Clustering extension of MOVICAB-IDS to distinguish intrusions in flow-based data

Raúl Sánchez[1], Emilio Corchado[2], and Álvaro Herrero[1]

[1]Department of Civil Engineering, University of Burgos, Spain
C/ Francisco de Vitoria s/n, 09006 Burgos, Spain
{ahcosio, rsarevalo}@ubu.es

[2]Departamento de Informática y Automática, Universidad de Salamanca
Plaza de la Merced, s/n, 37008 Salamanca, Spain
escorchado@usal.es

Abstract. Much effort has been devoted to research on intrusion detection in recent years because intrusion strategies and technologies are constantly and quickly evolving. As an innovative solution based on visualization, MOVICAB-IDS was previously proposed, conceived as a hybrid-intelligent Intrusion Detection System. It was designed to analyse continuous network data at a packet level and is extended in present paper for the analysis of flow-based traffic data. By incorporating clustering techniques to the original proposal, network flows are investigated trying to identify different types of attacks. The analysed real-life data (the well-known dataset from the University of Twente) come from a honeypot directly connected to the Internet (thus ensuring attack-exposure) and is analysed by means of clustering and neural techniques, individually and in conjunction. Promising results are obtained, proving the validity of the proposed extension for the analysis of network flow data.

Keywords: Network Intrusion Detection, Network Flow, Neural Projection, Clustering, MOVICAB-IDS.

1 Introduction

Intrusion Detection (ID) is a field that focuses on the identification of successful or ongoing attacks, in both networks and computers. The huge amount of previous work focused on network-based ID can be categorized by different criteria. One of them is the nature of the analysed data and, according to that, there are two categories based on the source of network data to be analysed: packets or flows. Some network Intrusion Detection Systems (IDSs) analyse every IP packet travelling along a network and then extract information from the different fields in the packet (headers mainly and payload sometimes). On the other hand, some IDSs deal with flows, being defined as "a set of IP packets passing on an observation point in the network during a certain time interval and having a set of common properties" [1]. In flow-based IDSs, rather than looking at all packets in a network, these tools look at aggregated information of related packets in the form of a flow, so the amount of data to be analysed is summarized and then reduced. With the rise of network speed and number and types of attacks, existing IDSs face challenges of capturing every single packet. Hence, a flow-based IDS has an overall lower amount of data to be processed, therefore it is the logical choice for high speed networks [2, 3].

MObile VIsualisation Connectionist Agent-Based IDS (MOVICAB-IDS) was proposed [4] as a novel IDS comprising a Hybrid Artificial Intelligent System. Its main goal was to apply an unsupervised neural projection model to extract traffic dataset projections and to display them through a mobile visualisation interface. One of its main drawbacks was its dependence on human processing as MOVICAB-IDS could not automatically raise the alarm when detecting an intrusion. Human users could fail to detect an intrusion even when displayed as an anomalous one, when visually processing big amounts of data [5]. Additionally, MOVICAB-IDS did not provide network administrators with much information to distinguish between the different kinds of attacks that a network may face. It is a desirable property of an IDS not only to detect an intrusion but also to identify the kind of intrusion that is detected in order to optimize the countermeasures to be applied. Taking into account the above described features, MOVICAB-IDS is being extended by the application of clustering techniques in conjunction with neural visualization, to overcome some of its limitations. Results on different combinations of such techniques have been obtained and are shown in section 4. Based on successful results obtained by upgrading MOVICAB-IDS with clustering techniques to detect different attacks on packet-based data [6], [7], present work focuses on flow-based data. Hence, present work proposes the combination of MOVICAB-IDS and different clustering techniques to analyse flow-based segments containing attack situations and released by the University of Twente [8]. The experimental study in present paper tries to know whether clustering could be more informative applied over the projected data rather than the original flow data captured from the network, not only to identify intrusions but also to differentiate the different kinds of malicious situations.

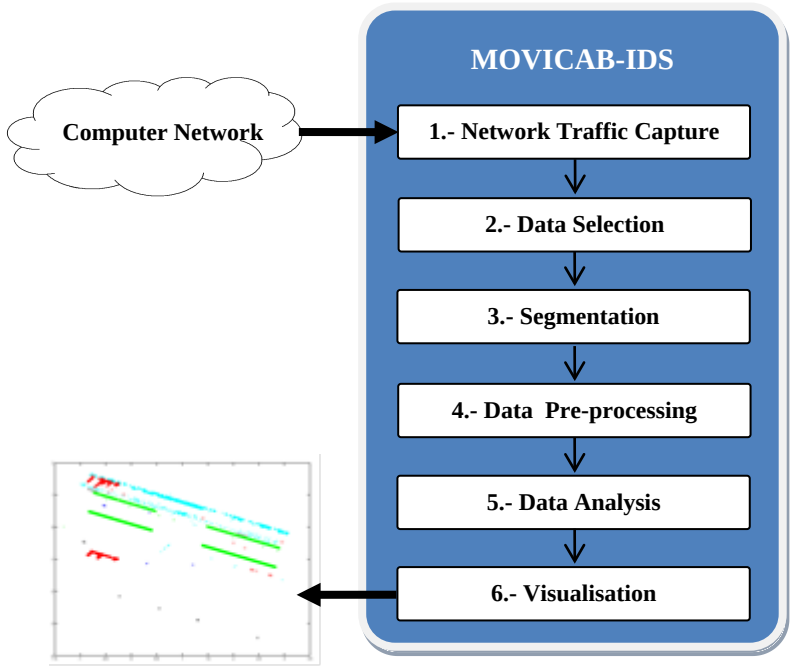
The remaining sections of this study are structured as follows: section 2 describes previous related work, while section 3 introduces the applied visualization and clustering techniques. Experiments and results are presented in section 4, and the conclusions of this study as well as future work are discussed in section 5.

2 Related Work

As previously stated, present work proposes extending the initial conception of MOVICAB-IDS to incorporate clustering techniques for flow analysis. This IDS was initially proposed [4] as a tool combining features extracted from packet headers to depict each simple packet by using neural unsupervised methods based on Exploratory Projection Pursuit (EPP) [9]. Then,

its architecture was improved as a Hybrid Intelligent System based on a Multi-agent System [5], that later on was bounded for real-time analysis [10]. The MOVICAB-IDS task organisation comprised the six tasks depicted in Fig. 1.

Fig. 1. Original MOVICAB-IDS task overview.



As it was already proposed for packet-based intrusion detection [6], the Data Analysis task is now performed by neural projection and clustering techniques, in order to obtain an enhanced visualization while at the same time, enabling automatic response. At the same time, for flow analysis, the three first tasks are adapted, according to details described in section 4.

Apart from previous work by paper authors, clustering has been previously applied to intrusion detection: [11] proposes an alert aggregation method, clustering similar alerts into a hyper alert based on category and feature similarity. From a similar perspective, [12] proposes a two-stage clustering algorithm to analyse the spatial and temporal relation of the network intrusion behaviours' alert sequence. [13] describes a classification of network traces through an improved nearest neighbour method, while [14] applies data mining algorithms for the same purpose and the results of preformatted data are visually displayed. Finally, [15] discusses on how the clustering algorithm is applied to intrusion detection and analyses an intrusion detection algorithm based on clustering problems. Differentiating from previous work, the approach proposed in present paper applies clustering to previously projected data (processed by neural models). The combination of clustering and neural visualization techniques have been previously [6] applied to the identification of different anomalous situations, but only to those related to the SNMP network protocol (network scans, MIB transfers and community string searches) and analysing packet information. In present paper, this combination of techniques is applied to network flows.

3 Proposed Clustering Extension

To better detect intrusions, an upgrade of MOVICAB-IDS, combining projection and clustering results is being proposed. The techniques combined within MOVICAB-IDS are described in this section.

3.1 Cooperative Maximum Likelihood Hebbian Learning

One neural implementation of EPP [9], applied under the frame of MOVICAB-IDS for the data analysis task, is Maximum Likelihood Hebbian Learning (MLHL) [16]. It identifies interestingness by maximising the probability of the residuals under specific probability density functions which are non-Gaussian.

An extended version of this model is the Cooperative Maximum Likelihood Hebbian Learning (CMLHL) [17] model. CMLHL is based on MLHL [16], adding lateral connections [17], which have been derived from the Rectified Gaussian Distribution [18]. The resultant net can find the independent factors of a data set but does so in a way that captures some type of global ordering in the data set.

Considering an N-dimensional input vector (x), and an M-dimensional output vector (y), with W_{ij} being the weight (linking input j to output i, ranging between the dimensionality of input and output vectors, respectively), then CMLHL can be expressed [17] as:

1. Feed-forward step:

$$y_i = \sum_{j=1}^{\mathbf{N}} W_{ij} x_j, \forall i \ . \quad \textbf{(1)}$$

2. Lateral activation passing:

$$y_i(t+1) = \left[y_i(t) + \tau(b - Ay) \right]^+ \ . \quad \textbf{(2)}$$

3. Feedback step:

$$e_j = x_j - \sum_{i=1}^{M} W_{ij} y_i, \forall j \ . \quad \textbf{(3)}$$

4. Weight change:

$$\Delta W_{ij} = \eta . y_i . sign(e_j) | e_j |^{p-1} \ . \quad \textbf{(4)}$$

Where: η is the learning rate, τ is the "strength" of the lateral connections, b the bias parameter, and p a parameter related to the energy function [17]. A is a symmetric matrix used to modify the response to the data [17], whose values range from -1 to 1. The effect of this matrix is based on the relation between the distances separating the output neurons.

3.2 Clustering

Cluster analysis [19], [20] consist in the organization of a collection of data items or patterns (usually represented as a vector of measurements, or a point in a multidimensional space) into clusters based on similarity. Hence, patterns within a valid cluster are more similar to each other than they are to a pattern belonging to a different cluster.

Pattern proximity is usually measured by a distance function defined on pairs of patterns. A variety of distance measures are in use in various communities [21], [22]. There are different approaches to data clustering [19], but given the high number and the strong diversity of the existent clustering methods, a representative technique for partitional as well as hierarchical clustering are applied in present study.

In general terms, there are two main types of clustering techniques: hierarchical and partitional approaches. Hierarchical methods produce a nested series of partitions (illustrated on a dendrogram which is a tree diagram) based on a similarity for merging or splitting clusters, while partitional methods identify the partition that optimizes (usually locally) a clustering criterion. Hence, obtaining a hierarchy of clusters can provide more flexibility than other methods. A partition of the data can be obtained from a hierarchy by cutting the tree of clusters at certain level.

Hierarchical methods generally fall into two types:

1. Agglomerative: an agglomerative approach begins with each pattern in a distinct cluster, and successively joins clusters together until a stopping criterion is satisfied or until a single cluster is formed.
2. Divisive: a divisive method begins with all patterns in a single cluster and performs splitting until a stopping criterion is met or every pattern is in a different cluster. This method is neither applied nor discussed in this paper.

Partitional clustering aims to directly obtain a single partition of the data instead of a clustering structure, such as the dendrogram produced by a hierarchical technique. Many of these methods are based on the iterative optimization of a criterion function that reflects the similarity between a new data and each one of the initial patterns selected for a specific iteration. Partitional methods have advantages in applications involving large data sets for which the construction of a dendrogram is computationally prohibitive. The problem of these algorithms is the need of the number of desired output clusters. Exhaustive search over all the set of possible initial labelling for an optimum output is clearly computationally prohibitive. Therefore, in practice, the algorithm is typically run a number of times with different starting states, and the best configuration obtained from all of the runs is used as the output clustering. Hence, we can meet different results depending on the initial labelling chosen (usually random). Additional techniques for the grouping operation include density-based [23], probabilistic [24], graph-theoretic [25] and mixture-resolving clustering methods, but they are not used on this paper.

In present study, one hierarchical (Agglomerative) and a partitional (*k*-means) clustering methods are applied for comparison purposes.

As similarity is fundamental to the definition of a cluster, a measure of the similarity is essential to most clustering methods and it must be carefully chosen. Present study applies well-known distance criteria used for examples whose features are all continuous when applying the *k*-means algorithm:

- sqEuclidean: squared Euclidean distance. Each centroid is the mean of the points in that cluster.
- Cityblock: sum of absolute differences. Each centroid is the component-wise median of the points in that cluster.
- Cosine: one minus the cosine of the included angle between points (treated as vectors). Each centroid is the mean of the points in that cluster, after normalizing those points to unit Euclidean length.
- Correlation: one minus the sample correlation between points (treated as sequences of values). Each centroid is the component-wise mean of the points in that cluster, after centering and normalizing those points to zero mean and unit standard deviation.

In the case of agglomerative clustering, a variety of linking methods can be designed. In present study, the following ones are applied:

- Single: shortest distance.
- Complete: furthest distance.
- Ward: inner squared distance (minimum variance algorithm), appropriate for Euclidean distances only.
- Median: weighted center of mass distance (WPGMC: Weighted Pair Group Method with Centroid Averaging), appropriate for Euclidean distances only.

- Average: unweighted average distance (UPGMA: Unweighted Pair Group Method with Arithmetic Averaging).
- Centroid: centroid distance (UPGMC: Unweighted Pair Group Method with Centroid Averaging), appropriate for Euclidean distances only.
- Weighted: weighted average distance (WPGMA: Weighted Pair Group Method with Arithmetic Averaging).

Additionally, clustering validation techniques have been applied in present study for the *k*-means algorithm. Such techniques evaluate the goodness of clustering results [26] by taking into account a certain criterion. The following criteria have been applied in present work, for comparison purposes:

- Calinski-Harabasz Index [27]: it evaluates the cluster validity based on the average between-and within-cluster sum of squares. Index measures separation based on the maximum distance between cluster centers, and measures compactness based on the sum of distances between objects and their cluster center.
- Silhouette Index [28]: it validates the clustering performance based on the pairwise difference of between and within cluster distances. In addition, the optimal cluster number is determined by maximizing the value of this index.
- Gap criterion [29]: it uses the output of any clustering algorithm, comparing the change in within-cluster dispersion with that expected under an appropriate reference null distribution. This index is especially useful on well-separated clusters and when used with a uniform reference distribution in the principal component orientation.

4 Experimental Study

As previously stated, neural projection and clustering techniques are applied to analyse flow-based data. To do so, two different alternatives are considered: clustering on original (flow) data and clustering on projected (reduced to 3 dimensions by CMLHL) data. This section describes the dataset used for evaluating these alternatives, the experimental settings and the obtained results.

4.1 Dataset

The analysed dataset contains flow-based information (14.2 M flows) from traffic collected by the University of Twente [8] using a honeypot in September 2008. A honeypot can be defined as an "environment where vulnerabilities have been deliberately introduced to observe attacks and intrusions" [30]. The honeypot was installed on a virtual machine directly connected to the Internet (ensuring traffic to be realistic) and ran several typical network services, such as:

- **ssh:** the OpenSSH service running on Debian was patched to track active hacking activities by logging sessions: for each login, the transcript (user typed commands) and the timing of the session was recorded.
- **Apache web server:** a simple webpage with a login form.
- **ftp:** proftp that uses the auth/ident service was chosen for additional authentication information about incoming connections.

The monitoring window comprised both working days and weekend days. The data collection resulted in a 24 GB dump file containing 155.2 M packets. Among the services, the most often contacted ones are ssh and http. The majority of the attacks targeted the ssh service and they can be divided into two categories: the automated and the manual ones. The first ones are well-known automated brute force scans, where a program enumerates usernames and passwords from large dictionary files. This attack is particularly easy to observe at flow level, since it generates a new flow for each connection. Attacks of the second type are manual connection attempts, amounting to 28 in the trace and 20 of them succeed.

The http alerts labelled in the data set are automated attacks that try to compromise the service by executing a scripted series of connections. No manual http attacks are present in the dataset. The types of flows labelled on the database are: ssh_scan, ssh_conn, ftp_scan, ftp_conn, http_scan, http_conn, authident_sideeffect, irc_sideeffect, icmp_sideeffect.
Regarding the ftp traffic, the data set contains only 6 connections to this service on the honeypot, during which an ftp session has been opened and immediately closed.

Original data have been split in different overlapping segments, as MOVICAB-IDS usually do with network traffic (see Section 2). As a result, several segments obtained from this dataset have been generated and analysed. Every segment contains all the flows whose timestamp is between the segment initial and final time limit. Segment length is stated as 782 seconds to cover the whole database, whose length is 539,520 seconds (from the beginning of the first flow to the end of the last one), that results in 709 segments. As defined for MOVICAB-IDS, there is a slight time overlap of 10 seconds between each pair of consecutive segments.

Seven out of the 709 generated segments have been chosen for present study. The selection of the segments was made by taking into account the minimum number of flows present with a specific number of types of data (attacks) in the segment, trying to cover all types of attacks included in the dataset. As a result, the segments described in table 1 were selected.

Table 1. Analysed segments from the "University of Twente" dataset.

# Segment	# Attack Types	Attack Types	# Flows
59	2	ssh_conn and irc_sideeffect	1,215
545	2	ssh_conn and http_conn	731
30	3	ssh_conn, ftp_conn and irc_sideeffect	12,172
58	3	ssh_conn, http_conn and irc_sideeffect	1,213
1	3	ssh_conn, irc_sideeffect and icmp_sideeffect	38,806
107	4	ssh_conn, authident_ sideeffect, irc_sideeffect and icmp_sideeffect	19,061
131	4	ssh_conn, http_conn, irc_sideeffect and icmp_sideeffect	122,274

In the case of segment 131, only results of *k*-means clustering algorithm could be obtained due to the large amount of memory that agglomerative clustering require over segments with such amount of data. Although there were some other segments including four types of attacks, those segments were not selected as they include a greater number of flows preventing them to be analysed by agglomerative clustering in present study.

For each one of the flows in the above described segments, the following fourteen features were extracted:

- **id**: the ID of the flow.
- **src_ip**: anonymized source IP address (encoded as 32-bit number).
- **dst_ip**: anonymized destination IP address (encoded as 32-bit number).
- **packets**: number of packets in the flow.
- **octets**: number of bytes in the flow.
- **start_time**: UNIX start time (number of seconds).
- **start_msec**: start time (milliseconds part).
- **end_time**: UNIX end time (number of seconds).
- **end_msec**: end time (milliseconds part).
- **src_port**: source port number.
- **dst_port**: destination port number.
- **tcp_flags:** TCP flags obtained by ORing the TCP flags field of all packets of the flow.
- **prot:** IP protocol number.
- **type:** alert type.

Two of the above listed features are not provided to the models: the first one (added to identify each single flow) and the last one (alert type).

This set of features has been processed in order to summarize the four features related to time (start_time, start_msec, end_time, and end_msec), being joined in only one feature, named as flow_length. By doing so, new datasets have been generated, comprising 9 features. The analysis has been done in both data (14 features and 9 features) for segments 59 an 545. All the data sets (detailed time information vs. flow length) have been studied but, as the results are pretty similar, only those for the 9 features data sets are shown in present paper, and the subsequent analysis for the rest of the segments has been done only on the 9-features processed data sets.

4.2 Results

The best results obtained by applying the previously introduced techniques to the described datasets are shown in the following subsections. The results are projected through CMLHL and further information about the clustering results is added to the projections, mainly by the glyph metaphor (different colours and symbols). The figures comprise a legend that states the colour and symbol used to depict each flow, according to the original data type (attack). Tables 3, 5, 7, and 9 show results obtained by applying *k*-means to different segments. In these tables, the "Replicates" column shows the number of times the clustering was repeated using new initial cluster centroid positions, and "Sum of Distances" shows the local minimum solution after all iterations on each replication. As the honeypot is designed to only capture "unexpected" traffic, it can be assumed that most of the recorded traffic is at least suspicious. As the majority of the attacks targeted the ssh service (ssh_conn type), for the comparison of clustering techniques it has been used as the base type of attack to show the clustering results. Tables below show the percentage of ssh_conn flows that are assigned to the same cluster (SSH_C) and the percentage of flows from other types that are assigned to a cluster containing ssh_conn flows (SSH_W). As previously mentioned, the results for the original segments 59 and 545 are very similar so, the subsequent analysis of the selected segments has been only applied to the (flow length) segments.

Apart from clustering results, visualizations from the analysis of the seven segments by applying both clustering techniques (k-means and agglomerative clustering) over the original (flow length) segment, together with the projection of the same segment obtained by CMLHL are shown below. In the case of the visualization of *k*-means results on original data (Figures 2.b, 4.b, 6.b and 8.b), only the two first dimensions of the data are depicted in order to easily obtain a 2D visualization of the flows.

Results on Cluster Evaluation

As an initial experiment, cluster evaluation has been applied to estimate the optimal number of cluster for *k*-means by using Calinski-Harabasz, Silhouette and Gap criteria. Both original (9 features) and projected (by CMLHL) segments have been studied by setting 10 as the maximum *k* value; it is high enough as the number of different attack types ranges from 2 to 4. Results are shown in Table 2, including the *k* value estimated for original data (*k*-od) and for projected data (*k*-pd).

Table 2. Cluster evaluation results by using different criteria.

# Segment	# Attack Types	Calinski-Harabasz		Silhouette		Gap	
		***k*-od**	***k*-pd**	***k*-od**	***k*-pd**	***k*-od**	***k*-pd**
59	2	10	9	8	2	10	10
545	2	10	10	4	2	10	10
30	3	10	5	10	5	10	10
58	3	10	8	10	2	10	9
1	3	10	9	10	10	10	10
107	4	10	10	10	7	10	10
131	4	10	9	10	10	-	-

From results shown in Table 2, it can be concluded that cluster evaluation on projected data obtains equal or lower (in most cases) optimal values of *k* than original data for all the criteria. As these values are closer to the number of attacks in the dataset than those obtained on original data (10 in all cases for Calinski-Harabasz and Gap criteria), it supports one of the main ideas of present study: intrusion analysis on neural projected data could outperform the analysis on original data. The Silhouette criterion is the only one suggesting a *k* value smaller than 10 for the segments with the minimum number of attacks (50 and 545). This criterion is the only one that, for these same segments (projected data), suggests a *k* value similar to the number of attack types. At the same time, there is only case (Silhouette criterion, segment 58 and projected data) where the suggested value of *k* is smaller than the number of attack types. The missing values for segment 131 and Gap criterion are due to the large amount of data present on this segment which prevented from completing such experiments.

After cluster evaluation, agglomerative techniques were applied to the seven selected datasets. For the sake of brevity, only results for some of the following segments are shown:

- **Segment 545**: selected as a representative of the segments comprising 2 types of attacks due to its reduced size (it is the smallest analysed segment).
- **Segment 30**: selected as a representative of the segments comprising 3 types of attacks due to the presence of attack type ftp_conn, that is not present in any other of the analysed segments.
- **Segments 107 and 131**: these are the segments comprising 4 types of attacks. As only k-means results are shown for segment 131 (biggest analysed segment), results on segment 107 are also shown to compare the different clustering techniques.

Results on Segment 545

Table. 3 shows the clustering results obtained by *k*-means on segment 545 (both projected and original), as well as the different values of parameters.

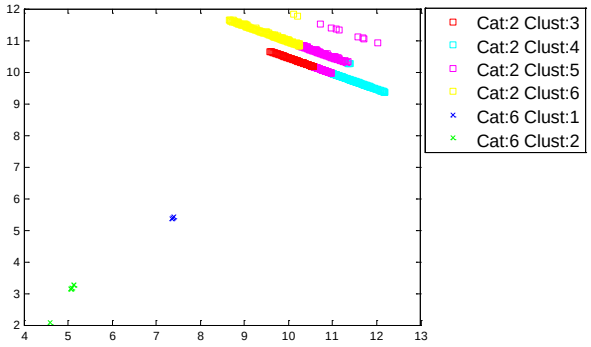
Table 3. *k*-means results on segment 545.

Data	*k*	Distance Criteria	SSH_C	SSH_W	Replicates/ Iterations	Sum of Distances
Projected	2	sqEuclidean	42.4863 %	0 %	5/12	1710.13
Original	2	sqEuclidean	49.1803 %	0.8197 %	5/2	3.65729E+17
Projected	4	sqEuclidean	15.9836 %	0 %	5/21	854.837
Original	4	sqEuclidean	0 %	0 %	5/3	3.88519E+10
Projected	6	sqEuclidean	0 %	0 %	5/20	121.144
Original	6	sqEuclidean	0 %	0 %	5/12	1.13974E+10
Projected	2	Cityblock	46.7213 %	0 %	5/8	814.403
Original	2	Cityblock	49.1803 %	0.8197 %	5/2	2.11713E+09
Projected	4	Cityblock	11.7486 %	0 %	5/21	593.5
Original	4	Cityblock	22.2678 %	0 %	5/14	1.05984E+09
Projected	6	Cityblock	0 %	0 %	5/16	402.065
Original	6	Cityblock	10.5191 %	0 %	5/12	1.0584E+09
Projected	2	Cosine	51.3661 %	0 %	5/9	1.37952
Original	2	Cosine	49.1803 %	0.8197 %	5/2	0.00345234
Projected	4	Cosine	25.1366 %	0 %	5/36	0.259027
Original	4	Cosine	0 %	0 %	5/3	1.04725E-09
Projected	6	Cosine	0 %	0 %	5/26	0.105203
Original	6	Cosine	0 %	0 %	5/12	3.07186E-10
Projected	2	Correlation	55.6011 %	0 %	5/8	53.3814
Original	2	Correlation	49.1803 %	0.8197 %	5/2	0.00437153
Projected	4	Correlation	24.8634 %	0 %	5/12	19.5017
Original	4	Correlation	0 %	0 %	5/3	1.14901E-09
Projected	6	Correlation	17.4863 %	0 %	5/27	13.6746
Original	6	Correlation	0 %	0 %	5/10	3.36364E-10

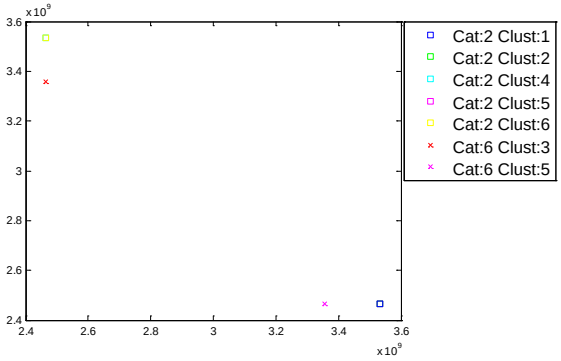
As a complementary result, Fig. 2 shows one of the experiments from Table 3, both projected (Fig. 2.a) and original (Fig. 2.b) data for *k*=6 and cityblok distance criterion. The data has been labelled as follows, according to the type of attack: ssh_conn flows (Cat. 2) and http_conn (Cat. 6)..

Fig. 2. Visualization of *k*-means results on segment 545 (*k*=6 and cityblock distance).

2.a *k*-means on projected data.



2.b *k*-means on original data.



In the case of the results shown in Fig. 2, clustering on original data obtained a non-zero (but low) SSH_C Rate value, while projected data obtained no error. For the ssh_conn flows on projected data (Cat. 2 in Fig. 2.a), the clustering technique groups data with no errors, even though the number of clusters (*k* parameter) is higher than the number of attack types, hence some clusters group only data from the same category. Additionally, the Sum of Distances is lower for the projected data, as the dimensionality of the data has been previously reduced through CMLHL.

The experiments results (with no error) obtained by the agglomerative method on the same segment (545) are shown in Table 4.

Table 4. Agglomerative clustering results on segment 545.

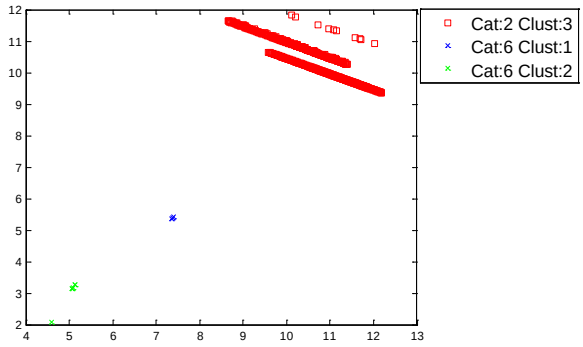
Data	Distance	Linkage	Cutoff	# Clusters	SSH_C
Projected	Euclidean	Single	8	3	0 %
Projected	sEuclidean	Complete	8	3	0 %
Projected	Cityblock	Average	12	3	0 %
Projected	Minkowski p=3	Weighted	8	3	0 %
Projected	Chebychev	Single	5	3	0 %
Projected	Mahalanobis	Single	6	3	0 %
Projected	Cosine	Complete	0.08	3	0 %
Projected	Correlation	Average	0.05	8	0 %
Original	Euclidean	Single	$1x10^8$	4	0 %
Original	Cityblock	Complete	$1x10^8$	4	0 %
Original	Minkowski p=3	Average	$1x10^8$	4	0 %
Original	Chebychev	Weighted	$1x10^8$	4	0 %
Original	Cosine	Single	0.0001	4	0 %
Original	Correlation	Complete	0.0001	4	0 %

It can be seen that, in the case of projected data, the minimum number of clusters without error is 3, while in the case of original data it is 4, with appropriate distance method. In the case of original data, the sEuclidean distance and Mahalanobis distance cannot be applied because the maximum recursion level has been reached in the first case, and the covariance matrix cannot be computed in the second case.

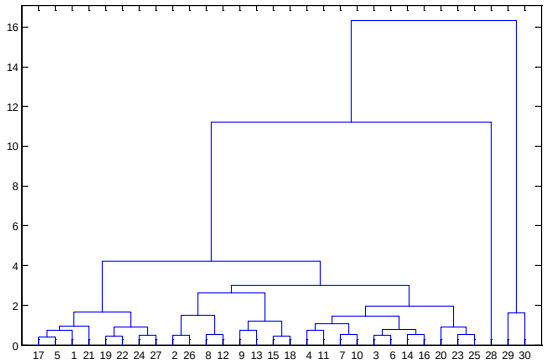
Results of one of the best experiments from Table 4 are depicted in Fig. 3, including segment visualization and the associated dendrogram on projected data. The parameters of the shown result are: sEuclidean distance, Complete linkage, cutoff: 8 and 3 groups with no clustering error.

Fig. 3. Visualization of best results by agglomerative clustering on segment 545.

3.a Agglomerative clustering on projected data.



3.b Corresponding dendrogram.



Results on Segment 30

Table. 5 shows the clustering results obtained by *k*-means on segment 30 (both projected and original), as well as the different values of parameters.

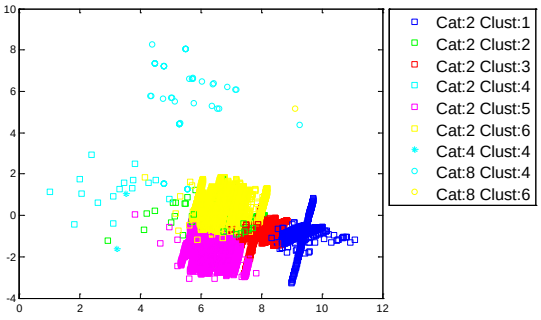
Table 5. *k*-means results on segment 30.

Data	k	Distance Criteria	SSH_C	SSH_W	Replicates/ Iterations	Sum of Distances
Projected	3	sqEuclidean	74.1358 %	0.0164 %	5/33	20480
Original	3	sqEuclidean	49.7414 %	0.1642 %	5/3	5.66802E+019
Projected	6	sqEuclidean	14.4922 %	0 %	5/39	10625.4
Original	6	sqEuclidean	49.7414 %	0 %	5/6	1.15164E+018
Projected	9	sqEuclidean	16.9308 %	0 %	5/33	7869.1
Original	9	sqEuclidean	0 %	0 %	5/7	7.6726E+016
Projected	3	Cityblock	45.3650 %	0.0246 %	5/12	18742.9
Original	3	Cityblock	65.4816 %	0 %	5/3	7.76029E+010
Projected	6	Cityblock	58.0507 %	0 %	5/25	13362.4
Original	6	Cityblock	31.4722 %	0 %	5/5	7.74968E+010
Projected	9	Cityblock	23.7786 %	0 %	5/31	11649.4
Original	9	Cityblock	20.3712 %	0 %	5/6	7.74823E+010
Projected	3	Cosine	77.2888 %	0 %	5/17	44.6866
Original	3	Cosine	49.7496 %	0.1642 %	5/3	0.627461
Projected	6	Cosine	26.8906 %	0 %	5/20	21.194
Original	6	Cosine	49.7414 %	0 %	5/5	0.00966831
Projected	9	Cosine	14.3033 %	0 %	5/37	15.1101
Original	9	Cosine	0 %	0 %	5/5	0.00069063
Projected	3	Correlation	63.6259 %	0 %	5/10	49.0796
Original	3	Correlation	49.7414 %	0.1642 %	5/3	0.747774
Projected	6	Correlation	52.3195 %	0 %	5/35	25.0537
Original	6	Correlation	0 %	0 %	5/3	0.000873173
Projected	9	Correlation	35.6269 %	0 %	5/44	10.158
Original	9	Correlation	0 %	0 %	5/4	0.000873149

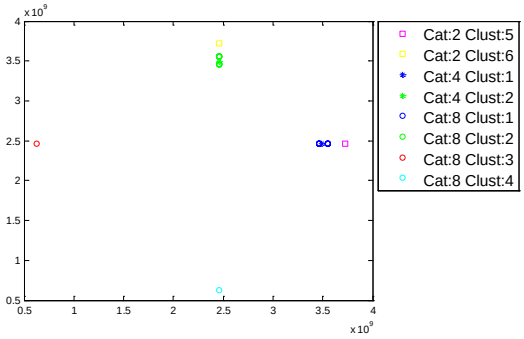
As a complementary result, Fig. 4 shows one of the experiments from Table 5, both projected (Fig. 4.a) and original (Fig. 4.b) data for *k*=6 and sqEuclidean distance criterion. The data has been labelled as follows: ssh_conn flows (Cat. 2), ftp_conn (Cat. 4) and irc_sideeffect (Cat. 8).

Fig. 4. Visualization of *k*-means results on segment 30 (*k*=6 and sqEuclidean distance).

4.a *k*-means on projected data.



4.b *k*-means on original data.



Clustering on both original and projected data obtained a non-zero SSH_C Rate value, being lower in the case of projected data. For the ssh_conn flows on projected data (Cat. 2 in Fig. 4.a), the clustering technique distributed them in all clusters, even though the number of clusters (*k* parameter) is higher than the number of categories, hence it can be concluded that this clustering technique may not be appropriate for this type of data.

The experiments results obtained by the agglomerative method on the same segment (30) are shown in Table 6.

Table 6. Agglomerative clustering results on segment 30.

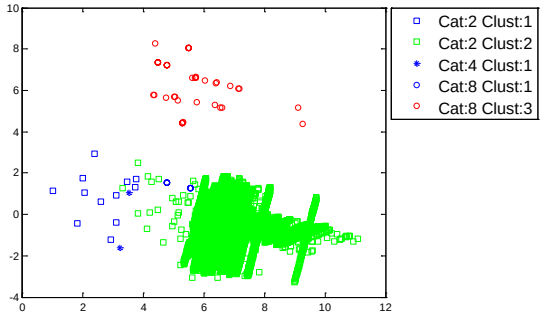
Data	Distance	Linkage	Cutoff	Cluster	SSH_C
Projected	Euclidean	Complete	9	4	0.1560 %
Projected	sEuclidean	Average	6	3	0.1806 %
Projected	Cityblock	Complete	12	3	0.1971 %
Projected	Minkowski p=3	Weighted	5	3	0.1806 %
Projected	Chebychev	Average	5	3	0.0985 %
Projected	Mahalanobis	Average	6	3	0.1067 %
Projected	Cosine	Average	0.1	5	0.1067 %
Original	Euclidean	Single	$1.5x10^8$	6	0 %
Original	Cityblock	Complete	$1.5x10^8$	6	0 %
Original	Minkowski p=3	Average	$1.5x10^8$	6	0 %
Original	Chebychev	Weighted	$1.5x10^8$	6	0 %
Original	Cosine	Single	0.0001	6	0 %
Original	Correlation	Complete	0.0001	6	0 %

It can be seen that, in the case of projected data, the minimum number of clusters is 3 (with a very low SSH_C rate) while in the case of original data it is 6, with appropriate distance method. In the case of original data, the sEuclidean and Mahalanobis distances cannot be applied because the maximum recursion level has been reached in the first case, and the covariance matrix cannot be computed in the second case.

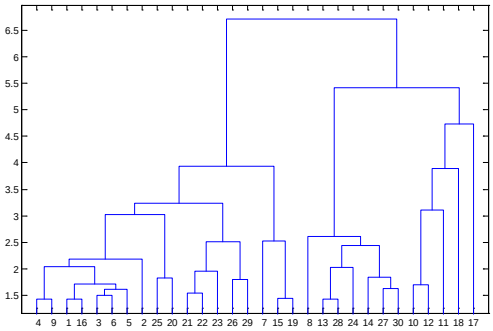
Results of one of the best experiments from Table 6 are depicted in Fig. 5, including segment visualization and the associated dendrogram on projected data. The parameters of the shown result are: Chebychev distance, Average linkage, cutoff: 5 and 3 groups with an SSH_C rate of 0.0985 %.

Fig. **5.** Visualization of best results by agglomerative clustering on segment 30.

5.a Agglomerative clustering on projected data.



5.b Corresponding dendrogram.



Results on Segment 107

Table 7 shows the clustering results obtained by *k*-means on segment 107 (both projected and original data), as well as the different values of parameters.

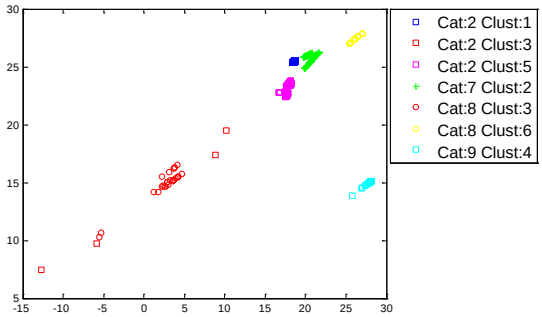
Table 7. *k*-means results on segment 107.

Data	k	Distance Criteria	SSH_C	SSH_W	Replicates/ Iterations	Sum of Distances
Projected	4	sqEuclidean	8.3255 %	0.4617 %	5/5	6611.9
Original	4	sqEuclidean	98.7777 %	0 %	5/4	2.28156E+019
Projected	6	sqEuclidean	0.0210 %	0 %	5/11	1697.64
Original	6	sqEuclidean	6.7044 %	0.2308 %	5/5	3.49696E+019
Projected	8	sqEuclidean	0.0210 %	0 %	5/20	1262.85
Original	8	sqEuclidean	49.2655 %	0 %	5/15	3.49696E+019
Projected	4	Cityblock	52.7594 %	0 %	5/10	3807.32
Original	4	Cityblock	37.3308 %	0.0787 %	5/26	8.99686E+010
Projected	6	Cityblock	47.8858 %	0 %	5/40	5543.52
Original	6	Cityblock	55.8021 %	0 %	5/32	8.99597E+010
Projected	8	Cityblock	34.1412 %	0 %	5/27	5456.32
Original	8	Cityblock	52.5496 %	0 %	5/32	7.07776E+010
Projected	4	Cosine	8.3202 %	0.4617 %	5/10	1.24694
Original	4	Cosine	6.7149 %	0.2308 %	5/5	1.06286
Projected	6	Cosine	8.3202 %	0 %	5/24	1.19717
Original	6	Cosine	6.7149 %	0 %	5/11	1.06286
Projected	8	Cosine	8.3202 %	0 %	5/43	1.19585
Original	8	Cosine	6.7149 %	0 %	5/16	1.06286
Projected	4	Correlation	8.3097 %	0.3095 %	5/8	4.94807
Original	4	Correlation	6.7149 %	0.2308 %	5/4	1.27183
Projected	6	Correlation	13.2567 %	0 %	5/25	138.004
Original	6	Correlation	6.7149 %	0 %	5/10	1.27183
Projected	8	Correlation	13.2514 %	0 %	5/ 27	4.82337
Original	8	Correlation	6.7149 %	0 %	5/17	1.27183

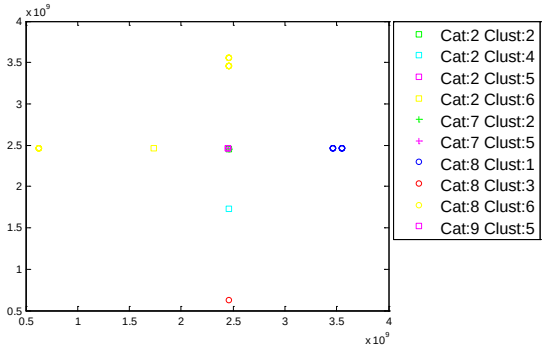
As a complementary result, Fig. 6 shows one of the experiments from Table 7, both projected (Fig. 6.a) and original (Fig. 6.b) data for *k*=6 and sqEuclidean distance criterion. The data has been labelled as follows: ssh_conn flows (Cat. 2), authident_sideeffect (Cat. 7), irc_sideeffect (Cat. 8) and icmp_sideeffect (Cat. 9).

Fig. 6. Visualization of *k*-means results on segment 107 (*k*=6 and sqEuclidean distance).

6.a *k*-means on projected data.



6.b *k*-means on original data.



Although very low, clustering on original data obtained a non-zero SSH_C and SSH_W rates, while clustering on projected data obtained a zero SSH_W Rate and lower SSH_C rate. For the ssh_conn flows on projected data (Cat. 2 in Fig. 6.a), the clustering technique groups data with some of the flows from Cat. 8. This clustering technique groups Cat. 7 and Cat. 9 in different clusters.

The experiments results obtained by the agglomerative method on the same segment (107) are shown in Table 8.

Table 8. Agglomerative clustering results on segment 107.

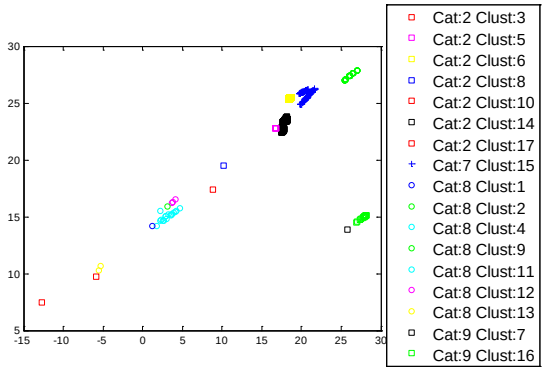
Data	Distance	Linkage	Cutoff	Cluster	SSH_C
Projected	Euclidean	Average	2	13	0.0052 %
Projected	sEuclidean	Weighted	2	13	0.0052 %
Projected	Cityblock	Complete	4	13	0.0052 %
Projected	Minkowski p=3	Weighted	2	12	0.0052 %
Projected	Chebychev	Single	0.5	17	0 %
Projected	Mahalanobis	Average	2	13	0.0052 %
Projected	Cosine	Average	0.0005	16	0 %
Projected	Correlation	Single	0.0001	14	0.0105 %
Original	Euclidean	Single	30000	16	6.6939 %
Original	Cityblock	Complete	80000	16	6.6939 %
Original	Minkowski p=3	Average	40000	16	6.6939 %
Original	Chebychev	Single	30000	16	6.6939 %
Original	Cosine	Weighted	$1x10^{-10}$	18	6.6939 %
Original	Correlation	Complete	$1x10^{-10}$	6	6.6939 %

From Table 8, it can be seen that in the case of projected data, the minimum number of clusters without error is 16 (it can be reduced to 12 clusters accepting a non-zero but very low SSH_C Rate), while in the case of original data it is 6 (with a higher SSH_C Rate), with appropriate distance method. Hence agglomerative clustering on projected data obtains better results. In the case of original data, the sEuclidean and Mahalanobis distances cannot be applied because the maximum recursion level has been reached in the first case, and the covariance matrix cannot be computed in the second case.

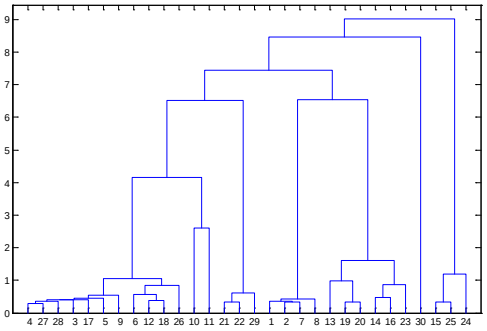
Results of one of the best experiments from Table 8 are depicted in Fig. 7, including segment visualization and the associated dendrogram on projected data. The parameters of the shown result are: Chebychev distance, Single linkage, cutoff: 0.5 and 17 groups with no clustering error.

Fig. **7.** Visualization of best results by agglomerative clustering on segment 107.

7.a Agglomerative clustering on projected data.



7.b Corresponding dendrogram.



Results on Segment 131

Table 9 shows the clustering results obtained by *k*-means on segment 131 (both projected and original), as well as the different values of parameters.

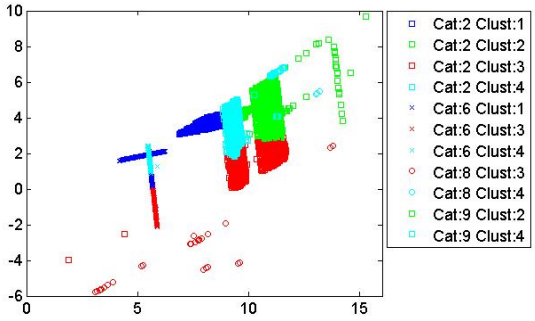
Table 9. *k*-means results on segment 131.

Data	k	Distance Criteria	SSH_C	SSH_W	Replicates/ Iterations	Sum of Distances
Projected	4	sqEuclidean	72.2797 %	0.0237 %	5/27	143881
Original	4	sqEuclidean	35.3580 %	0.0147 %	5/3	3.3944E+019
Projected	6	sqEuclidean	65.3593 %	0 %	5/29	94148.9
Original	6	sqEuclidean	18.2425 %	0 %	5/3	3.33533E+019
Projected	8	sqEuclidean	35.3425 %	0 %	5/21	57594
Original	8	sqEuclidean	17.1270 %	0 %	5/3	3.27634E+019
Projected	4	Cityblock	62.4323 %	0.0188 %	5/19	155608
Original	4	Cityblock	53.6005 %	0 %	5/3	8.50048E+011
Projected	6	Cityblock	64.6346 %	0 %	5/26	127850
Original	6	Cityblock	53.6005 %	0 %	5/10	8.49582E+011
Projected	8	Cityblock	44.2167 %	0 %	5/24	99653.2
Original	8	Cityblock	41.4321 %	0 %	5/13	8.44494E+011
Projected	4	Cosine	60.8088 %	0.1677 %	5/27	224.305
Original	4	Cosine	35.3580 %	0.0147 %	5/4	0.851133
Projected	6	Cosine	75.4176 %	0.0303 %	5/39	146.051
Original	6	Cosine	35.3580 %	0.0147 %	5/4	0.0121125
Projected	8	Cosine	75.6695 %	0.0213 %	5/32	107.85
Original	8	Cosine	35.3580 %	0 %	5/5	0.00636469
Projected	4	Correlation	60.1929 %	0.2053 %	5/24	43.4839
Original	4	Correlation	35.3580 %	0.0237 %	5/4	1.02209
Projected	6	Correlation	84.7884 %	0.0163 %	5/38	21.5802
Original	6	Correlation	35.3580 %	0.0147 %	5/4	0.0153503
Projected	8	Correlation	85.3673 %	0.0106 %	5/67	13.2321
Original	8	Correlation	35.3580 %	0 %	5/4	0.00806508

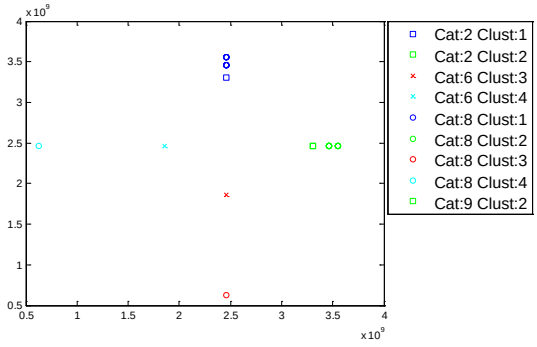
As a complementary result, Fig. 8 shows one of the experiments from Table 9, both projected (Fig. 8.a) and original (Fig. 8.b) data for *k*=4 and sqEuclidean distance criterion. The data has been labelled as follows: ssh_conn flows (Cat. 2), http_conn (Cat. 6), irc_sideeffect (Cat. 8) and icmp_sideeffect (Cat. 9).

Fig. 8. Visualization of *k-means* results on segment 131 (*k*=4 and sqEuclidean distance).

8.a *k-means* on projected data.



8.b *k-means* on original data.



Both experiments obtained a non-zero SSH_C and SSH_W rates. Experiments for the agglomerative method on segment 131 could not be completed due to the high amount of data contained in this segment.

5 Conclusions and Future Work

A clustering extension of MOVICAB-IDS has been proposed and applied to real-life flow-based in present paper. Descriptive details about the performed experiments on the different datasets and with several different clustering techniques and criteria can be found in section 3. Experimental results show that some of the applied clustering methods obtain a good clustering performance on the analysed data, comprising many different types of attacks. The obtained results vary from the different segments and the behaviour of the applied clustering techniques. These results are consistent with those previously obtained for packet-based data [6, 7].

There is not a distance criterion which obtains the best results, hence its selection depends on the analysed data. Comparing projected data results with the ones from original data, it can be said that better results (fewer number of groups with no errors) are obtained when processing projected data.

From the experimental results shown, it can be concluded that clustering methods could help in intrusion detection over flow-based network data. On the other hand, using clustering techniques, automatic response could be added to MOVICAB-IDS, to quickly abort intrusive actions while happening.

Future work will focus on extending the experimental setup to cover some different clustering techniques, as well as some other data sources to increase the detection rate and diversity of the proposed IDS. It would be interesting to test the proposed extension on data comprising both anomalous and legitimate traffic once a dataset containing such traffic is publicly available and accommodates to MOVICAB-IDS.

References

1. Quittek, J., Zseby, T., Claise, B., Zander, S.: Requirements for IP flow information export (IPFIX). (2004)
2. Sperotto, A., Schaffrath, G., Sadre, R., Morariu, C., Pras, A., Stiller, B.: An Overview of IP Flow-Based Intrusion Detection. Communications Surveys & Tutorials, IEEE 12, 343-356 (2010)
3. Sperotto, A., Pras, A.: Flow-based intrusion detection. In: Integrated Network Management (IM), 2011 IFIP/IEEE International Symposium on, pp. 958-963. (2011)
4. Corchado, E., Herrero, Á.: Neural Visualization of Network Traffic Data for Intrusion Detection. Applied Soft Computing 11, 2042–2056 (2011)
5. Herrero, Á., Corchado, E.: Mobile Hybrid Intrusion Detection: The MOVICAB-IDS System. Springer (2011)
6. Sánchez, R., Herrero, Á., Corchado, E.: Clustering Extension of MOVICAB-IDS to Identify SNMP Community Searches. Logic Journal of IGPL 23, 121-140 (2015)
7. Sánchez, R., Herrero, Á., Corchado, E.: Visualization and Clustering for SNMP Intrusion Detection. Cybernetics and Systems: An International Journal 44, 505-532 (2013)
8. Sperotto, A., Sadre, R., Vliet, F.v., Pras, A.: A Labeled Data Set For Flow-based Intrusion Detection. IP Operations and Management, pp. 39-50, Berlin (2009)
9. Friedman, J.H., Tukey, J.W.: A Projection Pursuit Algorithm for Exploratory Data-Analysis. IEEE Transactions on Computers 23, 881-890 (1974)
10. Herrero, Á., Navarro, M., Corchado, E., Julián, V.: RT-MOVICAB-IDS: Addressing Real-Time Intrusion Detection. Future Generation Computer Systems 29, 250-261 (2013)
11. Zheng, Q.H., Xuan, Y.G., Hu, W.H.: An IDS Alert Aggregation Method Based on Clustering. In: Zhang, H., Shen, G., Jin, D. (eds.) Advanced Research on Information Science, Automation and Material System, Pts 1-6, vol. 219-220, pp. 156-159. Trans Tech Publications Ltd, Stafa-Zurich (2011)

12. Qiao, L.B., Zhang, B.F., Lai, Z.Q., Su, J.S., Ieee: Mining of Attack Models in IDS Alerts from Network Backbone by a Two-stage Clustering Method. 2012 Ieee 26th International Parallel and Distributed Processing Symposium Workshops & Phd Forum, pp. 1263-1269. Ieee, New York (2012)
13. Jiang, S., Song, X., Wang, H., Han, J.-J., Li, Q.-H.: A clustering-based method for unsupervised intrusion detections. Pattern Recognition Letters 27, 802-810 (2006)
14. Cui, K.Y., Ieee: Research On Clustering Technique In Network Intrusion Detection. Ieee Computer Soc, Los Alamitos (2012)
15. Ge, L., Zhang, C.Q.: The Application of Clustering Algorithm in Intrusion Detection System. In: Jin, D., Lin, S. (eds.) Advances in Future Computer and Control Systems, Vol 1, vol. 159, pp. 77-82. Springer-Verlag Berlin, Berlin (2012)
16. Corchado, E., MacDonald, D., Fyfe, C.: Maximum and Minimum Likelihood Hebbian Learning for Exploratory Projection Pursuit. Data Mining and Knowledge Discovery 8, 203-225 (2004)
17. Corchado, E., Fyfe, C.: Connectionist Techniques for the Identification and Suppression of Interfering Underlying Factors. International Journal of Pattern Recognition and Artificial Intelligence 17, 1447-1466 (2003)
18. Seung, H.S., Socci, N.D., Lee, D.: The Rectified Gaussian Distribution. Advances in Neural Information Processing Systems 10, 350-356 (1998)
19. A.K. Jain, M.N.M., P.J. Flynn: Data Clustering: A Review. ACM Computing Surveys 31, (1999)
20. Xu, R., Wunsch, D.C.: Clustering. Wiley (2009)
21. Andreopoulos, B., An, A., Wang, X., Schroeder, M.: A roadmap of clustering algorithms: finding a match for a biomedical application. Briefings in Bioinformatics 10, 297-314 (2009)
22. Zhuang, W.W., Ye, Y.F., Chen, Y., Li, T.: Ensemble Clustering for Internet Security Applications. Ieee Transactions on Systems Man and Cybernetics Part C-Applications and Reviews 42, 1784-1796 (2012)
23. Tu, Q., Lu, J.F., Yuan, B., Tang, J.B., Yang, J.Y.: Density-based hierarchical clustering for streaming data. Pattern Recognition Letters 33, 641-645 (2012)
24. Brailovsky, V.L.: A probabilistic approach to clustring. Pattern Recognition. Lett. 12, 4 193-198 (1991)
25. Argyrou, A.: Clustering Hierarchical Data Using Self-Organizing Map: A Graph-Theoretical Approach. In: Principe, J.C., Miikkulainen, R. (eds.) Advances in Self-Organizing Maps, Proceedings, vol. 5629, pp. 19-27. Springer-Verlag Berlin, Berlin (2009)
26. Liu, Y., Li, Z., Xiong, H., Gao, X., Wu, J.: Understanding of internal clustering validation measures. pp. 911-916. IEEE, (2010)
27. Caliński, T., Harabasz, J.: A dendrite method for cluster analysis. Communications in Statistics-theory and Methods 3, 1-27 (1974)
28. Rousseeuw, P.: Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. J. Comput. Appl. Math. 20, 53-65 (1987)
29. Tibshirani, R., Walther, G., Hastie, T.: Estimating the number of clusters in a data set via the gap statistic. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 63, 411-423 (2001)
30. Fabien Pouget, Marc Dacier: Honeypot-based forensics. AusCERT2004, AusCERT Asia Pacific Information technology Security Conference 2004, 23rd - 27th May 2004, Brisbane, Australia, (2004)