid Distributed Intrusion Detection System (SGDIDS) [9]. This concept uses interconnected, distributed IDS nodes in a network infrastructure. The concept was tested with a modified KDD-NSL dataset and three different anomaly detection methods were used. Also, clustering algorithms CLONALG and AIRS2Parallel were used in order test unsupervised classification efficiency. The basic principle is to have IDS systems in every network layer, which are able to communicate between the IDS nodes in the hierarchy. In case a node in the lower level of the hierarchy cannot classify the data, the packet is passed to the next instance, which has more powerful detection algorithms. Since this approach uses a popular infrastructure as well as a widely usable and retrofittable anomaly detection, this work aims to further improve the concept by providing a new approach for anomaly detection algorithms, which are used in the IDS nodes.

E. Ensemble Classification Concept

Ensemble classification scenarios already exist in different implementations [10], [13]. Ensemble techniques, as they are used in [11], [12], were successfully implemented in many other pattern recognition tasks. A common technique is to combine differently trained models in a voting system in order to improve the classification results. Usually, a large number of weak classifiers are used to achieve this. Different approaches exist to realize the combination, as stated in [13]. In general, two common methods show that models can be combined either with a majority vote or with a weighted technique. When a majority vote is used, the most votes collected for a single decision wins. This usually requires an uneven amount of votes or a decision rule, in case the votes are even. Another method is to use a weighted vote, which implies that for every classifier model, a certain weight is assigned. Each classifier votes with its weight and a decision will be made based on the highest assigned weight. The weight calculation can be done with different methods. In this approach, the weight is calculated based on the performance of a test set. Also, to achieve the best results, different weights will be used when a classifier either votes for an attack or against it. The difference in common methods is the use of a diverse set of algorithms in the combination, instead of differently trained models of the same algorithm. A similar concept has been used in [14]. To carry out this voting and weight checking, each classifier is first trained and the performance is determined by classifying a test set.

The combination is carried out by integrating true negative (TN) and true positive (TP) rates in a function and assign the calculated weight. In this case, it is intended that weights for normal and attack decisions differ. Each of the model weights are summed up and compared afterwards, in order to carry out the final decision. To assign and calculate weights for each model, a formula was developed to balance the given amount in an efficient way. Due to the problem that several classifiers might work better than others, it is crucial to assign more or less weight, based on the performance. To do this, the following function $f(x)$ was developed:

$$f(x) = \frac{1}{(1-x) * a + b} \tag{1}$$

The variables a and b are used for various adjustments. The variable x stands for the precision of TP or TN values. Those values are between 0 and 1 and represent the probability of a correct prediction of either normal traffic or an attack. The closer the value is to 1, the better is the prediction rate. This means that high values should produce high weights, which is enforced by the formula. Weight values increase strongly as they get closer to one. On the other hand, lower values in classification performance will be penalized with low weight. The a value is used to control the slope of the function. A small a value will create a slower rising slope, beginning with lower x values. This means that even low performance numbers get a higher base weight and the impact of the formula is decreased. On the other hand, when higher values are used, only very well performing classifiers will have higher weights assigned. The maximum achievable weight is also decreased. To be able to control the highest assigned value and to avoid an infinite number for the weight, the b value can be adjusted. Very small b values will allow the weight value to rise very high. When x is close to 1, smaller b values will decrease the slope and also decrease the impact of the formula. For example, if a 1 is chosen for b, the weight formula is practically non-existent and, therefore, the classification can be compared to a majority vote.

III. Test Setup

For testing and result analysis, a python environment with the Scikit-learn library [15] was used. The project was complemented with developed code and the open source library is available on Bitbucket[1]. The tested smart grid network is identical to that in [9] and the metholology for when and how to pass information between layers is also identical.

A. Dataset and Scaling

To evaluate the classification performance, the KDD-NSL dataset is used. Even though this work suggests a Smart Grid solution, an Internet traffic dataset was chosen for the evaluation. This is due to the lack of available Smart Grid communication data. A dataset with attacks and Smart Grid traffic is not yet available. The files can be downloaded on the website of the Information Security Center of Excellence [16]. This dataset is a modified version of the dataset KDD Cup 1999. For the classification purpose, 41 features with four different attack types are contained in the dataset. It is composed with normal traffic, U2R, R2L, Probing and DoS attacks. Contrary to the KDD Cup 1999, the NSL dataset comes down to a size of about 20 Megabytes, which makes it very applicable for experimenting with classification systems. The NSL training set consists of 125,973 instances, the testing set has 22,544. The Training dataset has 45,927 DoS and 11,656 Probing instances. The amount of R2L and U2R types is rather small, only 52 instances of U2R and 995 of R2L. In contrast to that, the amount of R2L and U2R attacks in the testing set is fairly large. There are 2,938 R2L instances and 781 U2R attacks. This means that an efficient training for those attacks is rather difficult with the provided data. This condition implies that an IDS needs high zero-day attack detection capability, in order to be efficient.

[1]https://bitbucket.org/bgsufhs/python-intrusion-detection

TABLE I
Model results for atan-scaled KDD-NSL test set.

	Decision Tree	AdaBoost	kNN	SVM
True Positive	85%	91%	97%	96%
True Negative	78%	76%	68%	70%
False Positive	15%	9%	3%	4%
False Negative	22%	24%	32%	30%
Accuracy	81.2%	83.1%	79%	80%

For the dataset preparation, several steps had to be completed. In order to use the set with Scikit-Learn modules, the string types had to be mapped to numeric values. This was carried out with a developed library provided within the project. In addition, scaling methods were used for more efficient classification and to produce a variety of different outputs. This helped to further identify the voting-classifier performance. After the KDD-NSL set had been mapped to only numeric values, either no scaling was applied, normalized scaling with the range between -1 and 1 or Arctangent scaling was used.

Since the dataset has a very specific test set, another modification was carried out to add three more testing scenarios. Therefore, the whole KDD-NSL set, including test and training data, was randomized. This balanced the intrusion and normal traffic instances and generated a dataset with increased training and testing performance. The newly generated sets were split into sets and scaling was applied. An issue that had to be addressed was the requirement of a weight-calibration set, in order to determine each classifier performance and calculate the weight. To do this, the test sets were bisected and one part was used for calibration. The other half was used for testing. For the modified dataset, a 5% split of the randomized data was used for the calibration task.

B. Classifiers

The classifier selection was composed of a K-Nearest Neighbour (kNN) , Decision Tree, AdaBoost and a Support Vector Machine (SVM) model. kNN was configured with $k = 1$ and "distance" as weight. The Decision Tree classifier used a minimum of 20 leaf samples and the AdaBoost classifier was configured to use a DecisionTree algorihm with a maximum depth of 3. The maximum allowed estimators were set to 70. For the SVM, a Radial Basis Function kernel was chosen, with $C = 1$ and a degree of 3. For evaluation purposes, a confusion plot was created and the overall TN,TP, false positive (FP) and false negative (FN) percentages were calculated. In addition to that, the accuracy for overall efficiency was given. The stated setup of classifiers uses different algorithms, where each produce a different output. Table I shows a test run with an arctagent scaled NSL-KDD dataset.

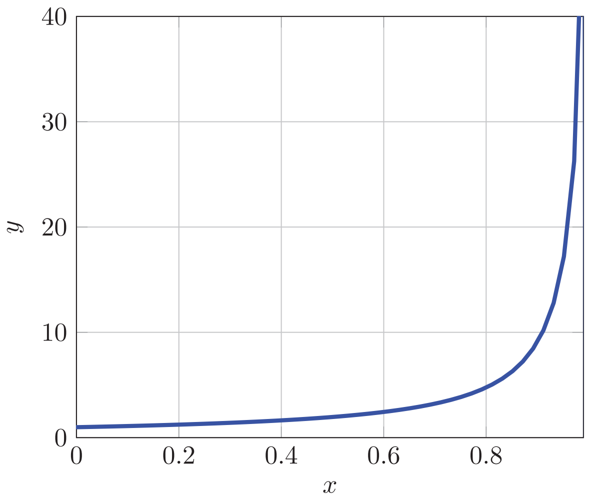

Fig. 1. Weight calculation function.

For the weight calibration, the constant a was set to 1, b was set to 0.01, resulting in (2) being used for weighting. For x, the specific TN or TP rate is used. TP rates are used to calculate the attack weight and TN is used for normal traffic weight. The chosen function values result in a graph as it is shown in Fig. 1.

$$f(x) = \frac{1}{(1-x)+0.01} \tag{2}$$

The graph rises when x is close to 0.85, which implies that 85% of the existing normal or attack instances in the dataset have been found. The intention with this formula is to reward good performances with a higher weight and increase the likelihood of a correct vote-prediction. To carry out the final classification, the assigned, summed up weights for attacks and normal predictions are compared with each other.

IV. Initial Results

For result analysis, the best accuracy, which was achieved by a single model, and the occurring false negative and positive values are compared with those of the vote-classifier. The results for all six scenarios are presented in Table II.

This data shows that voting was able to obtain a higher level of accuracy in most of the cases. To further illustrate the net performance improvements, Table 3 shows the subtracted values.

When the modified test set was used, each of the models started to become very efficient. In two cases, a large performance gap occurred between the classifiers, which caused the vote classification to be less accurate. In classification scenarios with lower performance, overall accuracy was increased significantly and either or both of the FP and FN rates were lowered. In the last case, the increase of performance is limited to very few false predictions, since the performance is already close to the 100%. Best possible results were achieved with the atan-scaled NSL set and the modified normalized set. The weighted vote was able to improve both the FP and FN rates and increased the accuracy. In two cases, individual models produced a lower FP rate than the vote classifier, but were not able to uncover many attacks. To further test, if this generalized approach of combination is exact enough, a script was programmed to iterate through the a and b values of the function, to find better working parameters. The results showed that only very little improvements on individual scenarios were achieved, which implies that the suggested function might be a good approximation to combine the models efficiently.

TABLE III
Net improvement values of vote classification.

Set/Scaling	Accuracy	False Positive	False Negative
NSL (Arc.)	+3.60%	-0.53%	-4.01%
NSL (Norm.)	+2.69%	+6.70%	-5.60%
NSL (None)	+2.31%	+4.26%	-4.06%
Mod. NSL (Arc.)	-0.02%	-0.09%	+0.10%
Mod. NSL (Norm.)	0.13%	-0.11%	-0.12%
Mod. NSL (None)	-0.03%	+0.00%	+0.05%

V. Conclusion

The reduction of FN or FP-rates and with it, the improvement of classification accuracy is a complicated task for even a single classifier. In the presented scenario, this is even more difficult as the use of an ensemble of techniques is being used, resulting in increased complexity. In contrast to this downside, several benefits have been discovered with the use of the voting-based ensemble classifier. With the experiments carried out on the classifier-voting system, it has been observed that a voting technique was able to show, in most cases, significant increase in prediction accuracy. When a combination of very efficient classifiers was used, the improvement of accuracy was either the same or only slightly significant. The positive aspects of this study included dropping FP- or FN-rates. Low FP rates are in many cases more favorable in machine-to-machine traffic, when the same accuracy can be achieved. Another aspect treated was the combination technique. Although the output is always dependent on the chosen classifiers and their performance, the weight balancing formula was able to produce favorable results in most of the test scenarios. This general approach with the stated formula proved to be successful in the different test scenarios. Based on these findings, it might be possible to apply a successful voting mechanism in a Smart Grid network. Especially since there are not any detailed attack scenarios available yet, clustering classifications or outlier detection might be used in the future. Those algorithms often have lesser performance than supervised classifiers and a voting scenario might be able to improve the resulting accuracy. Also, due to the machine-to-machine generated traffic of Smart Grid applications, a feature space for efficient classification might be developed more easily than in Internet applications. This will result in strong classification algorithms which can be improved afterwards with voting. Even if the enhancements are limited to several

TABLE II
MODEL RESULTS FOR ATAN-SCALED KDD-NSL TEST SET COMPARING SINGLE CLASSIFERS TO VOTE CLASSIFIERS.

	Single Classifier			**Weighted Voting Classifier**		
Dataset (Scaling Method)	**Accuracy**	**False Positive**	**False Negative**	**Accuracy**	**False Positive**	**False Negative**
NSL (Arctangent)	83.41%	8.75%	24.19%	87.01%	8.22%	20.18%
NSL (Normalized)	78.45%	2.57%	32.93%	81.14%	9.27%	27.33%
NSL (None)	78.03%	3.65%	33.04%	80.34%	7.91%	28.98%
Modifed NSL (Arctangent)	99.81%	0.20%	0.17%	99.79%	0.11%	0.27%
Modifed NSL (Normalized)	99.67%	0.23%	0.40%	99.80%	0.12%	0.28%
Modifed NSL (None)	99.86%	0.11%	0.15%	99.83%	0.11%	0.20%

percent, the output can avoid thousands of false or negative detections on the long run. Elaborated scenarios also showed the strong dependency on the chosen scaling method and training data. In terms of FP-Rates, one scenario showed a decrease of false attack predictions from 0.23% to 0.12%. Even though this does not seem much, it reduces the amount of FPs by roughly 50%. For Smart Grid implementations, this can be crucial to avoid false alarms and stabilize the system behavior. In addition to that, defective packets, which might be produced due to system errors, can very likely be detected by an anomaly IDS. In general, the scenarios showed that implementations may benefit from improved accuracy in Smart Grid applications. By combining this technique with other anomaly detection approaches, this technique could find its way in reliable IDS system for the Smart Grid.

VI. ACKNOWLEDGMENTS

The financial support of the Josef Ressel Center by the Austrian Federal Ministry of Science, Research and Economy and the Austrian National Foundation for Research, Technology and Development is gratefully acknowledged. Funding by the Austrian Marshall Plan Foundation is gratefully acknowledged.

REFERENCES

[1] R. Anderson and S. Fuloria, “Who controls the off switch?” in *IEEE International Conference on Smart Grid Communications (SmartGridComm)*, October 2010, pp. 96–101.

[2] C.-H. Lo and N. Ansari, “CONSUMER: A Novel Hybrid Intrusion Detection System for Distribution Networks in Smart Grid,” *IEEE Transactions on Emerging Topics in Computing*, vol. 1, no. 1, pp. 33–44, June 2013.

[3] C. Wueest, “Targeted attacks against the energy sector: Security response,” Symantec, Tech. Rep., 2014.

[4] M. Ahmed, A. N. Mahmood, and J. Hu, “A survey of network anomaly detection techniques,” *Journal of Network and Computer Applications*, vol. 60, pp. 19–31, 2016.

[5] A. Murillo, “Review of Anomalies Detection Schemes in Smart Grids,” Grupo de Teleinformatica e AutomacaoSymantec, Tech. Rep., 2013.

[6] J. Naruchitparames, M. Giine, and C. Evrenosoglu, “Secure communications in the smart grid,” in *IEEE Consumer Communications and Networking Conference*, January 2011, pp. 1171–1175.

[7] R. Mitchell and I.-R. Chen, “Behavior-Rule Based Intrusion Detection Systems for Safety Critical Smart Grid Applications,” *IEEE Transactions on Smart Grid*, vol. 4, no. 3, pp. 1254–1263, September 2013.

[8] O. Linda, M. Manic, and T. Vollmer, “Improving cyber-security of smart grid systems via anomaly detection and linguistic domain knowledge,” in *International Symposium on Resilient Control Systems*, Aug 2012, pp. 48–54.

[9] Y. Zhang, L. Wang, W. Sun, R. Green, and M. Alam, “Distributed Intrusion Detection System in a Multi-Layer Network Architecture of Smart Grids,” *IEEE Transactions on Smart Grid*, vol. 2, no. 4, pp. 796–808, December 2011.

[10] Y. Ren, L. Zhang, and P. Suganthan, “Ensemble Classification and Regression-Recent Developments, Applications and Future Directions,” *IEEEComputational Intelligence Magazine*, vol. 11, no. 1, pp. 41–53, February 2016.

[11] X. Mu, J. Lu, P. Watta, and M. Hassoun, “Weighted voting-based ensemble classifiers with application to human face recognition and voice recognition,” in *International Joint Conference on Neural Networks*, June 2009, pp. 2168–2171.

[12] M. Panda and M. Patra, “Ensemble Voting System for Anomaly Based Network Intrusion Detection,” *International Journal of Recent Trends in Engineering*, vol. 2, no. 5, 2009.

[13] T. G. Dietterich, “Ensemble Methods in Machine Learning,” in *Proceedings of the First International Workshop on Multiple Classifier Systems*, ser. MCS ’00. London, UK, UK: Springer-Verlag, 2000, pp. 1–15.

[14] A. Borji, *Advances in Computer Science – ASIAN 2007. Computer and Network Security: 12th Asian Computing Science Conference, Doha, Qatar, December 9-11, 2007*, Berlin, Heidelberg, 2007, ch. Combining Heterogeneous Classifiers for Network Intrusion Detection, pp. 254–260.

[15] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, “Scikit-learn: Machine Learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[16] Information Security Center of Excellence, “Nsl-kdd data set for network-based intrusion detection systems,” http://nsl.cs.unb.ca/NSL-KDD/, 2009.