Combining ensemble methods and social network metrics for improving accuracy of OCSVM on intrusion detection in…

3 authors:

Leandros Maglaras
De Montfort University
109 PUBLICATIONS   **306** CITATIONS

SEE PROFILE

Jianmin Jiang
Tianjin University
167 PUBLICATIONS   **1,494** CITATIONS

SEE PROFILE

Tiago J. Cruz
University of Coimbra
59 PUBLICATIONS   **121** CITATIONS

SEE PROFILE

Some of the authors of this publication are also working on these related projects:

Project   IEEE P61158 – Standard for Industrial Hard Real-Time Communication Working Group View project

Project   Image Processing View project

Combining ensemble methods and social network metrics for improving accuracy of OCSVM on intrusion detection in SCADA systems

Leandros A. Maglaras[a,*], Jianmin Jiang[b], Tiago J. Cruz[c]

[a]*School of Computer Science and Informatics, De Montfort University, Leicester, U.K.*
[b]*Department of Computing, University of Surrey, Guildford, Surrey, U.K.*
[c]*Department of Informatics Engineering, University of Coimbra, 15780, Portugal*

Abstract

Modern Supervisory Control and Data Acquisition SCADA systems used by the electric utility industry to monitor and control electric power generation, transmission and distribution are recognized today as critical components of the electric power delivery infrastructure. SCADA systems are large, complex and incorporate increasing numbers of widely distributed components. The presence of a real time intrusion detection mechanism, which can cope with different types of attacks, is of great importance, in order to defend a system against cyber attacks This defense mechanism must be distributed, cheap and above all accurate, since false positive alarms, or mistakes regarding the origin of the intrusion mean severe costs for the system. Recently an integrated detection mechanism, namely *IT-OCSVM* was proposed, which is distributed in a SCADA network as a part of a distributed intrusion detection system (IDS), providing accurate data about the origin and the time of an intrusion. In this paper we also analyze the architecture of the integrated detection mechanism and we perform extensive simulations based on real cyber attacks in a small SCADA testbed in order to evaluate the performance of the proposed mechanism.

Keywords: OCSVM, Intrusion detection, SCADA systems, Social analysis

---

*Principal corresponding author
  Email addresses:* `l.maglaras@surrey.ac.uk` (Leandros A. Maglaras),
`jianmin.jiang@surrey.ac.uk` (Jianmin Jiang), `tjcruz@dei.uc.pt` (Tiago J. Cruz)

1. Introduction

Cyber-physical systems are becoming vital for modernizing national critical infrastructure systems. Cyber attacks often target valuable infrastructure assets, taking advantage of architectural/technical vulnerabilities or even weaknesses in defense systems. Most weaknesses in CIs arise from the fact that many adopt off-the-shelf technologies from the IT world, without a significant change in terms of the operator mindset, thus remaining based on the "airgap" security principle that falsely assumes that an apparently isolated and obscure system is implicitly secure. Consequently, once a system is open to receiving off-the-shelf solutions, this increases its exposure to cyberattacks.

The proliferation of new technologies, especially Internet-like communications networks, may introduce some new threats to the security of a smart grid. In such a network there are three crucial aspects of security that may be threatened due to the CIA-triad, these being: confidentiality, integrity, and availability [2]. Confidentiality is the property that information is not made available or disclosed to unauthorized individuals, entities or processes. An attack on this occurs when an unauthorized person, entity or process enters the system and accesses the information. Integrity refers to safeguarding the accuracy and completeness of assets, which ensures that the information in the system will not be modified by attacks. Availability pertains to the property of being accessible and usable upon demand by an authorized entity. The resources need to be kept accessible at all times to authorized entities or processes.

Beyond cyber threats like malware, spyware, and computer viruses that currently threaten the security of computer communication networks, the introduction of new and distributed technologies, such as smart meters, sensors, and other sub-networks can bring new vulnerabilities to a smart grid [18]. In the three main control systems of a CI, the SCADA the central nerve system that constantly gathers the latest status from remote units. The communication system for wide-area protection and control of a power grid can be blocked or cut off due to component failures or communication delays. If one of the crucial communication channels fails to connect in the operational environment, inability to control or operate important facilities may occur with the possibility of power outages. In this situation, the effect of some widely known attacks can have devastating consequences on SCADA systems. Moreover, the design of SCADA systems is different from

conventional IT networks, even when based on the same physical technology, such as Ethernet networks. That is, industrial control-specific protocols are used in SCADA systems, where a limited number packet types are exchanged between entities of the network.

Intrusion detection systems can be classified into centralized intrusion detection systems (CIDS) and distributed intrusion detection systems (DIDS) by the way in which their components are distributed [1]. A CIDS is such that the analysis of the data will be performed in some fixed locations without considering the number of hosts being monitored [14], while a DIDS is composed of several IDS over large networks whose data analysis is performed in a number of locations proportional to the number of hosts. As one part of an intrusion detection system, the DIDS has specific advantages over the CIDS. For instance, it is highly scalable and can provide gradual degradation of service, easy extensibility and scalability [4]. It is evident that the development of distributed IDS specifically designed for SCADA systems, being able to ensure an adequate balance between high accuracy, low false alarm rate and reduced network traffic overhead, is needed. The above discussion clearly indicates that specific intrusion detection systems that reassure both high accuracy, low rate of false alarms and decreased overhead on the network traffic need to be designed for SCADA systems.

1.1. Motivation

Among other approaches, neural networks, support vector machines, K-nearest neighbor (KNN) and the hidden Markov model can be used for intrusion detection. While existing signature-based network IDS, such as Snort or Suricata can be effective in SCADA environments, they require specific customization for such a purpose. Also, they are not effective against rogue threats for which known patterns or signatures are not known. OCSVM principles have shown great potential in the area of anomaly detection [26, 16], being a natural extension of the support vector algorithm in the case of unlabeled data, especially for the detection of outliers.

Social network analysis (*SNA*) can be used in order to discover security policy breaches in a network and refers to the use of network theory to analyze social networks. That is, it views social relationships in terms of network theory, consisting of nodes, representing individual actors within the network, and ties which represent relationships between the individuals, such as friendships, kinships, organizations and sexual relationships. By

using comparative metrics of network's structure during normal and abnormal operation, we can discover security policy breaches in a network. One can assume that network communication between nodes, constitutes a social network of users and their applications, so the appropriate methods of social network formal analysis can be applied [12]. In on-line social systems perpetrators of malicious behavior often display patterns of interaction that are quite different from regular users, which can be identified through the application of anomaly detection techniques. Thus, in accordance [8, 7], network anomalies can be defined as patterns of interaction that significantly differ from the norm and in order to capture the appropriate patterns of interaction, specific aspects of entities' behavior are used (e.g. email analysis). Similar to this, in a SCADA system, individual entities demonstrate a quite different communication behavior when infected, in terms of mean packet generation frequency (traffic burst), protocol distribution or interaction pattern.

Discovering anomalies in the context of a network system is a challenging issue due to the complexity of the environment and the different nature of the induced attacks. Regarding node behavior related decisions it makes sense to ask more than one decision mechanism, since this practice assures a more trusted final decision. Ensemble systems of classifiers are widely used for intrusion detection in networks [5, 21]. These aim to include mutually complementary individual classifiers, which are characterized by high diversity in terms of classifier structure [24], internal parameters [11] or classifier inputs[13].

In real time systems, in addition to fast response and accuracy limited communication between detection modules is also desirable. By sending an explicit message for every anomaly detected, the intrusion detection mechanism will flood the medium with messages that will cause a delay in the communication between entities in the SCADA system. Moreover, since the detection mechanism needs to be sited at several locations in the SCADA system in order to recognize the intrusion near the origin, the communication overhead caused by the detection mechanism is further increased. One solution is the addition of a control channel, whereby these messages can be communicated without affecting the system's performance, but this is not always feasible. For this reason, an aggregation mechanism that groups initial alerts and sends a limited number of messages reporting the fault/intrusion accurately and on time is needed.

1.2. Contributions

The present article analyzes and evaluates the performance of a recently proposed intrusion detection mechanism, namely the *IT-OCSVM* [17], against the baseline OCSVM method. The mechanism uses a central OCSVM and a cluster of automatically produced ones, one for each source that induces significant traffic in the system, an embedded ensemble mechanism, social metrics, an aggregation method and a k-means clustering procedure that categorizes aggregated alerts. The mechanism runs in a distributed way and produces dedicated IDMEF (Intrusion Detection Message Exchange Format) messages that report the severity of the attack detected. The proposed mechanism is incorporated in a distributed IDS (intrusion detection system) communicating with other detection and management components of the system. *IT-OCSVM* is evaluated using datasets extracted from a testbed that mimics a small scale SCADA system under normal and malicious operation. The evaluation of the proposed method attests the superiority of the new structure in terms of accuracy, false alarm rate and system overhead.

The rest of this article is organized as follows: Section 2 presents the use of OCSVM in SCADA systems; Section 3 describes the IT-OCSVM method; Section 4 presents the simulation environment and results; an Section 5 concludes the article.

2. OCSVM for intrusion detection in SCADA systems

SCADA systems (Figure 1) that tie together decentralized facilities such as power, oil and gas pipelines as well as water distribution and wastewater collection systems, were designed to be open, robust, and easily operated and repaired, but not necessarily secure. Cyber-attacks against these systems are considered extremely dangerous for critical infrastructure (CI) operation and must be addressed in a specific way [27].

The one-class classification problem is a special case of the conventional two class classification problem, where only data from one specific class are available and well represented. This class is called the target class. Another class, which is called the outlier class, can be sampled very sparsely, or even not at all. This smaller class contains data that appear when the operation of the system varies from normal, due to a possible attack. OCSVM [10] possesses several advantages for processing SCADA environment data and automates SCADA performance monitoring, which can be highlighted as:
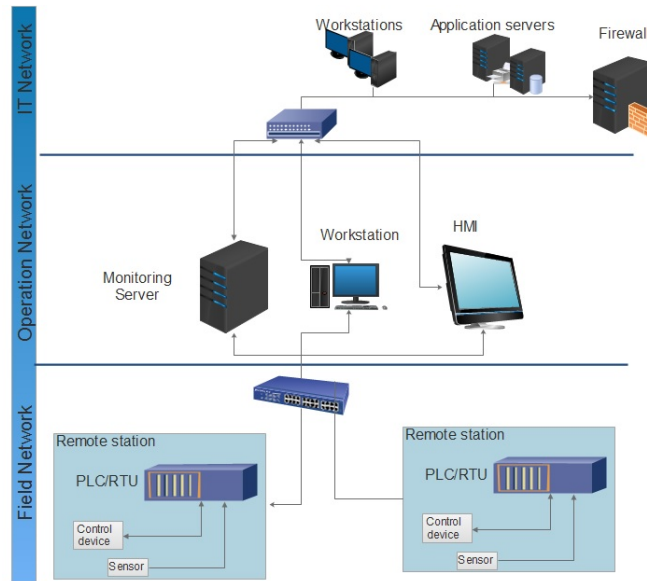
Figure 1: A typical SCADA system

- In the case of SCADA performance monitoring, which patterns in the data are normal or abnormal may not be obvious to operators. Since OCSVM does not require any signatures of data to build the detection model it is well suited for intrusion detection in SCADA environments.

- Since the detection mechanism does not require any prior information of the expected attack types, OCSVM is capable of detecting both known and unknown (novel) attacks.

- In practice, training data taken from SCADA environment could include noise Samples and most of the classification based intrusion detection methods are very sensitive to noise. However, the OCSVM detection approach is robust to noise samples in the training process.

- Algorithm configuration can be controlled by the user to regulate the percentage of anomalies expected.

- Due to the low computation time, OCSVM detectors can operate fast enough for on line SCADA performance monitoring.

- Typical monitoring of data of SCADA systems consists of several attributes and OCSVM is capable of handling multiple attributed data.

3. IT-OCSVM detection mechanism

The main purpose of the *IT-OCSVM* detection mechanism is to perform anomaly detection in a time-efficient way, with good accuracy and low overhead, within a temporal window adequate for the nature of SCADA systems. In order to achieve the aforementioned goals several operation stages need to be carried out.

- **Pre-processing** of raw input data in order to feed the *IT-OCSVM* module. The attributes in the raw data of the testbed contain all forms: continuous, discrete, and symbolic, with significantly varying resolution and ranges. Most pattern classification methods are not able to process data in such a format. Hence, pre-processing is required before pattern classification models can be built. This consists of two steps: the first involves mapping symbolic-valued attributes to numeric-valued attributes and second is implemented scaling. Different pre-processing techniques are used based on the characteristics of each feature type [15].

- **Selection of the most appropriate features** for the training of the *IT-OCSVM* model. These features are divided into content and time based features. Since the majority of DoS and probing attacks may use hundreds of packets, time-based features are mostly used.

- **Creation of cluster of OCSVM models** that are trained on discrete sources. There are many slow probing attacks that scan the hosts using much larger intervals, thus being able to merge into the overall traffic in the network. As a consequence, these attacks cannot be detected using derived time based features and in order to capture them, the raw data after arriving in the module is split into different datasets according to the sender of the packet. An OCSM module is created and trained for each split dataset. The cluster of split OCSVMs run in parallel with the central OCSVM and produce errors targeted to the specific source.

  It is important to mention that a split OCSVM is not created for each source, but only for those sources that produce high traffic in the network during the training period. In order to separate significant nodes

7

a threshold $P_{packets}$ is used and very source that produces a number of packets over this threshold during the training period is marked as a significant node. If during the testing period a source is showing big activity, while not being marked as significant, a medium alarm is fired for it.

- **Testing of the traffic dataset** that contains malicious attacks. Based on the models created from the training phase the new dataset is tested against normal patterns. Each OCSVM module returns a function f that takes the value $+1$ in a region capturing normal data points (i.e. for events implying normal behaviour of the SCADA system) and takes a negative value elsewhere (i.e. for events implying abnormal behaviour of the SCADA system).

- **Ensemble of classifiers** The initial outcomes of the different OCSVM modules are combined by the ensemble based mechanism that uses mean majority voting.

- **Social analysis** Social network analysis is executed based on the network traces and Spearman's rank correlation coefficient is used in order to add weight to alerts produced from different sources.

- **Fusion of the information** Due to the possible existence of multiple anomalies in a SCADA system, the outcomes of the different models are gathered and classified in terms of importance. This importance is derived through aggregation and k-means clustering of the outputs.

- **Communication of the mechanism**. In order to cooperate with other components of the IDS the mechanism sends IDMEF files. The created files describe the nature of the alert, in terms of importance, the position in the system, time etc.

3.1. Ensemble system

Ensemble methods can be differentiated according to the extent each classifier affects the others. This property indicates whether the classifiers are dependent or independent. The first situation occurs when the outcomes of a certain classifier affects the creation of the next [19] and the latter, when each classifier is built independently and their results are combined in some fashion [23]. Our proposed mechanism uses an independent ensemble mechanism.

Apart from the central OCSVM, which is trained on the entire dataset, a cluster of split OCSVMs is automatically created though the decomposition of the traffic dataset into disjoint subsets 2. The idea is based on the work of the authors in [3], who achieved high accuracy by building thousands of classifiers trained from small subsets of data in a distributed environment and through this decomposition high diversity of the methods is achieved. Diversity is an essential feature of an ensemble mechanism in order to achieve high accuracy [25].

Figure 2: Decomposition of the traffic dataset to disjoint subsets

A combination procedure is then applied so as to produce a single classification for a given instance. There exist many ensemble classification methods, e.g. majority voting, performance weighting, distribution summation an order statistics. These methods have some advantages and disadvantages in terms of accuracy and computational cost. In a real time system, both parameters are crucial and for this reason a simple but effective ensemble mechanism must be chosen [20]. In order to achieve a balance between these two parameters the outcomes of both the central and the split OCSVMs are combined using a simple algebraic weighted sum rule using Equation 1. The ensemble based mechanism is represented in Figure 3.

$$q_e(i,j) = \sum_{n=1}^{N} w_i d_t(i,j) \tag{1}$$

where, $d_t(i,j)$ is the outcome of each individual classifier n for sample data i that originates from node j and w_i is the weight given to each classifier.
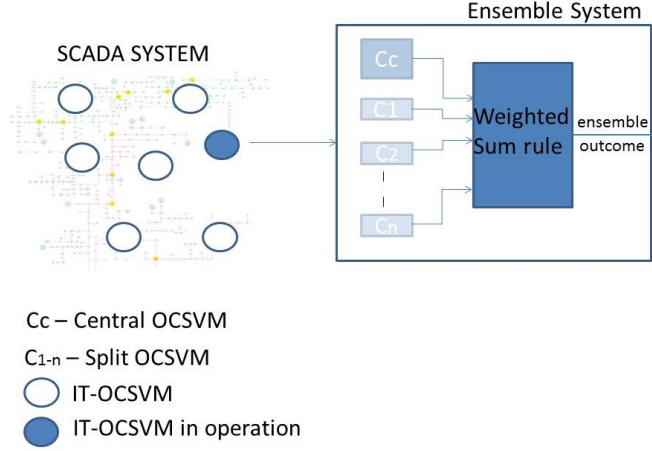
9

Figure 3: Ensemble based system

The values that are produced from this stage are weighted using the social analysis module, according to the Spearman's rank coefficient for every significant source. These values are then fed into the aggregation module.

3.2. Social analysis module

Deviation from the normal protocol operation in communication networks has received considerable attention from the research community in recent years. A malicious node may use the vulnerabilities of the system architecture to perform different kinds of attacks. Hence, for network reliability it necessary to develop an efficient technique to detect misbehaving clients in a timely manner and the correlation coefficients between entities can effectively detect malicious nodes [9].

In order to enhance the performance of the OCSVM module, parallel statistical algorithms are executed. For each significant source that is detected during training, a list of the protocols used is created, which are ranked and stored along with the IP/ MAC addresses of each source. During the testing phase the same procedure is executed, thus producing a list of the protocols used for each significant source. The two lists are compared using Spearman's correlation coefficient [22].

$$p = 1 - \frac{6 \sum d_i^2}{n(n^2 - 1)} \tag{2}$$

10

The final output is a number q_j for each source j that indicates whether there is a differentiation in the way that each source behaves during the training and testing period. The output of the method is a value between 1 and 0, with a number close to 1 indicating similar behaviour of the source both in the training and testing sessions. This value is used in order to give additional significance to alerts produced from the specific source from the OCSVM modules (see Equation 3. In Figure 4 an example of the protocols that are mainly used during normal and abnormal operation of a node is presented.

$$q_s(i,j) = \frac{q_e(i,j)}{p_j}, \forall\ q_e(i,j)\ with\ source\ node\ j \qquad (3)$$

Figure 4: Most used protocols used by a node during normal (left) and abnormal (right) operation

3.3. Fusion of alarms - final outputs

The OCSVM module produces one initial alarm for any deviation in network traffic from normal, based on the training models that it has created. In real time systems in addition to fast response and accuracy limited communication between the detection modules is desirable. For this reason we implement a fusion procedure which groups alerts per source node and gives final scores to aggregated alerts based on the initial values and the number of similar initial alerts.

The first stage of fusion consists of an aggregation mechanism that groups individual alerts according to their origin. Each alert i has an initial value q_i (and a source j) based on the procedures described in the previous sections 3.1,3.2. Using equations 4 each aggregated alert j is assigned two values qa_j and qb_j that represent its severity. This severity comes from both the sum of the values of the initial alerts and the number of attacks that originate from the same node.

$$qa_j = \sum_i q_s(i,j), \quad qb_j = \sum_i 1, \forall \ q_s(i,j) \ with \ source \ node \ j \qquad (4)$$

During the second stage of fusion the system uses K-means clustering so as to divide the alarms into: possible, medium and severe. The k-means clustering algorithm is one of the simplest and most commonly used clustering algorithms. It is a partitional algorithm that heuristically attempts to minimize the sum of squared errors.

$$SSE = \sum_{k=1}^{K} \sum_{j=1}^{N_k} ||qa_j - \mu_k||^2 \qquad (5)$$

where, N_k is the number of instances belonging to cluster k and μ_k is the mean of cluster k, calculated as the mean of all the instances belonging to the cluster i

$$\mu_{k,i} = \frac{1}{N_k} \sum_{q=1}^{N_k} qa_q, j \ \forall j \qquad (6)$$

The algorithm begins with an initial set of cluster centers and in each iteration each instance is assigned to its nearest cluster center according to the euclidean distance between the two. We use two k-means algorithms that run in parallel which partition alerts into two categories: possible and severe. These decisions are then combined according to the above equations and are assigned their final classification: possible, medium or severe, according to Figure 5.

| Aggregated alerts | K-means clustering (instances) | | | |
| K-means clustering (Values) | | | | |

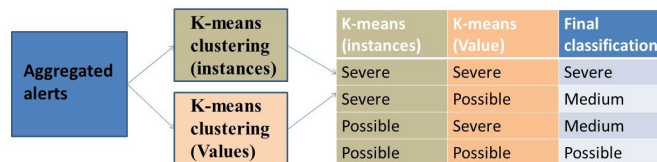| K-means (instances)	K-means (Value)	Final classification
Severe	Severe	Severe
Severe	Possible	Medium
Possible	Severe	Medium
Possible	Possible	Possible

Figure 5: K-means clustering module

Using the fusion procedure, the final alarms that the system produces are significantly decreased and all the sources with suspect behavior are reported. The whole procedure is described in Figure 6.
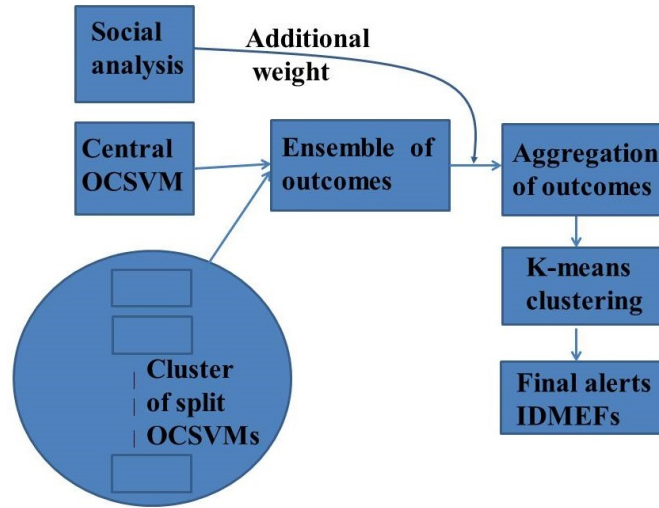
12

Figure 6: Architecture of detection mechanism

In order to cooperate with the other modules the *IT-OCSVM* mechanism is integrated in the *PID* system and communicates with the other modules created under the "Cockpit CI project" using IDMEF [6] files. The IDMEF defines the experimental standard for exchanging intrusion detection related events and 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. Knowledge of the source node where the intrusion originates is a very important feature an IDS system must have. Once the infected node is spotted the infection can be limited by the isolation of this node from the rest of the network. Fast and accurate detection of the source node of a contamination is crucial for the correct function of an IDS.

3.4. Nature of the trial

The trial is conducted off line with the use of two datasets extracted from the testbed (Figure 8. The testbed architecture mimics a small-scale SCADA system, comprising the operations and field networks, which include a Human-Machine Interface Station (for process monitoring), a managed switch (with port monitoring capabilities for network traffic capture), and

13

```
<?xml version="1.0" encoding="UTF-8"?>
- <idmef:IDMEF-Message version="1.0" xmlns:idmef="http://iana.org/idmef">
  - <idmef:Alert>
    - <idmef:Analyzer analyzerid="test">
      - <idmef:Node category="unknown">
          <idmef:location>IT Network</idmef:location>
          <idmef:name>OCSVM</idmef:name>
        </idmef:Node>
      </idmef:Analyzer>
      <idmef:CreateTime ntpstamp="0x1130fdd3.0xa0000000">2014-08-19T16:19:43+01:00</idmef:CreateTime>
    - <idmef:Source>
      - <idmef:Node>
        - <idmef:Address category="ipv4-addr">
            <idmef:address>172.27.224.32</idmef:address>
          </idmef:Address>
        </idmef:Node>
      </idmef:Source>
    - <idmef:Target>
      - <idmef:Node>
        - <idmef:Address category="ipv4-addr">
            <idmef:address>172.27.224.3</idmef:address>
          </idmef:Address>
        </idmef:Node>
      </idmef:Target>
      <idmef:Classification text="SEVERE ALARM"/>
    </idmef:Alert>
</idmef:IDMEF-Message>
```

Figure 7: Typical IDMEF message produced by the IT-OCSVM mechanism

two Programmable Logic Controller Units, for process control. The *NIDS* and *IT-OCSVM* modules are co-located on the same host, being able to intercept all the traffic flowing in the network.
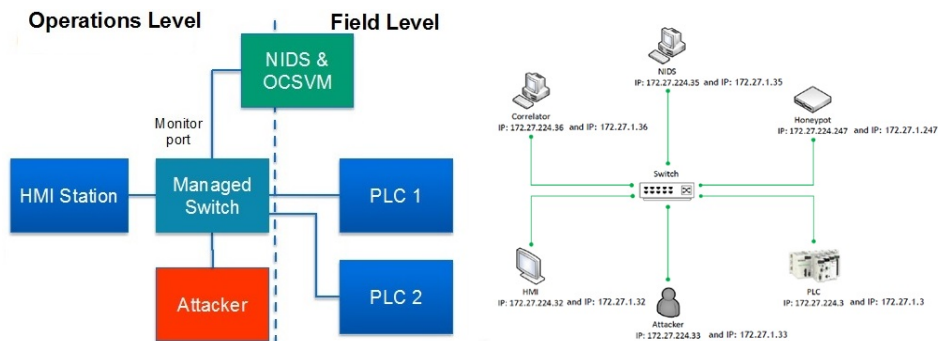

Figure 8: Architecture of the testbed

Three kinds of attacks are evaluated:

- **Network scan attack** In a 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 detect.

- **ARP cache spoofing - MITM attack ARP cache spoofing** is a technique where an attacker sends fake ARP messages. The aim is

14

to associate the attacker's MAC address with the IP address of another host, thus causing any traffic meant for that IP to be sent to the 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 refers to the situation where the attacker floods the connection with the PLC by sending SYN packets.

4. Results and analysis

Each OCSVM that is trained produces a separate model file which is used in order to classify new data as normal or malicious. All OCSVMs use Gaussian RBF Kernel functions with default parameters $\sigma = 0.01$ and $\nu = 0.001$. The parameters used during evaluation of the proposed IT-OCSVM detection method are shown in table 1.

Table 1: Evaluation parameters

$Indepedent Paramater$	$Default\ value$
ν	0.001
σ	0.01
$P_{packets}\ -\ (No\ of\ split\ OCSVMs)$	$\frac{1}{100}(5)$
$Ensemble\ mechanism$	$Weighted\ sum\ value$
$k-means$	$2\ stages$

During execution of the proposed detection mechanism, a separate file that contains information about the split sources is also created. This file is used in order to split the testing dataset according to the sources that are categorized as important during training. A snapshot of the sources file is shown in figure 9. For each row, the IP/MAC address of the source is followed by the 5 most used protocols during the training period, which are sorted in descending order. These protocols are compared with those used by the same source in the testing period in order to calculate the Spearman's correlation coefficient value for each separate source.

```
172.27.224.32,eth:ip:tcp:mbtcp:modbus,eth:ip:tcp, , ,
172.27.224.3,eth:ip:tcp:mbtcp:modbus,eth:ip:icmp:data, , ,
172.27.224.33,eth:ip:icmp:data, , , ,
10.3.3.28,eth:ip:udp:data, , , ,
d0:7e:28:8e:40:9b,eth:llc:stp, , , ,
```

Figure 9: Sources file created by the OCSVM module

4.1. Initial network traffic analysis

In Figure 10 the rate of the injected packets in the system during normal and abnormal operation of the SCADA system is monitored. In the lower part of the figure, we can observe that when the DOS attack takes place, the rate of the packets injected into the system is much higher compared to a normal operation period.
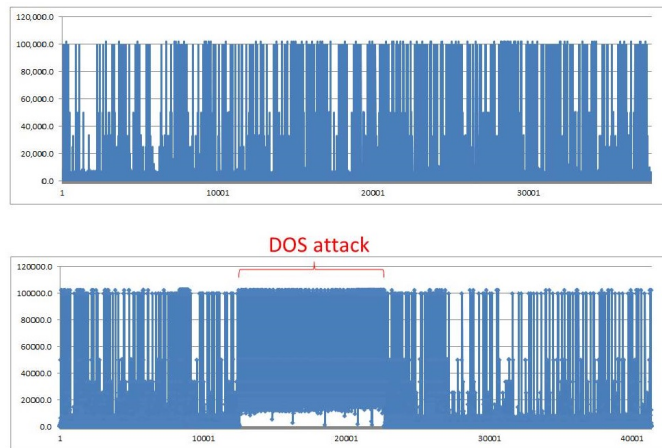

Figure 10: Rate of packets during normal and abnormal operation time

MITM and network scan attacks, on the other hand, do not have such profound consequences on the traffic rate, since they use few messages (MITM) or they merge with the overall traffic. As can be observed from Figure 11, an ARP spoofing attack can be identified by using feature 5 in the overall dataset. Apart from the actual attack there are also other instances where the central OCSVM would probably fire an alarm due to high values of this feature.

When subsets are used the occurrence of this attack is more obvious. In Figure 12 the values of feature 5 over time for two sources that induce
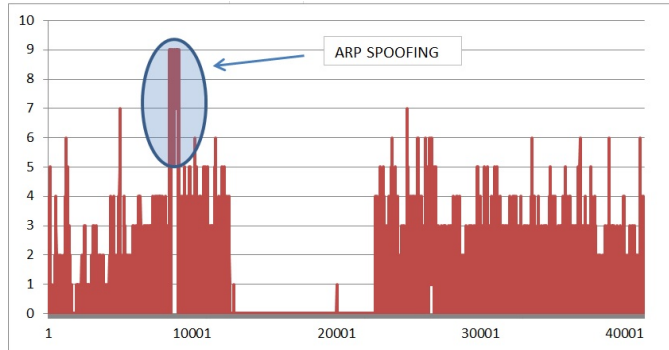
16

Figure 11: ARP spoofing attack - overall dataset

significant traffic in the system are shown. The left diagram presents the
number of ARP packets sent by the intruder during the testing period, while
the right presents the same distribution for a normal node of the system.
The difference in the performance of a node under attack is more evident
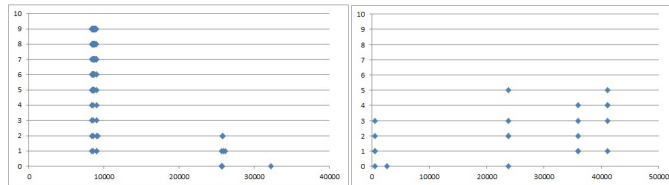using the split datasets.

Figure 12: ARP spoofing attack - split datasets

Moreover, the Spearman's rank correlation coefficient of the source that
executes a scan attack (Network scan or ARP scan) is heavily affected since
node behavior deviates from normal in terms of protocol distribution, thus
enhancing the detection capability of our mechanism.

4.2. Method evaluation

This subsection describes the performance of the proposed OCSVM based
intrusion detection algorithm for the simulated data. During the testing
period several attack scenarios are simulated in the testbed, which include:
a network scan, a network flood and an MITM attack. Since the attacks are

17

performed during different time periods we divide the testing dataset into several smaller ones, each containing a different attack. The testing data consists of normal and attack data and the composition of the data sets is 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**

The IT-OCSVM method, as shown in [17], performs well in terms of detection accuracy and false positive rate 2.

Table 2: Performance evaluation of the IT-OCSVM module

$Dataset$	DA	FAR
A	98.81%	1.18%
B	94.6%	3.25%
C	95.20%	1.51%
D	96.37%	2.3%
All	96.3%	2.5%

4.3. Ensemble mechanism. Parameter $P_{packets}$

The ensemble module affects the performance of the IT-OCSVM. Discrete OCVSMs that are created by the mechanism have an impact on the accuracy of the detection mechanism. Regarding which, the cluster of automatically produced OCSVMs can be significantly large(one OCSVM per source) or very small (a total of one or two OCSVMs) according to the value of the parameter $P_{packets}$. Figure 13 shows the relation between the value of the threshold parameter $P_{packets}$ and the number of created OCSVMs as well as how this affects the performance of the detection mechanism.
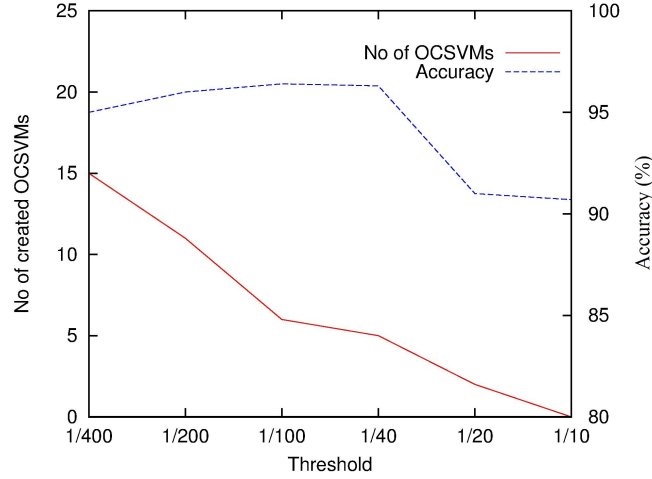
Figure 13: Number of automatically created OCSVMs against accuracy of the mechanism

The parameter $P_{packets}$ is given a value as a proportion of the number of rows of the entire training dataset. According to Figure 13, it is evident that an appropriate selection of parameter $P_{packets}$ is essential for the correct operation of the proposed mechanism. When the parameter is given a very big value then the created OCSVMs are too few to improve accuracy and the performance of the mechanism deteriorates. When this value is extremely big $\frac{1}{10}$ then the mechanism degrades to a simple OCSVM module. For values relative low the number of created OCSVMs grows without an improvement in systems accuracy. As observed in Figure 13, when the number of created OCSVMs is relatively large the accuracy of the mechanism drops. As split training sets are created for sources with no significant traffic in the training period, the OCSVMs that are created are not correctly trained and instead of helping they harass the decision procedure. For values of the threshold parameter between $\frac{1}{40}$ and $\frac{1}{100}$ the accuracy of the detection mechanism is optimal. In fact, for these values, the accuracy of the IT-OCSVM mechanism compared to the initial OCSVM is *6 %* better. That is, the simple OCSVM achieves accuracy of *90.7 %* while the proposed IT-OCSVM reaches *96.4%* (Figure 13). Apart from the profound improvement in terms of accuracy the proposed mechanism has other advantages compared to a simple OCSVM module. It creates a decreased number of alarms and also categorizes these

19

according to their severity, as described in the next subsection.

4.4. Impact of the fusion mechanism

The fusion of the alarms produced by the individual OCSVMs consists of two stages. The first is aggregation of alarms per source and the second is the clustering of them using a two stage k-means clustering algorithm, as described in section 3.3. The outcome of the fusion procedure is the deduction of the communicated alarms in the system and also their classification as: possible, medium and severe. In Table 3 the number of initial and final aggregated alarms is presented. It can be observed that the number of final alarms is significantly lower compared to the initial ones, thus reducing the communication costs that such a distributed mechanism have in the network.

Table 3: Aggregated alarms produced by IT-OCSVM are significantly decreased compared to the initial ones

Dataset	Initial alarms	Aggregated alarms
A	129	16
B	658	21
C	9273	18
D	203	16
All	10507	22

The IT-OCSVM system categorizes alarms according to the level of severity they have, with most being classified as possible and those few originating from real attacks in the system are termed severe (see Figure 14). Since the proposed mechanism is part of a distributed PIDS, the information sent by the IT-OCSVM can be combined with those sent by the other detection modules. For this reason this categorization of the alarms is important, if correct final decisions about the situation in the network are to be taken.

4.5. Computational cost and time overhead

The complexity of an IDS can be attributed to hardware, software and operational factors. For simplicity, it is usually estimated as the computing time required to perform classification of the dataset and output the final alarms. Increasing the number of classifiers usually increases the computational cost and decreases their comprehensibility. For this reason, special
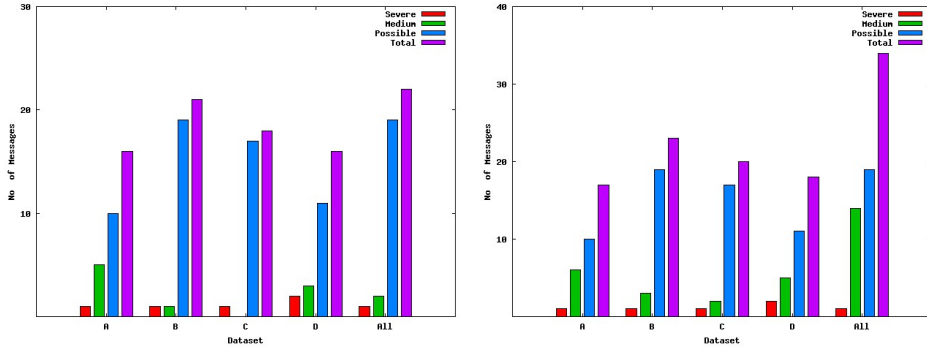
Figure 14: IT-OCSVM categorizes aggregated alarms. The left diagram shows aggregated alarms created by IT-OCSVM without the additional medium alarms and the right diagram illustrates all the final alarms created by the IT-OCSVM

care must be taken when choosing parameter $P_{packets}$. As mentioned in subsection 4.3, this parameter determines the number of created split datasets and thus, split OCSVMs. While the increase in number of split OCSVMs does not impose any significant increase in the method's performance, this may slow down the detection mechanism.

In figure 15, we illustrate the time performance of the method compared to a simple OCSVM. The evaluation was conducted on a PC with Intel core 2 duo 1.7 Mhz CPU, 2GB main memory, 80GB hard disk 7200 rpm hard disk and Microsoft windows 7 64bit.

According to Figure 15, the execution time of the proposed IT-OCSVM is bigger compared to a simple OCSVM method. However,or the extreme configuration where *15* OCSVMs are created the performance gap increases and the proposed IT-OCSVM is *75 %* slower than the simple OCSVM, whereas when the IT-OCSVM operates under the default configuration (*5-6* OCSVMs) the performance gap is *55 %*. Based on these observations we conclude that the system, under the default configuration, performs a classification in a comparable time to that of a simple OCSVM classifier and thus, it can be adopted in soft real-time applications.

We have to mention that the performance evaluation conducted in this subsection, does not include the time that each detection mechanism needs in order to create and disseminate IDMEF messages or the computational and time costs that the correlator needs in order to collect and analyze the alarms. It is evident that the OCSVM classifier, compared to the proposed
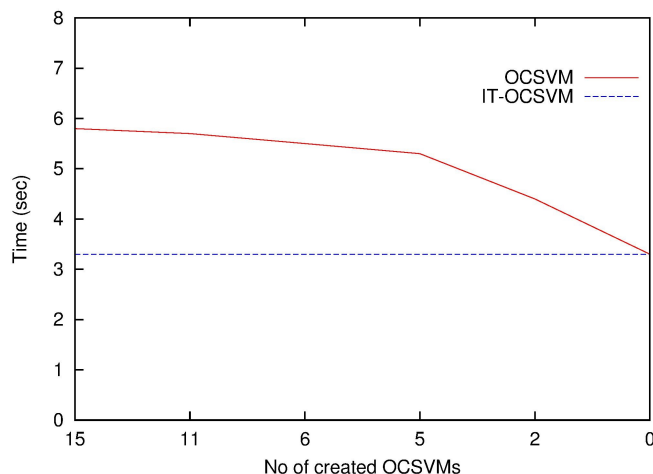
Figure 15: Approximate execution time for the entire testing dataset

IT-OCSVM, needs significant additional time in order to send all the detected alarms. Moreover, the categorization that is performed by the IT-OCSVM mechanism reduces the computational load of the correlator that collects the alarms from the distributed detection agents.

5. Conclusion

This article analyzes the IT-OCSVM mechanism and evaluates it against the baseline method for different attack scenarios. The detection mechanism, which runs in a distributed way, can be used in large SCADA networks with no additional modifications. The combination of social network analysis metrics with machine learning classification techniques, enhances the performance of the detection mechanism and improves accuracy for all the simulation scenarios investigated. Moreover, the aggregation procedure embedded in the proposed mechanism decreases the overhead of the IT-OCSVM and makes it easily incorporable in a soft real time system. That is, the method produces a small amount of final alerts and manages to detect all the simulated attacks.

In future work the proposed mechanism will be further enhanced in order to decrease false alarms and increase detection accuracy. It will be tested in a bigger hybrid testbed under different attack scenarios an other behav-

ior patterns will be also evaluated, e.g. the patterns of interaction among entities, thereby adding more sophistication to the detection mechanism.

[1] Balasubramaniyan, J. S., Garcia-Fernandez, J. O., Isacoff, D., Spafford, E., Zamboni, D., 1998. An architecture for intrusion detection using autonomous agents. In: Computer Security Applications Conference, 1998. Proceedings. 14th Annual. IEEE, pp. 13–24.

[2] Bennett, C., Highfill, D., Nov 2008. Networking ami smart meters. In: Energy 2030 Conference, 2008. ENERGY 2008. IEEE. pp. 1–8.

[3] Chawla, N. V., Hall, L. O., Bowyer, K. W., Kegelmeyer, W. P., 2004. Learning ensembles from bites: A scalable and accurate approach. The Journal of Machine Learning Research 5, 421–451.

[4] Crosbie, M., Spafford, E. H., 1995. Active defense of a computer system using autonomous agents.

[5] Curiac, D.-I., Volosencu, C., 2012. Ensemble based sensing anomaly detection in wireless sensor networks. Expert Systems with Applications 39 (10), 9087–9096.

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

[7] Garca-Teodoro, P., Daz-Verdejo, J., Maci-Fernndez, G., Vzquez, E., 2009. Anomaly-based network intrusion detection: Techniques, systems and challenges. Computers & Security 28 (12), 18 – 28.

[8] Gogoi, P., Bhattacharyya, D., Borah, B., Kalita, J. K., 2011. A survey of outlier detection methods in network anomaly identification. The Computer Journal 54 (4), 570–588.

[9] Hamid, M. A., Islam, M. S., Hong, C. S., 2008. Misbehavior detection in wireless mesh networks. In: Advanced Communication Technology, 2008. ICACT 2008. 10th International Conference on. Vol. 2. IEEE, pp. 1167–1169.

[10] Jiang, J., Yasakethu, L., 2013. 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, pp. 82–88.

[11] Kim, M.-J., Kang, D.-K., 2010. Ensemble with neural networks for bankruptcy prediction. Expert Systems with Applications 37 (4), 3373–3379.

[12] Kołaczek, G., Prusiewicz, A., 2011. Social network approach to anomaly detection in network systems. Intrusion Detection Systems.

[13] Krawczyk, B., Woźniak, M., 2014. Diversity measures for one-class classifier ensembles. Neurocomputing 126, 36–44.

[14] Kumar, S., 1995. Classification and detection of computer intrusions. Ph.D. thesis, Purdue University.

[15] Lazarevic, A., Ertöz, L., Kumar, V., Ozgur, A., Srivastava, J., 2003. A comparative study of anomaly detection schemes in network intrusion detection. In: SDM. SIAM, pp. 25–36.

[16] Maglaras, L., Jiang, J., 2014. Intrusion detection in scada systems using machine learning techniques. In: Proceedings of the SAI conference, 2014. pp. 626–631.

[17] Maglaras, L. A., Jiang, J., Cruz, T. J., December 2014. Integrated ocsvm mechanism for intrusion detection in scada systems. Electronics Letters 50, 1935–1936(1).

[18] McDaniel, P., McLaughlin, S., May 2009. Security and privacy challenges in the smart grid. Security Privacy, IEEE 7 (3), 75–77.

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

[20] Rokach, L., 2010. Pattern classification using ensemble methods. Vol. 75. World Scientific.

[21] Shieh, A. D., Kamm, D. F., 2009. Ensembles of one class support vector machines. In: Multiple Classifier Systems. Springer, pp. 181–190.

[22] Spearman, C., 1904. The proof and measurement of association between two things. The American journal of psychology 15 (1), 72–101.

[23] Tax, D. M., Duin, R. P., 2001. Combining one-class classifiers. In: Multiple Classifier Systems. Springer, pp. 299–308.

[24] Tsoumakas, G., Katakis, I., Vlahavas, I., 2004. Effective voting of heterogeneous classifiers. In: Machine Learning: ECML 2004. Springer, pp. 465–476.

[25] Tumer, K., Ghosh, J., 1996. Error correlation and error reduction in ensemble classifiers. Connection science 8 (3-4), 385–404.

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

[27] Zhu, B., Joseph, A., Sastry, S., 2011. 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, pp. 380–388.