A real time OCSVM Intrusion Detection module with low overhead for SCADA systems

Leandros A. Maglaras
Department of Computing, University of Surrey
Guilford, UK
Email: l.maglaras@surrey.ac.uk

Jianmin Jiang
Department of Computing, University of Surrey
Guilford, UK
Email: jianmin.jiang@surrey.ac.uk

***Abstract*—In this paper we present a intrusion detection module capable of detecting malicious network traffic in a SCADA (Supervisory Control and Data Acquisition) system. Malicious data in a SCADA system disrupt its correct functioning and tamper with its normal operation. OCSVM (One-Class Support Vector Machine) is an intrusion detection mechanism that does not need any labeled data for training or any information about the kind of anomaly is expecting for the detection process. This feature makes it ideal for processing SCADA environment data and automate SCADA performance monitoring. The OCSVM module developed is trained by network traces off line and detect anomalies in the system real time.**

In order to decrease the overhead induced by communicated alarms we propose a new detection mechanism that is based on the combination of OCSVM with a recursive k-means clustering procedure. The proposed intrusion detection module $\mathcal{K}-\mathcal{OCSVM}$ is capable to distinguish severe alarms from possible attacks regardless of the values of parameters σ and ν, making it ideal for real-time intrusion detection mechanisms for SCADA systems. The most severe alarms are then communicated with the use of IDMEF files to an IDSIDS (Intrusion Detection System) system that is developed under CockpitCI project. Alarm messages carry information about the source of the incident, the time of the intrusion and a classification of the alarm.

***Keywords*—SCADA systems; OCSVM; intrusion detection**

I. Introduction

Cyber-physical systems are becoming vital to modernizing the national critical infrastructure systems. Cyber attacks usually target valuable infrastructures assets, taking advantage of architectural/technical vulnerabilities or even weaknesses in the defense systems. While there is the case in some situations, most weaknesses in CIs arise from the fact that most CIs are adopting off-the-shelf technologies from the IT world, without a significant change in terms of the operator mindset, still based on the ”airgap” security principle that suggests that an apparently isolated and obscure system is implicitly secure. Once you open the system to off-the-shelf solutions, you also increase its exposure to cyber-attacks. Several techniques and algorithms have been reported by researchers for intrusion detection. One big family of intrusion detection algorithms is rule based algorithms. In real applications though, during abnormal situations, the behavior of the system cannot be predicted and does not follow any known pattern or rule. This characteristic makes rule based algorithms incapable of detecting the intrusion.

Generally, anomaly detection can be regarded as binary classification problem and thus many classification algorithms which are utilized for detecting anomalies, such as neural networks, support vector machines, K-nearest neighbour (KNN) and Hidden Markov model can be used. However, strictly speaking, they are not intrusion detection algorithms, as they require knowing what kind of anomaly is expecting, which deviates the fundamental object of intrusion detection. In addition these algorithms may be sensitive to noise in the training samples.

Segmentation and clustering algorithms seem to be better choices because they do not need to know the signatures of the series. The shortages of such algorithms are that they always need parameters to specify a proper number of segmentation or clusters and the detection procedure has to shift from one state to another state. Negative selection algorithms on the other hand,are designed for one-class classification; however, these algorithms can potentially fail with the increasing diversity of normal set and they are not meant to the problem with a small number of self-samples, or general classification problem where probability distribution plays a crucial role. Furthermore, negative selection only works for a standard sequence, which is not suitable for online detection. Other algorithms, such as time series analysis are also introduced to anomaly detections, and again, they may not be suitable for most of the real application cases.

To minimize the above mention drawbacks an intelligent approach based on OCSVM [One-Class Support Vector Machine] principles are proposed for intrusion detection. OCSVM is a natural extension of the support vector algorithm to the case of unlabeled data, especially for detection of outliers. The OCSVM algorithm maps input data into a high dimensional feature space (via a kernel) and iteratively finds the maximal margin hyperplane which best separates the training data from the origin (Figure 1).

OCSVM principles have shown great potential in the area of anomaly detection [1]–[4]. IDS can provide active detection and automated responses during intrusions [5]. Commercial IDS products such as NetRanger, RealSecure, and Omniguard Intruder alert work on attack signatures. These signatures needed to be updated by the vendors on a regular basis in order to protect from new types of attacks. Most of the current intrusion detection commercial softwares are based on approaches with statistics embedded feature processing, time series analysis and pattern recognition techniques. Several extensions of OCSVM method have been introduced lately [6]–[8]

- Certain sensors deliver more or less information about certain types of attacks. The object oriented approach allows flexibility while the subclassing rule provides the integrity of the model.

A typical IDMEF file produced by our system is shown in Figure 7. The IDMEF message contains information about the source of the intrusion, the time of the intrusion detection, the module that detected the problem and a classification of the detected attack. The source node that the intrusion is detected is very important feature in an IDS system. Once the infected node is spotted the infection can be limited by the isolation of this node from the rest network. Fast and accurate detection of the source node of a contamination is crucial for the correct function of an IDS.

```
<?xml version="1.0" encoding="UTF-8"?>
<idmef:IDMEF-Message version="1.0" xmlns:idmef="http://iana.org/idmef">
  - <idmef:Alert>
      - <idmef:Analyzer>
          - <idmef:Node category="unknown">
              <idmef:location>IT Network</idmef:location>
              <idmef:name>OCSVM</idmef:name>
            </idmef:Node>
        </idmef:Analyzer>
        <idmef:CreateTime ntpstamp="0x1123">2014-02-13</idmef:CreateTime>
      - <idmef:Source>
          - <idmef:Node>
              - <idmef:Address>
                  <idmef:address>AsustekC_b2:ce:52</idmef:address>
                </idmef:Address>
            </idmef:Node>
        </idmef:Source>
        <idmef:Classification text="POSSIBLE ALARM"/>
    </idmef:Alert>
</idmef:IDMEF-Message>
```

Fig. 7. Typical IDMEF message produced by OCSVM module

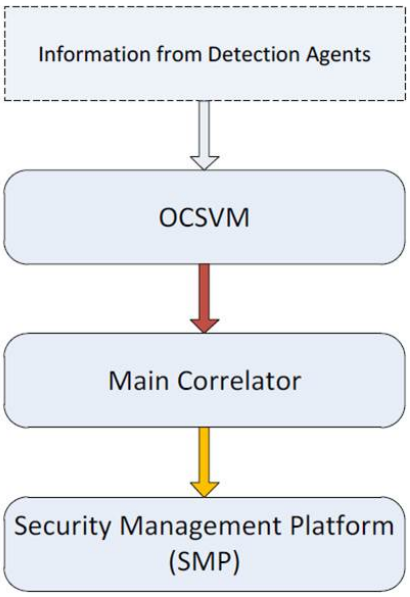

Fig. 8. Interfaces linked with OCSVM detection module

```
1 1: 0 2: 0.101694915254237
1 1: 1.08894782018269E-05 2: 0.101694915254237
1 1: 0.318975236045454 2: 0.108474576271186
1 1: 0.000146687676954043 2: 0.108474576271186
1 1: 1 2: 0.101694915254237
```

Fig. 9. Format of the transformed Network trace file

B. OCSVM interfaces

Once the detector training phase is complete, OCSVM detection is capable of detecting possible intrusions (abnormal behavior) on the SCADA system. Detection agents will gather new monitoring data (with corresponding attributes) and will feed the data to the OCSVM detection module. The detection module will classify each event whether it is a normal event or a possible intrusion. This information will then be send to the main correlator in order to react accordingly to the detected intrusions as shown in Figure 8

VII. Performance Evaluation

A. Training of OCSVM model

Training of OCSVM module was conducted using the transformed network trace file (Figure 9). To train the OCSVM, we adopt the RBF for the kernel equation. This kernel nonlinearly maps samples into a higher dimensional space so it can handle the case when the relation between class labels and attributes is nonlinear. The parameter σ is chosen 0.07 and the parameter ν 0.01.

The training model that is extracted after the training of the OCSVM is used for on line detection of malicious data. Since the model is based on features that are related to network traffic, and since the traffic of the system varies from area to area and from time period to time period, possible generation of multiple models could improve the performance of the module.

The network traffic in electric grids varies according to the activity which is not constant during the day. Also in some areas the activity follows different patterns according to the local demand. These characteristics maybe be critical for the proper training of the module and the accurate detection of intruders.

B. Testing of OCSVM model

In order to test our model we use the initial network trace file and we also spit the trace file in two separate files (A,B). The split was random and two datasets were constructed from the initial trace file. The two datasets were then used for training and testing. The dataset A was initially used for training and dataset B for testing and vice versa. The size of dataset A is 1000 rows and of B 570 rows. The results of our OCSVM detection module for each split are shown in Table VII-B. The accuracy of the classification of the data is high for all the tests conducted.

TABLE I. Accuracy of OCSVM module under different split of data.

Split	*Accuracy*
All	98.8796
A	98.42
B	99.12

The outcomes of the classification method for the testing conducted in whole the dataset and in the two spitted sets are

shown in Figure 11. The observed malicious data give small negative values and can only be classified as possible alerts. This is due to the fact that all the testing datasets are part of the initial trace file, which is captured during a normal operation of the system.

Malicious datasets that represent attack scenarios e.g. Man in the Middle (MITM) by ARP (Address Resolution Protocol) poisoning, SYN Flooding and honeypot [21] interaction, are used in the next subsection in order to test the performance of our $\mathcal{K}-\mathcal{OCSVM}$ module.

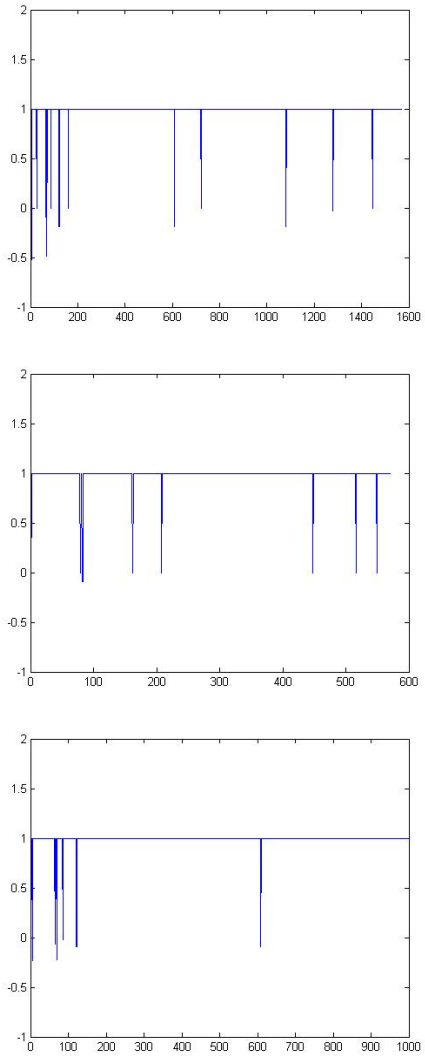

Fig. 10. OCSVM classification outcome for the three training / testing datasets (Upper figure:Original dataset, Middle figure: A/B dataset, Lower figure: B/A dataset

C. Testing of K-OCSVM model

We evaluated the performance of the method using data from the wireless network of the University campus and from a testbed that mimics a small-scale SCADA system. The parameters used for the evaluation of the performance of $\mathcal{K}-\mathcal{OCSVM}$ are listed in Table II.

Parameter	*Range of Values*	*Default value*
σ	0.1 - 0.0001	0.007
ν	0.002 - 0.05	0.01
$Threshold$	2-3	2

TABLE II. EVALUATION PARAMETERS

1) **Wireless network:** In order to test our model we use another network trace files sniffed from the wireless network. The testing trace file consists of 30.000 lines. We compare the performance of our proposed model against OCSVM classifiers having the same values for parameters σ and ν. We name each OCSVM classifier according to the parameters σ and ν : $OCSVM_{0.07,0.01}$ stands for OCSVM classifier with parameters $\sigma = 0.07$ and $\nu = 0.01$.

In Table III we show the number of observed anomalies detected from OCSVM and $\mathcal{K}-\mathcal{OCSVM}$ respectively. From this table it is shown how parameters σ, ν affect the performance of OCSVM. Even for a value of ν equal to 0.005, OCSVM produces almost 500 possible attacks, making the method inappropriate for a SCADA system where each false alarm is costly.

Parameter σ	*Parameter* ν	$\mathcal{K}-\mathcal{OCSVM}$	*ocsvm*
0.007	0.002	3	408
0.007	0.01	3	299
0.007	0.005	2	408
0.0001	0.01	3	274
0.1	0.01	2	295

TABLE III. PERFORMANCE EVALUATION OF $\mathcal{K}-\mathcal{OCSVM}$ AND OCSVM FOR $K_{thers} = 2$.

In figure 11 we present the outcome that OCSVM produces for the training network trace under different values of parameters σ and ν. From this figure it is obvious that the outcome is strongly affected by the values of these parameters, making $\mathcal{K}-\mathcal{OCSVM}$ necesary tool for proper intrusion detection.

2) **Testbed scenario:** The second trial is conducted off line with the use of two datasets extracted from the testbed (Figure 12. The testbed architecture mimics a small-scale SCADA system, comprising the operations and field networks and including a Human-Machine Interface Station (for process monitoring), a managed switch (with port monitoring capabilities, for network traffic capture), and two Programmable Logic Controller Units, for process control. The NIDS and OCSVM modules are co-located on the same host, being able to intercept all the traffic flowing on the network scopes.

During the testing period several attack scenarios are simulated in the testbed. These scenarios include network scan, network flood and MITM attack. Three kinds of attacks are being evaluated:

- **Network scan attack** In typical network scan attack, the attacker uses TCP/FIN scan to determine if ports are closed to the target machine. Closed ports answer with RST packets while open ports discard the FIN message. FIN packets blend with background noise on a link and are hard to be detected.
- **ARP cache spoofing - MITM attack ARP cache spoofing** is a technique where an attacker sends fake ARP messages. The aim is to associate the attacker's

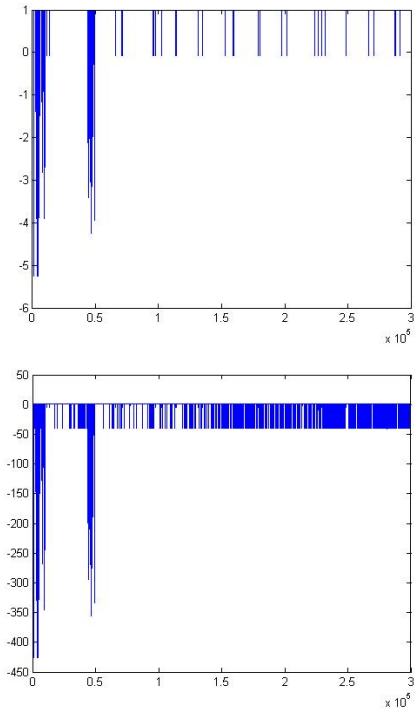

Fig. 11. OCSVM classification outcome for different values of parameters σ, ν Upper diagram : $OCSVM_{0.007,0.001}$, Lower diagram : $OCSVM_{0.01,0.05}$

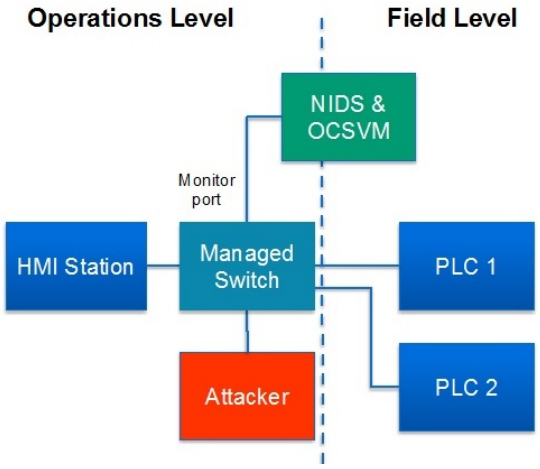

Fig. 12. Architecture of the testbed

MAC address with the IP address of another host, causing any traffic meant for that IP to be sent to attacker instead. The attacker could choose to inspect the packets, modify data before forwarding (**man-in-the-middle attack**) or launch a denial of service attack by causing some of the packets to be dropped.

- **DoS attack** Network flood is the instance where the attacker floods the connection with the PLC by sending SYN packets. In a TCP SYN flooding attack, an attackesends many SYN messages, with fictitious (spoofed) IP addresses, to a single node (victim). Although the node replies with SYN/ACK messages, these messages are never acknowledged by the client. As a result, many halfopen connections exist on the victim, consuming its resources. This continues until the victim has consumed all its resources, hence can no longer accept new TCP connection requests.

In Table IV we show the number of alert messages (IDMEF) sent from OCSVM and $\mathcal{K}-\mathcal{OCSVM}$ respectively. From this table it is shown how parameters σ, ν affect the performance of OCSVM for the testbed scenario. While for the same network trace file OCSVM produces from 10529 to 10704 alert messages according to the values of the parameters, $\mathcal{K}-\mathcal{OCSVM}$ produces the same 120 alert messages. All the reported attacks are concering the DoS attack that creates the biggest flunctuation in the network traffic.

Parameter σ	*Parameter* ν	$\mathcal{K}-\mathcal{OCSVM}$	*ocsvm*
0.007	0.002	120	10529
0.007	0.01	120	10703
0.007	0.005	120	10584
0.0001	0.01	120	10602
0.1	0.01	120	10704

TABLE IV. PERFORMANCE EVALUATION OF $\mathcal{K}-\mathcal{OCSVM}$ AND OCSVM FOR $K_{thers} = 2$.

3) **Testbed scenario with split testing periods:** Since the attacks are performed during different time periods we divide the testing dataset in several smaller ones, each containing a different attack. Testing data consists of normal data and attack data and the composition of the data sets are as follows:

- Testing set-A' : 1 - 5000: Normal data
- Testing set-B' : 5000 - 10000: Normal data + **Arp spoofing** attack + **Network scan**
- Testing set-C' : 10000 - 25000: Normal data + **Flooding Dos attack** + **Network scan**
- Testing set-D' : 25000 - 41000: Normal data + **MITM attack**

Dataset	*Initial alarms*	*Aggregated alarms*
A	129	2
B	658	3
C	9273	120
D	203	3
All	10507	3

TABLE V. AGGREGATED ALARMS PRODUCED BY $\mathcal{K}-\mathcal{OCSVM}$ ARE SIGNIFICANTLY DECREASED COMPARED TO THE INITIAL ALARMS

From table V we observe that not only the most important intrusions are detected and reported but also the total overhead on the system is limited. For all time periods the messages communicated reflect actual attacks in the network, except from the testing set-A'. In this time period HMI station demonstrated a singificant variation in the rate that it injected packets in the system between testing and training of the module. This is due to the limited training of the OCSVM and can be avoided if training dataset consists of data that represent the traffic in the network during under work loads. The increased number of alarms created from $\mathcal{K}-\mathcal{OCSVM}$ for the dataset B' is due to the fact in this time period the attacker uses an exceshive number of SYN packets in order to flood the communication channel.

VIII. Conclusion

We have presented a intrusion detection module for SCADA systems that is based in OCSVM technique. The module is trained offline by network traces, after the attributes are extracted from the network dataset. The initial attributes used for training and testing of the module are rate and packet size which represent the traffic in the system. The intrusion detection module is part of an IDS system developed under CoCkpitCI. Output of the detection module is communicated to the system by IDMEF files that contain information about the source, time and severity of the intrusion.

After the execution of the $\mathcal{K}-\mathcal{OCSVM}$ method only severe alerts are communicated to the system by IDMEF files that contain information about the source, destination, protocol and time of the intrusion. The method is stable and its performance is not influenced by the selection of parameters ν and σ. The main feature of $\mathcal{K}-\mathcal{OCSVM}$ module is that it can perform anomaly detection in a time-efficient way, with good accuracy and low overhead. Low overhead is an important evaluation metric of a distributed detection module that is scattered in a real-time system, since frequent communication of IDMEF files from detection agents degrade the performance of the SCADA network. Recursive k-means clustering, reassures that small fluctuations on network traffic, which most of the times cause OCSVM to trigger false alarms, are ignored by the proposed detection module.

As future work we will conduct an in depth performance evaluation on proposed mechanism. Using malicious and attack-free datasets of the SCADA testbed, we are going to evaluate $\mathcal{K}-\mathcal{OCSVM}$'s performance in terms of false positive rate, accuracy and runtime. Using the evaluation outcomes we are planning to enhance the proposed $\mathcal{K}-\mathcal{OCSVM}$ in order to further decrease false alarms and improve overall performance. Additional attributes like reputation of the source and the protocol used for communication may add more precision to our system and it is a matter of future research. Use of different models according to the area or the time period may further improve the performance of the method.

Acknowledgment

The authors wish to acknowledge the financial support of the project CockpitCI, funded under European Framework-7 Programme (contract No. 285647).

References

[1] Y. Wang, J. Wong, and A. Miner, "Anomaly intrusion detection using one class svm," in *Information Assurance Workshop, 2004. Proceedings from the Fifth Annual IEEE SMC*. IEEE, 2004, pp. 358–364.

[2] K. Heller, K. Svore, A. D. Keromytis, and S. Stolfo, "One class support vector machines for detecting anomalous windows registry accesses," in *Workshop on Data Mining for Computer Security (DMSEC), Melbourne, FL, November 19, 2003*, 2003, pp. 2–9.

[3] J. Ma and S. Perkins, "Time-series novelty detection using one-class support vector machines," in *Neural Networks, 2003. Proceedings of the International Joint Conference on*, vol. 3, July 2003, pp. 1741–1745 vol.3.

[4] K.-L. Li, H.-K. Huang, S.-F. Tian, and W. Xu, "Improving one-class svm for anomaly detection," in *Machine Learning and Cybernetics, 2003 International Conference on*, vol. 5. IEEE, 2003, pp. 3077–3081.

[5] D. Dasgupta and F. A. Gonzalez, "An intelligent decision support system for intrusion detection and response," in *Information Assurance in Computer Networks*. Springer, 2001, pp. 1–14.

[6] A. Glazer, L. Michael, and S. Markovitch, "q-ocsvm: A q-quantile estimator for high-dimensional distributions," in *In Proceedings of The 27th Conference on Neural Information Processing Systems (NIPS-2013),Lake Tahoe, Nevada*, 2013.

[7] X. Song, G. Fan, and M. Rao, "Svm-based data editing for enhanced one-class classification of remotely sensed imagery," *Geoscience and Remote Sensing Letters, IEEE*, vol. 5, no. 2, pp. 189–193, 2008.

[8] L. Maglaras and J. Jiang, "Ocsvm model combined with k-means recursive clustering for intrusion detection in scada systems," in *Proceedings of the 10th Qshine conference*. EAI, 2014.

[9] E. Menahem, L. Rokach, and Y. Elovici, "Combining one-class classifiers via meta learning," in *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*. ACM, 2013, pp. 2435–2440.

[10] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural computation*, vol. 13, no. 7, pp. 1443–1471, 2001.

[11] D. TAX, "One-class classification," *PhD thesis,Delft University of Technology, The Netherlands*, 2001.

[12] T. P. Runarsson and M. T. Jonsson, "Model selection in one-class ν-svms using rbf kernels," in *Proceedings of 16th International Congress and Exhibition on Condition Monitoring and Diagnostic Engineering Management*, 2003.

[13] X. Li, L. Wang, and E. Sung, "Adaboost with svm-based component classifiers," *Engineering Applications of Artificial Intelligence*, vol. 21, no. 5, pp. 785–795, 2008.

[14] R. R. R. Barbosa and A. Pras, "Intrusion detection in scada networks," in *Mechanisms for Autonomous Management of Networks and Services*. Springer, 2010, pp. 163–166.

[15] B. Zhu, A. Joseph, and S. Sastry, "A taxonomy of cyber attacks on scada systems," in *Internet of Things (iThings/CPSCom), 2011 International Conference on and 4th International Conference on Cyber, Physical and Social Computing*. IEEE, 2011, pp. 380–388.

[16] S. Karnouskos, "Stuxnet worm impact on industrial cyber-physical system security," in *IECON 2011-37th Annual Conference on IEEE Industrial Electronics Society*. IEEE, 2011, pp. 4490–4494.

[17] J. Jiang and L. Yasakethu, "Anomaly detection via one class svm for protection of scada systems," in *Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2013 International Conference on*. IEEE, 2013, pp. 82–88.

[18] R. Zhang, S. Zhang, Y. Lan, and J. Jiang, "Network anomaly detection using one class support vector machine," in *Proceedings of the International MultiConference of Engineers and Computer Scientists*, vol. 1, 2008.

[19] L. Maglaras and J. Jiang, "Intrusion detection in scada systems using machine learning techniques," in *Proceedings of the 2nd SAI conference*. SAI, 2014.

[20] H. Debar, D. A. Curry, and B. S. Feinstein, "The intrusion detection message exchange format (idmef)," 2007.

[21] L. Spitzner, "Honeypots: definitions and value of honeypots," *URL: http://www. t racking-hackers. com/papers/honeypots. html*, 2003.