# COMBINED DATA MINING APPROACH FOR INTRUSION DETECTION

U. Zurutuza*, R. Uribeetxeberria, E. Azketa, G. Gil, J. Lizarraga and M. Fernández
*Computer Science Department, Mondragon University, Mondragon (Gipuzkoa), Spain*

Keywords: Computer security, intrusion detection, alert correlation, data mining.

Abstract: This paper presents the results of the project MIAU, a data mining approach for intrusion detection alert correlation. MIAU combines different data mining techniques in order to properly solve some existing problems in the management and analysis of alerts generated by actual intrusion detection systems. Some of these data mining methods and their application to MIAU are introduced in this paper. Experiments have been carried out with the purpose of demonstrating the validity of the proposed model and some conclusions about them are extracted. Finally, some possible improvements for the system and further work are exposed.

## 1 INTRODUCTION

Actual intrusion detection systems present some problems like the generation of a large amount of different alerts which analysis is in some cases unviable. Other well known problem is the generation of false positives which distort the real vision of the malicious network traffic. Nowadays, the approach to solve these problems undertakes the elimination of the false positives, as well as the correlation, clustering and fusion of positive alerts. The goal is offering a clearer vision of the attacks or intrusion attempts responsible of their generation.

The MIAU project focuses on the area of alert clustering and fusion. The goal of the project is increasing the semantic level of alerts (transforming them into meta-alerts) and their later chronological reordering to identify global attack scenarios. MIAU automates all the process by data mining techniques.

## 2 RELATED WORK

The scientific community has worked on the problems exposed previously using different techniques.

Ning et al. have been working on alert correlation for several years. Their work consists on generating hyper-alerts based on a knowledge base with prerequisites (pre-condition) and consequences (post-condition) of attacks (Ning, Ciu, Reeves, 2002). They also provide a visualisation mechanism, but the knowledge base must be previously generated, while the system of this paper does not need any previous knowledge.

(Debar, Wespi, 2001) implemented a method to aggregate and correlate alarms to show them in a more condensed view. They carried out the correlation by means of duplicates and consequences.

Similarly, (Cuppens, Miège, 2002) automatically created correlation rules declared by means of a predicate logic. They first specify and define offline logic links between the post-condition of an attack and the pre-condition of another attack. When a new alert arrives, it is checked whether it is potentially correlated with another stored alarm or not.

In a similar approach to the ones seen above, (Templeton, Levitt, 2000) and (Zhou, Heckman, Reynolds, Carlson, Bishop, 2007) present a requires/provides model to define relations between alarms. All these systems need the definition of a specific language to model the relations.

Data mining techniques have also been used before for alert clustering. (Julisch, 2003) clusters alerts using the AOI (Attribute Oriented Induction) algorithm, based on pre-configured generalisation hierarchies of alert attribute values. The target of his work is to reduce the number of alerts an IDS triggers, discovering the root cause of the alarms,

* *This author was supported in part by grant No. BFI 05.454 awarded by the Department of Education, Universities and Research of the Basque Government.*

Zurutuza U., Uribeetxeberria R., Azketa E., Gil G., Lizarraga J. and Fernández M. (2007).
COMBINED DATA MINING APPROACH FOR INTRUSION DETECTION.
In *Proceedings of the Second International Conference on Security and Cryptography*, pages 67-73
DOI: 10.5220/0002122800670073

but it does not provide a mechanism for attack scenario generation.

Both (Manganaris, Christensen, Zerkle, Hermiz, 2000) and (Treinen, Thurimella, 2006) have applied association rules algorithms to discover frequent alarm sets. The formers discover frequent alarm sets which are then treated as the normal behaviour and let the analysts focus on the anomalies. The later ones obtain rules which identify known attack patterns in alarm streams. (Clifton, Gengo, 2000) also made use of data mining in order to look for frequent alarm sequences (using frequent episodes algorithm) produced by normal operations, and this way, be able to remove most of the false positives. None of the works above that use data mining techniques provide a mechanism for attack scenario generation.

Another approach closer to the one presented in this paper is the work done by (Valdés, Skinner, 2001) and (Dain, Cunningham, 2001). Valdés and Skinner presented the idea of applying probabilistic similarity measures for the fusion of alerts into meta-alerts. Later, they try to rebuild the attack scenario relaxing the similarity measure of the attack type. Dain and Cunningham proposed an algorithm that generates scenarios by estimating the probability of an incoming alert to belong to a certain scenario. Within data mining techniques used by them, radial base function networks, multilayer perceptrons and decision trees were tested, being the best results obtained by the last one.

# 3 MODEL DESCRIPTION

## 3.1 Introduction

MIAU performs the correlation and analysis of alerts generated by IDSs using a multiple phase method and some data mining techniques. The first step is Prepocessing the available data. Prepocessing consists on extracting only the fields of the alert that will later be useful for the analysis. The second step to be taken is the Clustering phase where the whole alert collection is segmented in distinct clusters according to the similarities between them. After that, in the Association stage the similarities between alerts of the same cluster are deduced. Finally, each association rule extracted in the previous step is labelled as a known attack or network traffic in the Codification and Identification phase.

All the process is carried out automatically via data mining techniques supplied by the Java API of WEKA (http://weka.sourceforge.net/doc) an open source tool developed by (Frank, Hall, Trigg) and explained in (Witten, Frank, 2005).

## 3.2 Preprocessing

Data which is not relevant for the analysis exists among all the information of the IDS alerts. The Preprocessing phase consists on isolating the valid attributes for the analysis.

The task of selecting useful alert attributes has been tackled by trial and error methods. The clustering algorithm has been applied over an alert set which contains a known attack scenario and non malicious network traffic. Different combinations of attributes have been tested, and their validity has been measured by analysing the quality and homogeneity of the obtained clustering.

The chosen attributes have been the following: timestamp, source IP, destination IP, IP packet length, source port and destination port.

## 3.3 Clustering

The main target of the clustering stage consists on dividing all the alert set into clusters according to their attributes. This process seeks getting coherent and homogeneous clusters according to the similarities of the alerts that compose each of them.

Clustering offers a high level vision of the amount of different traffic types existing within the whole collection of alerts generated by IDSs.

The chosen clustering algorithm has been *Expectation Maximization (EM)* (Dempster, Laird, Rubin, 1977). This algorithm automatically distributes instances (alerts) into a certain number of clusters considered the optimum one for that dataset.

*EM* is a probabilistic model that alternates between performing an expectation step *(E)* and a maximization step *(M)*. *E* step computes the expectation of the likelihood by including the latent variables as if they were observed. On the other hand, *M* step computes the maximum likelihood estimates of the parameters by maximizing the expected likelihood found on the *E* step. The parameters found on the *M* step are then used to begin another *E* step, and the process is repeated iteratively.

Other clustering algorithms such as *K-means* and *COBWEB* have also been considered (Witten, Frank, 2005). Nevertheless, they have been discarded in this work. At the beginning, the quantity of existing different network traffic types is unknown, and that is why the number and type of generated alerts will also be unknown. Accordingly, we are not aware of the optimum number of needed clusters into which distribute all those alerts properly. This generates the necessity of using a clustering algorithm which automatically calculates the optimum quantity of clusters, and that is why *K-means* has been rejected. Among the clustering algorithms that are able to accomplish this fundamental requirement, those which make a probabilistic distribution of the alert collection ignoring the order the alarms are introduced in the training phase are preferable. The reason for this is that the order is altered according to the network alerts, and these are changing and unpredictable. *COBWEB* is rejected because of the reasons explained above and the *EM* clustering algorithm is the selected one.

## 3.4 Association

The main target of this stage is to automatically get association rules for each cluster generated in the Clustering phase.

The association rules algorithm utilised is the *Apriori* algorithm (Agrawal, Srikant, 1994).

The association rules extracted for each cluster have the format defined in Equations 1 and 2.

$$1.\ A(R)=>C(S)(U) \qquad [U=S/R\ \ (S<=R)] \tag{1}$$

$$n.\ A..N(W)=>C..M(X\,)(Y)\ [Y=X/W\ (X<=W)] \tag{2}$$

Symbol ‘=>’ halves the rule into two sides: the premise on the left and the consequence on the right. *A, C,… N* and *M* are alert attributes. *R, S, W* and *X* represent number of occurrences of the value of the attribute/s in the data set. Finally, *U* and *Y* define the confidence of the rule and it is always smaller or equal to one.

For each rule with confidence value of one, the algorithm generates another rule with the same attributes and values but with the premise and the consequence inverted. As a result, two rules providing the same information are obtained, being possible to discard any one of them.

The number of occurrences and attributes which may be at any side of the rule is diverse. Neither all rules contain the same attributes nor the quantity of attributes in each side of the rule has to be the same.

All deduced rules are inserted in a table whose data fields are alert attributes. They are stored ignoring the position these attributes have in the rules obtained by the algorithm. Each record in the table corresponds to one rule. Repeated rules are deleted from the table, and summarisation is made with those ones which, in spite of having different information, refer to the same sub-collection of alerts. Thus, the high number of rules deduced by *Apriori* algorithm is reduced to just a few ones for each cluster. Furthermore, some counts are done for each rule to facilitate the identification of certain attack types. Table 1 shows examples of them.

Table 1: Example of factors that may indicate certain attack types.

| *Factor* | *Possible attack type* |
|---|---|
| Different source IP addresses quantity | High value may indicate source IP spoofed DoS or DDoS attack |
| Different destination IP addresses quantity | High value may indicate IP sweep |
| Different destination ports quantity | High value may indicate port scan |

## 3.5 Codification and Identification

The target of the Codification is to obtain a rule-set independent of the attribute’s explicit value. This provides an abstraction level where the relevance of some attributes does not rely on their value, but in the way they relate with the rest of attributes, as well as the frequency this value appears for the different alerts.

Some alert attributes may contain data fields that take different values depending on particular variables such as source, destination, length, etc. For example: explicit values of source and destination are not relevant to identify a port scanning. On the other hand, the number of packets with the same length coming from the same source and going to different and consecutive destination port numbers is and important data. Therefore, attributes whose explicit value is not important in the task of identifying the attack pattern will be codified.

A table containing descriptions of codified well known attacks and normal network traffic patterns is built manually before the system is run (from now on referred as *ATP* table). To do so, the same attributes used during the alert analysis phase have been taken into account. These values are codified in such a way to ease the comparison between the coded association rules and *ATP* in order to identify each rule. Unrelated rules describe alerts generated by unknown attacks or not characterised network traffic. These are labelled as suspicious, as well as those other rules whose source IP address coincide with the one of a rule labelled as known attack.

The values of labelled rule attributes are compared with the attributes of all individual alerts, and those alerts whose values fit in the rule are labelled accordingly. This way, malicious alerts can be combined to attacks scenarios. It has been mentioned that *ATP* rules comparison can only be applied over previously extracted association rules because *ATP* patterns not only take into account attribute values but also the relationship between the attribute values among different alerts.

The values of *Count* data fields of *ATP* table define a threshold. Every codified rule with a value greater than this minimum limit and keeping the rest of fields as in *ATP* table (Table 2) will be labelled with its corresponding *Type* and *Method* values.

Table 2: Codified well known attack and normal network traffic patterns table *(ATP)* format.

| *Data field* | *Example value* |
|---|---|
| Type | IP sweep |
| Method | TCP |
| Signature identifier | - |
| Count different source IP | 1 |
| Count different destination IP | 25 |
| Destination port | - |
| Count different destination port | 0 |
| Transport protocol code | 6 |
| IP packet length | 1 |

# 4 EXPERIMENTAL RESULTS

## 4.1 Real Scenario

Two victims have been deployed in a LAN with approximately 1000 hosts: a Windows XP host in a VLAN and a Windows 2003 IIS Web Server in the DMZ. Two Linux with Snort have also been deployed in the same LAN segments where the victim hosts are placed.

The attack has been accomplished from another Windows XP from another VLAN, using the Core Impact penetration-testing tool. First, the network has been scanned looking for vulnerabilities. As a result, the RPC-DCOM vulnerability in the XP host has been discovered. Secondly, an exploit has been launched against that vulnerability and root privileges have been gained. After that, a DoS tool has been uploaded using TFTP. The previous information gathering has also revealed the IIS Web Server placed in the DMZ. Finally, a DoS attack has been launched against this web server by sending a great amount of large ICMP packets from the Windows XP host.

### 4.1.1 Preprocessing

Attributes exposed in section 3.2 have been used in the 26660 events accumulated in the experiment.

### 4.1.2 Clustering

Figure 1 shows the distribution made by the *EM* clustering algorithm over the alert set.

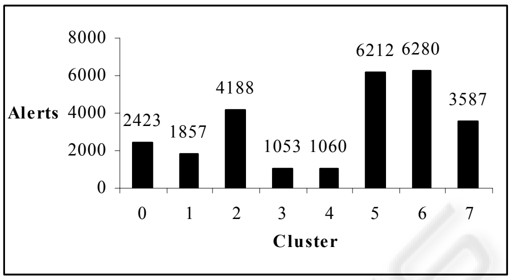

Figure 1: Distribution of 26660 alerts of real scenario.

### 4.1.3 Association

The *Apriori* association rules algorithm extracts 1133 rules. As shown in section 3.4, the application of the rule redundancy reduction method decreases the amount of them to 60 rules. Table 3 shows some of the obtained rules. Analysing them, it is possible to observe that the homogeneity of cluster 0 is not the desired one. Both *ID* 1 and *ID* 7 rules belong to cluster 0, but they seem to be quite different. However, getting a large rule-set allows identifying very different types of alerts clustered together. For example and continuing with the analysis, the *ID* 46 tuple shows a rule obtained from cluster 5. *Source IP* data field is empty, so it may indicate multiple source IP traffic with the attributes of this rule. Generalising, empty data fields are ignored. On the contrary, *Source IP* and *Destination IP* attributes are codified as explained in the next section.

### 4.1.4 Codification and Identification

Table 4 is the result of the Codification phase. First, applying the method exposed in section 3.5, the values of *Source IP* and *Destination IP* are codified to get an abstraction from their explicit values. After that, different source IP, different destination IP and different destination ports are counted for each rule in Table 3. This method allows identifying some types of attacks, as it is explained in section 3.4. The values of *Cluster*, *Signature Identifier*, *Destination port* and *Transport protocol code* are not modified because they are very useful during the rule Identification phase.

As it is exposed in section 3.5, the Identification stage consists on comparing codified association rules table represented in Table 4 with *ATP* table. As a result, each malicious and suspicious rule is identified and labelled with its corresponding category to ease its comprehension.

Table 5 shows that every attack and suspicious alert was grouped into cluster 0. The most precise identification was obtained in rules with 469 and 499 signature identifiers. The first one has been identified as *IP sweep* type attack using *ICMP*. The second one has been labelled as *DoS* attack using *ICMP Flood*. The rest *IP sweep* labelled rules may be considered as SNMP sweeps, and therefore, malicious traffic. Those labelled as suspicious were identified because their *Source IP* coincides with a previously labelled rule: in this case, *IP sweep*.

## 4.2 DARPA Data Sets Scenario

The experiment has been extended using the 2000 DARPA Intrusion Detection Data Sets. Not everybody (Mahoney, Chan, 2003) in the research community believes in the relevance of the results obtained with this sets. Anyhow, the authors think that it is an interesting experiment.

### 4.2.1 Preliminary Results

The analysis has been carried out over 12064 alerts.

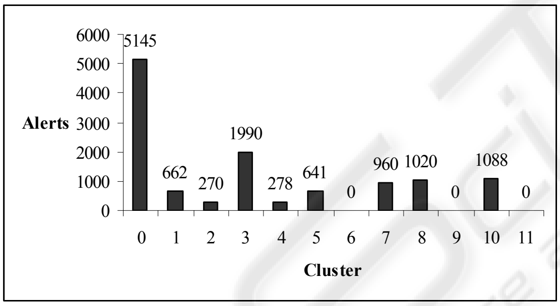

Figure 2: Distribution of 12064 alerts of DARPA Data Set.

### 4.2.2 Preprocessing

The attributes exposed in the Preprocessing section have been taken.

### 4.2.3 Clustering

Figure 2 shows the clustering. Note that when *EM* is initialised with a random cluster number, model can converge in empty clusters (6, 9 and 11).

### 4.2.4 Association

The *Apriori* association rules algorithm deduces 401 rules. The rule redundancy reduction method decreases the amount of them to 57 rules.

### 4.2.5 Codification and Identification

Table 6 shows the final results obtained with the DARPA Data Set scenario. The labelled association rules of Table 6 maintain a coherency with the attack steps exposed, so they can be called meta-alerts. The *IP sweep* and the *Spoofed source DoS* attack have been properly identified. The rest of meta-alerts have been tagged as *Suspicious*, since their *Source IP* coincides with the *Source IP* of the above mentioned rules. Besides, two meta-alerts with *Destination port* 111 can be seen. This is the attacked service. Launching of the exploit is the cause of meta-alert with *IP packet length* 1440. After this, a *Destination port* 23 meta-alert can also be seen. This is the service used to remotely control the DoS tool. Finally, the spoofed source DoS attack appears clearly identified.

It seems important to mention that the attack steps have been grouped in different clusters. IP sweep is from cluster 8. The vulnerability exploitation meta-alerts are included in cluster 7. The remote control via telnet (*Destination port* 23) is from cluster 3. And the DoS attack meta-alert belongs to cluster 2. This homogeneity may cause the extraction of valid rules (meta-alerts) and the precise identification of the attack steps.

# 5 CONCLUSIONS

Analysing the results of the DARPA Data Set scenario experiment, it seems clear that a homogeneous clustering is very useful in the task of identifying all the steps of an attack. Each step belongs to a different cluster and, consequently, their corresponding rules are properly deduced.

On the contrary, in the real scenario experiment the Clustering phase does not obtain a perfect segmentation of the whole alert-set: some clusters contain very different alert types while other ones group very similar alarms. That is why some phases of the attack, such as the exploit launching or the remote control, are not identified. In spite of this, the Clustering eases the extraction of association rules.

In the real scenario experiment, it seems that the association rules extraction phase does not obtain all the desirable rules. *Apriori* algorithm extracts rules whose information can be repeated from one rule to another. That is why, and with the purpose of obtaining a valid rule of a certain alert type, *Apriori* has to be configured to extract a large amount of them. Nevertheless, some clusters may contain several alert types. If some valid rules for each alert type are desired, the quantity of rules to be deduced by *Apriori* will be very large. This, apart from having a high computational cost, complicates the rules redundancy reduction task. In any case, the association rules provide many benefits: they are a representative sample of the whole alert-set and facilitate the legibility of the alarm-set.

As a conclusion, MIAU supplies an attack scenario detection system by applying a novel data mining algorithm combination. The results have been very encouraging and there is still room for improvement the system.

## 6 FURTHER WORK

One possible improvement of MIAU consists on tuning the *EM* algorithm configuration parameters in order to obtain a more coherent and homogeneous clustering. These configuration parameters, such as the number of iterations or the allowable maximum standard deviation, can be adjusted finely even by means of trial and error methods. That way, the Clustering phase would obtain a better segmentation of the whole alert-set, without including very different alerts into the same cluster. Consequently, the association rules algorithm would be able to extract more precise information because it would be working on a more homogeneous alert-set.

The association rules deducing phase may also be improved adjusting the *Apriori* algorithm configuration parameters, such as the number of rules or the metric type. In addition to this, the rules redundancy reduction algorithm may be optimised with the purpose of obtaining a more compact, precise and complete rule-set in less time.

Another possible improvement of MIAU is the creation of a complete *ATP* table, permitting the system to cover the whole type of traffic that can exist in a network. This can be made analysing the characteristics of known attacks and traffic and codifying them with the *ATP* table format.

## REFERENCES

Ning, P., Ciu, Y., Reeves, D., 2002. Analyzing Intensive Intrusion Alerts Via Correlation. In *Proceedings of the 5th International Symposium on Recent Advances in Intrusion Detection.* Springer-Verlag, Switzerland, pp.74-94.

Debar, H., Wespi, A., 2001. Aggregation and Correlation of Intrusion-Detection Alerts. In *Proceedings of the 4th International Symposium on Recent Advances in Intrusion detection*. Springer-Verlag, USA, pp. 85-103.

Cuppens, F., Miège, A., 2002. Alert Correlation in a Cooperative Intrusion Detection Framework. In *Proceeding of the 2002 IEEE Symposium on Security and Privacy.* IEEE Computer Society. USA, pp. 202

Templeton, S., Levitt, K., 2000. A requires/provides model for computer attacks. In *Proceedings of the Workshop on New Security Paradigms*. pp. 31–38.

Zhou, J., Heckman, M., Reynolds, B., Carlson, A., Bishop, M., 2007. Modelling network intrusion detection alerts for correlation. *ACM Transactions on Information and System Security 10 (1)*.

Julisch, K. 2003. Clustering Intrusion Detection Alarms to Support Root Cause Analysis. *ACM Transactions on Information and System Security 6 (4).* ACM Press, pp. 443-471.

Manganaris, S., Christensen, M., Zerkle, D., Hermiz, K., 2000. A Data Mining Analysis of RTID Alarms. *Computer Networks 34 (4).* Elsevier North-Holland, Inc., pp. 571-577.

Treinen, J.J., Thurimella, R., 2006. A framework for the application of association rule mining in large intrusion detection database. In *Proceedings of the 9th International Symposium on Recent Advances in Intrusion Detection.* Springer-Verlag, Zurich, pp. 1-18.

Clifton, C., Gengo, G., 2000. Developing custom intrusion detection filters using data mining. In *2000 Military Communications International Symposium.* USA. pp. 22-25.

Valdés, A., Skinner, K., 2001. Probabilistic Alert Correlation. In *Proceedings of the 4th International Symposium on Recent Advances in Intrusion Detection.* Springer-Verlag. USA.

Dain, O., Cunningham, R.K., 2001. Fusing Heterogeneous Alert Streams into Scenarios. In *Proceedings of the ACM CCS Workshop on Data Mining for Security Applications*. Barbará and Jajodia. USA.

Witten, I., Frank, E., 2005. *Data Mining: Practical Machine Learning Tools and Techniques.* Morgan Kaufmann Series in Data Management Systems.

Dempster, A., Laird, N., Rubin, D., 1977. *Maximum Likelihood for Incomplete Data via the EM Algorithm*. Royal Statistical Society, Vol.1, N.1.

Agrawal, R., Srikant, R., 1994. Fast Algorithms for mining association rules in large databases. In *Proceedings of 20th International Conference on Very Large Databases*. Santiago de Chile. pp. 487-489.

Table 3: Some tuples from association rule table.

| *ID* | *Cluster* | *Signature identifier* | *Source IP* | *Destination IP* | *Destination port* | *Transport protocol code* | *IP packet length* | *Confidence* |
|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 1418 | 192.168.100.30 | | 161 | 6 | 48 | 1 |
| 7 | 0 | 499 | 193.146.78.1 | 193.146.78.49 | | 1 | 60028 | 1 |
| 17 | 1 | 1653 | 192.168.100.37 | 192.168.100.15 | 80 | | | 1 |
| 25 | 2 | 1411 | 192.168.100.52 | 192.168.100.1 | 161 | 17 | 74 | 1 |
| 46 | 5 | 1917 | | 239.255.255.250 | 1900 | 17 | 161 | 0.91 |
| 60 | 7 | 466 | | 192.168.100.9 | | 1 | 60 | 0.99 |

Table 4: Some tuples from codified association rule table.

| *ID* | *Cluster* | *Signature identifier* | *Count different source IP* | *Count different destination IP* | *Destination port* | *Count different destination port* | *Transport protocol code* | *IP packet length* |
|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 1418 | 1 | 38 | 161 | 0 | 6 | 48 |
| 7 | 0 | 499 | 1 | 1 | | 0 | 1 | 60028 |
| 17 | 1 | 1653 | 1 | 1 | 80 | 0 | | |
| 25 | 2 | 1411 | 1 | 1 | 161 | 0 | 17 | 74 |
| 46 | 5 | 1917 | 675 | 1 | 1900 | 0 | 17 | 161 |
| 60 | 7 | 466 | 18 | 1 | | 0 | 1 | 60 |

Table 5: Automatically generated multi-step attack scenario of real scenario experiment, ordered by time.

| *Cluster* | *Type* | *Method* | *Signature identifier* | *Signature* | *Source IP* | *Destination IP* | *Destination port* | *Transport protocol code* | *IP packet length* |
|---|---|---|---|---|---|---|---|---|---|
| 0 | IP sweep | ICMP | 469 | ICMP ping NMAP | 192.168.100.30 | | | 1 | 28 |
| 0 | IP sweep | TCP | 1418 | SNMP request TCP | 192.168.100.30 | | | 6 | 48 |
| 0 | Suspicious | Port 162 | 1653 | | 192.168.100.30 | | 162 | | |
| 0 | Suspicious | Port 161 | 1417 | SNMP request UDP | 192.168.100.30 | | 161 | | |
| 0 | IP sweep | TCP | 1420 | SNMP trap TCP | 192.168.100.30 | | 162 | 6 | 48 |
| 0 | DoS | ICMP Flood | 499 | ICMP large packet | 193.146.78.1 | 193.146.78.49 | | 1 | 60028 |

Table 6: Automatically generated multi-step attack scenario of DARPA Data Set experiment, ordered by time.

| *Cluster* | *Type* | *Method* | *Signature identifier* | *Signature* | *Source IP* | *Destination IP* | *Destination port* | *Transport protocol code* | *IP packet length* |
|---|---|---|---|---|---|---|---|---|---|
| 8 | IP sweep | ICMP | 384 | ICMP ping | 202.77.162.213 | | | 1 | 38 |
| 7 | Suspicious | Port 111 | | | 202.77.162.213 | 172.16.112.50 | 111 | | |
| 7 | Suspicious | Port 111 | 585 | RPC sadmind request UDP | 202.77.162.213 | | 111 | 17 | 84 |
| 7 | Suspicious | | | | 202.77.162.213 | | | 17 | 1440 |
| 3 | Suspicious | Port 23 | | | 202.77.162.213 | 172.16.112.10 | 23 | | |
| 2 | Spoofed Source DoS | TCP Flood | 528 | Bad-Traffic loopback traffic | | 131.84.1.31 | | 6 | 40 |