CLUSTERED VISUALIZATION FOR NETWORK INTRUSION ANALYSIS

AURANGZIEB ZIEB RANA, MAO LIN HUANG

Faculty of Information Technology, University of Technology, Sydney
Broadway NSW 2007, Australia
E-mail: azrana@it.uts.edu.au, maolin@it.uts.edu.au

Abstract:

The attacks on computer networks and computer systems are increasingly becoming numerous and sophisticated in nature. Hence this has given a growing need for intrusion detection systems to identify these attacks. The use of these intrusion detection systems have given rise to another problem, the handling, and the presentation of large amounts of alerts generated by these systems. In this paper we introduce a layered framework for network intrusion analysis and use a clustered visualization technique to group and visualize large amounts of alerts, and other network nodes based on their similarities. This layered clustering visualization allows users to visually explore and analyze the intrusion data. The cross-navigation between layers is achieved by user's interaction. This allows users to actively display the groups and associated properties of these alerts and nodes in an efficient way.

Keywords:

Information visualization; clustered visualization; data clustering; Intrusion Detection Systems; Interactive visualization

1. Introduction

The traditional computer security efforts mainly have been aimed at rendering computer networks invulnerable to attacks by employing tools such as firewalls, and Intrusion Detection Systems. The organizations rely on Intrusion detection analysts to monitor output from intrusion detection systems. These analysts use IDS output complemented by other system, firewall, and network logs to carry out their analysis work. These logs can grow enormously in size containing huge amount of textual information, and hence making manual review impossible.

These causes lots of attacks to go unnoticed and are not detected as mentioned by global security survey (2003). Even though the IDSS identify potential intrusive alerts, however due to the complicated nature of detecting actual intrusions, most current IDSS place the job of distinguishing an actual alert from a set of large alerts on the IDS analyst, causing a significant cognitive load from an analyst. This cognitive load can be reduced by the use of information visualization by taking advantage of human perceptual abilities to increase cognition as talked by Shneiderman (1996).

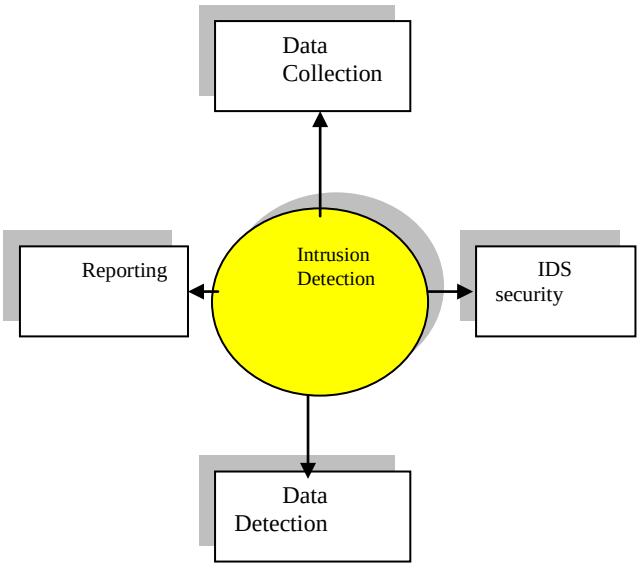

Figure 1: The main intrusion detection research areas

Figure 1 shows the main research areas in intrusion detection. As can be seen from the diagram they are data collection, data detection, reporting, and Intrusion detection system security. The data collection area is about defining what data to be collected as an input to the detection system. It raises research questions such as mechanisms for collecting data. How the data will be stored. The collection points in the network for the data as well as logging of the intrusive alerts. How they will be logged. The IDS

security research involves protecting various different components of the IDS against attacks especially the protection of data storage, and data collection points. The data detection and evaluation research concentrates on finding optimal algorithms to distinguish abnormal network data from network traffic. The reporting research is concerned with finding the best way to display the detected intrusion alerts to bring to the attention of the system administrator the series of events that are suspicious.

This paper concentrates on the reporting aspects of IDS. Next we discuss the related research in the area followed by our technique of layered approach to intrusion analysis, and clustering visualization to improve the way the intrusive data is displayed for visualization purposes. We then discuss advantages of using our approach followed by a conclusion and future work.

2. Related research

There are several visualization tools available that employ various visualization techniques to display intrusive network traffic. The work by Girardin and Brodbeck (1998) describes the combination of machine learning and visualization to provide automatic classification of firewall generated network events to facilitate network profiling and log analysis. The work carried out by Erabacher (2002) shows a two dimensional glyph based visual overview of a single host and a related network to represent intrusion detection visualization. Similarly a tool like NIVA as mentioned by Nyarko, and Capers (2002) uses the input from existing IDS sensors to interactively display that data using glyphs in three dimensional space. The paper by Solka (2002) uses statistical visualization methods such as parallel coordinate plots and circle plots to display the network data collected from existing IDSs.

However none of these techniques discuss the use of clustering to display related alert data in the form of clusters to display direct relationship between alerts, attacking hosts, and effected nodes. Our technique uses clustered visualization integrated with real-time visualization to visually display intrusive activities. The real-time visualization is enabled by using real-time visual querying.

The clustered visualization is a new type of the information visualization techniques, such as DA-TU developed by Eades and Huang (2000) and Node Grouping proposed by Six and Tollis (2001) that can be used to deal with the large scale visualization problem. In clustered visualization, related nodes are grouped in “clusters” or super-nodes and then are drawn in a geometric plant for visualization. The user sees a “summary” of the graph: the super-nodes and super-edges between the super-nodes. Some clusters may be shown in more detail than others.

The increase use of very high-speed lines has caused collected network data to become very large in volume. As a result the amount of data the security analyst must cope with is increasing in size. The grouping of alert data such that the data within a single cluster have similar characteristics, while points in different clusters are dissimilar can allow more effective visualization of large amount of alerts by allowing the security analyst to predict attack patterns. For instance, one cluster may consist of ping sweep attack, while other consisting of TCP port scan attack. The clusters can then be used to characterize the occurrence of different attacks in the form of alerts. These characterizations can then be used to derive targeting attackers such that specific attackers are directing their attacks towards specific group of hosts or a single host. This mapping of attackers to effected nodes can be used to conclude for instance target hosts for attacker A, and target nodes for attacker B. By further using real-time visualization to display exchanged dialogue between attacking node, and effected hosts, the security analyst can predict future behavior of a particular attacking host, based on their existing behavior. Next we discuss in more detail on how our approach works.

3. Our approach

We use a layered approach to analyze network intrusion data to facilitate both monitoring, and analysis phase of intrusion detection. Figure 2 shows our layered architecture for intrusion detection analysis. The layered approach to intrusion detection will allow monitoring of all attack alerts and will be used to identify potentially suspicious alerts. It will reduce cognitive load on security analysts in analyzing alert data, and analyzing other related data such as effected network nodes and attacking hosts to diagnose attacks.

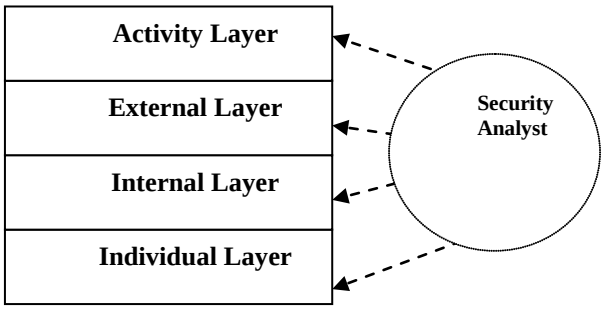

Figure2: Layered architecture for intrusion detection.

The layered approach to intrusion detection and analysis will provide an interactive monitoring of attacks through implicit relationship between layers described above. The first phase of intrusion detection is the surveillance of the network infrastructure and available network resources as mentioned by Koziol (2003). This task involves monitoring of IDS generated output. Our layered approach will provide the surveillance of the network infrastructure and available network resources. Next we discuss each layer. The transition from each layer will allow both the monitoring and analysis of intrusion detections.

3.1. The activity layer

The activity layer can provide the monitoring functionality in the form of 2D displays of alerts according to alert type. It will allow grouping of alerts based on common alert properties as shown in table 2. The grouping of alerts is discussed in the next section. The activity layer display of alerts will allow for continuous monitoring without the need for focused attention. The layer can provide the starting point for recognizing and flagging events that require further investigation.

3.2. The external layer

The external layer will provide clusters of attacking nodes according to displayed alerts by the activity layer. The clustering will be derived based on attributes of the nodes given in table 1. The section 4 of the paper discusses the grouping technique. This layer will relate the generated alerts to the attacking hosts and will provide overview display. The external layer will serve the purpose of displaying an overview of the current intrusion activity in the form of attacking node clusters derived from grouped alerts in the activity layer. Each attacking node group will correspond to the alert type displayed in the activity layer. The combination of activity and external layer will provide a big picture by relating alerts and attacking nodes.

3.3. The internal layer

The internal layer will provide a group of internal nodes affected by a particular node selected from external layer cluster. The attributes given in table 1 will be used to construct internal layer node clusters. This layer will allow further analysis of traffic between an attacking node and internal network node. The analysis will be achieved by the individual layer.

3.4. The individual layer

The individual layer will be used to allow real-time visualization of suspected dialogue between attacker and residential node by enabling real-time visual query system. At each layer excluding the individual layer the links between clusters will be formed using a force model as described by Eades and Huang (2000). The links at activity layer between the alerts will also be formed by using a force model discussed further in the paper. Next we discuss in more depth on how our technique of grouping alerts and nodes will work. The next section discusses how we will form cluster of similar alerts and nodes for visualization.

4. The grouping of alerts and nodes

The set of information associated with alerts and nodes (network devices) are called their properties. In this section we discuss the most common properties of both alerts and nodes. The grouping of nodes and alerts are based on the similarities of these properties.

4.1. The properties of alerts and nodes

The table 1 below shows the node attributes that can be used for grouping nodes together to form clusters of nodes. Similarly table 2 below shows the alert attributes in relation to grouping of alerts to derive clustered visualization.

4.2. Similarity rules

The alerts can be grouped together based on some similarity rules. We define our similarity rules based on considering that IDS generated alerts consists of several attributes such as destination and source IP addresses etc as shown in table 1. We can consider alerts to be a set $\{A_1 \ldots A_n\}$ and attributes to be as a set $\{P_1 \ldots P_n\}$. We can use this technique to derive common relationships based on using the common properties of the alerts. We can derive grouping of alerts based on using say an attribute P1 for both alert A1 and A2. Similarly we can derive subsets of the attribute set such as $\{P_1 \ldots P_9\}$ and can use the subset to derive alert groups. We can further group large amount of alerts together by based on particular attribute's value ranges. For instance using the destination ip address of an alert, we can derive a subnet and group alerts according to the subnet value.

This can be further illustrated by an example. If there are alerts A_1, A_2, A_3, and A_4, with destination addresses as

192.168.0.1, 192.168.0.2, 192.168.0.3, and 192.168.0.4 consecutively. We can use this information to derive a subnet for these ip addresses and then group alerts according to the subnet value. In this case a subnet can be 192.168.0.0. By using the same approach, we can also derive node clusters. For example we can consider nodes to be a set $\{N_1 \ldots N_n\}$ with attributes as $\{P_1 \ldots P_n\}$. Again we can produce subset such as $\{P_1 \ldots P_3\}$ or $\{P_1\}$ to derive grouping of nodes based on their attributes as show in table2. For instance the port ranges on the nodes can be used to derive grouped nodes. Let us consider alerts generated by the TCP-Scan. The TCP-Scan can be used to determine if the targeted system is windows. The intruder can obtain this knowledge about the targeted host by performing TCP-Scan and by observing that a NetBios session running on port 139 is open, this information is the characteristic of windows system. The nodes can be grouped together based on port value, in this case port 139.

IP address	Throughput
Port number	IP packet content
Time aggregate	type

Table1. The table shows the list of attributes for a node.

Source IP	IP flag
Source Port	Reset flag 1
Destination IP	Reset flag 2
Destination Port	Urgent flag
Protocol	Ack flag
Time	Push flag
Date	Syn packet flag
IP header length	Fin flag
IP datagram length	Data Urgent Pointer
Window Size	Ack number
Time to live	Sequence number
Classifiaction	Length of data in segment

Table2. The table shows the list of attributes for a typical alert.

4.3. Moving across different layers

In this section we discuss how to navigate between each intrusion detection layer visually using interactive visualization. The word interactive visualization is often used in different contexts. However in the purpose of our work, we will use it in terms of navigation, and animated transitions. The navigation consists of changing either the viewpoint or the position of an object in a scene. The animated transitions as shown in figure 3 allow viewers to have a much easier time retaining their mental model of an object if changes to its structure or its position are shown as smooth transitions instead of discreet jumps [7]. The security analyst can visualize the alerts at the activity layer in the form of grouped alerts. They will be able to see similar alerts according to similarity rules explained in section 4.2 in same clusters. As shown in figure 3 by using interactive visualization, they will be able to move from alerts clusters to clusters consisting of attacking nodes affecting the internal hosts as shown in the figure (External layer). They can then get further insights by interactively moving into 3r layer (Internal layer) containing internal nodes affected by particular external nodes. The transition from 3rd layer to 4th layer (individual layer) as shown in figure 2 will allow real time visualization and will display the connection requests as lines as shown in the figure 3.

The real time visualization environment will create visual representation of the network nodes (internal and external) as glyphs shown in Figure 4. The glyphs will incorporate visual attributes related to network nodes such as number of users, system load, status, and unusual or unexpected activity. The visual attributes will be easily interpretable for their actual meaning as described by Shneiderman (1996). For instance initial connection request from an attacking host will be represented as a single line as shown in Figure 4b. After connection has been established the single line will be replaced by a line indicating type of connection that has been established as shown in Figure 4 (d,e,f,g,h). The arrows in the lines contain the direction of the connection. The lines containing no arrows will represent unsuccessful connection attempts for different connection types as shown in Figure 3b.

The nodes and the lines will remain active until the user will terminate the session, at which point the node and lines will degrade and fade. The degradation will provide temporal relationships and assistance in making events more persistent. As many events, such as port scans, are instantaneous, it's important to ensure that events exist on screen for a sufficient duration.

The Figure 4 shows different connection types as lines. Figure 4d shows telnet and rlogin connections as solid lines.

The Figure 4e shows ftp sessions as dashed lines. The figure 4f indicates accessing NFS (Network file System). The figure 4g indicates suspicious connections without known connection type. The figure 4h shows the representation of a port scan. The figure4a represents number of incoming connections from an external node to a local host. The inner rings define the system load. The figure 4b represents the initial connection requests. The figure 4c indicates connections initiated by the external node indicated by solid and dashed arrow.

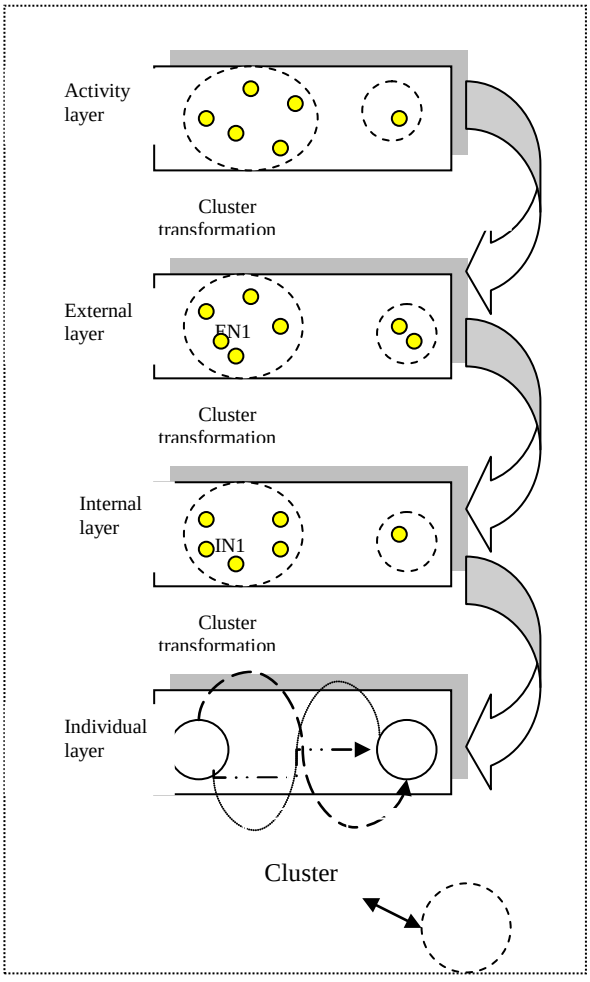

Figure3: The figure shows interactive visualization to move between activity, external, internal, and individual layers interactively.

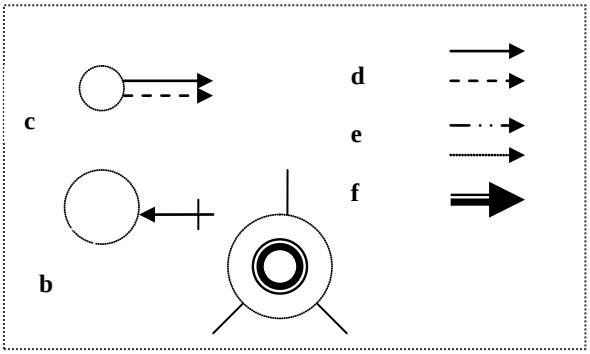

Figure4: The figure above shows the basic glyph organization for the real time visualization.

Next section discusses how we visually represent these graphically clustered nodes.

5. Clustered visualization

Once all nodes in the top three layers are grouped into clusters, we will use clustered visualization technique to visually present these clusters. Our visualization technique will allow users to interactively manipulate data in four levels of visual structures, including graph level, clustering level, abridgement level, and finally picture level. Next we discuss each of these visualization levels.

5.1. The graph level

In our visualization we use a graph model to represent alerts and other network related entities. A graph in our visualization is a classical undirected graph, consisting of *nodes* and *edges*. In our application it is likely a very large graph, containing hundreds or thousands of network entities and alerts. The graph may be dynamic, that is, the node and edge set may be changing; these changes may be a result of user interaction through the visualization, or they may be changed asynchronously by an outside agent.

5.2. The clustering level

Our visualization will use clusters to represent set of groups of nodes and alerts based on variety of similarity rules as defined above. The use of clustered visualization will significantly reduce a visual complexity of displaying large amounts of network related data. A *clustered graph C = (G, T)* consists of an undirected graph *G = (V, E)* and a rooted tree *T* such that the leaves of *T* are exactly the vertices of G as mentioned in a paper by Eades and Huang (2000). Figure 5 shows a clustered graph. Our clustered

visualization can operate on a clustered graph *C= (G, T)* by two basic operations, *create* and *destroy* a cluster. Both can be performed by user interaction, or by an algorithm attached to the visualization.

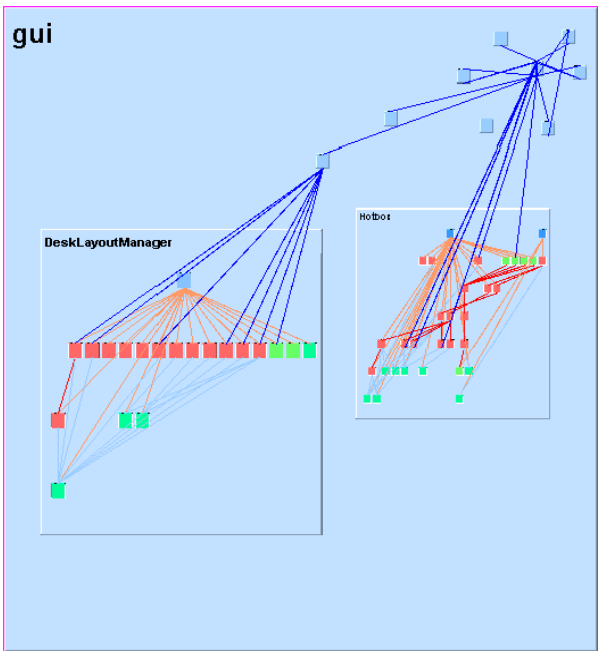

Figure5: A clustered visualization consists of ten clusters.

5.3. The abridgement level

The current Intrusion Detection systems lack in display focus. They do not consider user orientations and focuses of large amounts of data. The use of abridgement level in our visualization provides users with a focus+context viewing mechanism for navigating the large cluster tree. In many cases, the whole clustered graph is too large to show on the screen; further, it is too large for the user to comprehend. In our implementation, the clustered visualization draws only an ``abridgement'' of the entire clustered graph at a time.

We now give a formal definition of ``abridgement''. Suppose that *U* is a set of nodes of the cluster tree *T*. The sub-tree of *T* consisting of all nodes and edges on paths between elements of *U* and the root is called the *ancestor tree* of U. A clustered graph *C' = (G', T')* is an abridgement of the clustered graph *C = (G, T)* if *T'* is an ancestor tree of *T* with respect to a set *U* of nodes of *T* and there is an edge between two distinct nodes *u* and *v* of *G'* if and only if there is an edge in *G* between a descendent of *u* and a descendent of *v*. Our visualization has two elementary operations on abridgements; these change the basis of the abridgement. They are *open* a cluster and *close* a cluster.

5.4. The picture level

The picture level is concerned with the geometric position of the clustered graphs and its associated clustered trees as well as visual attributes to be associated to each displayed entity. Picture level of a clustered graph is shown in Figures 5. More formally, a *picture* of a clustered graph *C = (G, T)* contains a location *p(v)* for each vertex *v* of *G* and a route *c(u, v)* for each edge *(u, v)* of *G*, in the same way as drawings for classical graphs. Further, a picture has a region *b(v)* of the plane for each cluster *v* of *T*, such that if *u* is a leaf of *T* then *b(v)* is located at *p(v)*, and if *u* is a child of *v* in *T* then *b(u)* is contained in *b(v)*. The regions currently used by our visualization are rectangles.

Our visualization provides the usual operation of manually *moving* nodes in a picture. However, the main role of our visualization is animated automatic *layout.*

5.5. The force model

We use force-directed algorithm to draw the pictures of clustered graphs. It is based on the OFDAV force model described by Eades and Huang (2000). Our visualization has three types of spring forces:

- **Internal-spring-** A spring force between a pair of vertices in the same cluster.
- **External-spring-** A spring force between a pair of vertices in different clusters.
- **Virtual-spring-** A spring force between a vertex and a virtual (dummy) node along a virtual (dummy) edge.

As well as spring forces, between each pair of nodes there is a gravitational repulsion force. The forces are applied additively to give an aesthetically pleasing layout of the graph. The sum of forces on each node is continually computed, and the movement of the nodes according to the forces drives the animation, as with the visualization.

5.6. Animations

In our implementation, the whole visualization is fully animated. Every transition, whether triggered by the user, visualization, or by another agent, has its own specific animation. This greatly reduces the cognitive effort of the user in recognizing the new view and change; we aim for a

full preservation of the user's ``mental map". More specifically, there are eight types of animation that are implemented in our system. Five of these are specifically related to the clustering.

6. Conclusions

The current information visualization tools used for intrusion detection concentrate on the monitoring phase of intrusion detection with only limited support for analysis. They do not provide any solution to the large scale visualization problem. Our approach will allow the exploration of large amount of intrusion data in the form of clustered visualization and will support the entire process of detecting intrusions and follow up analysis activities to drill down and examine the attack activity in four separate layers with more detail. The visualization on each layer will provide users with different view of the network attacks.

7. Future work

We are currently developing our initial prototype to prove the concepts and our visualization technique. This will be used as a platform for further work. We will further conduct usability study to determine further improvements of the initial prototype that will address the current limitations in intrusion detection monitoring and analysis tools.

References

[1] Nyarko, K., Capers, T., Scott, C., and Ladeji-Osias, K., (2002): "Network Intrusion Visualization with NIVA, an Intrusion Detection Visual Analyzer with Haptic Integration", Proc. 10th IEEE Symp On Haptic Interfaces for Virtual Environments & Teleoperator Systems, Wahington DC, USA, 277-283.

[2] Global Security Survey, at http://www.deloitte.com/dtt/cda/doc/content/Global%20Security%20Survey%202003.pdf. Accessed 29 July, 2005.

[3] Koziol, J., (2003): Intrusion Detection with Snort, Chapter 1, pages 2-5, SAMS.

[4] Shneiderman, B., (1996): The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations. Proc. IEEE Conference on Visual Languages, Washington, DC, USA, 336-443, The IEEE Computer Press.

[5] Prelude: an open source, Hybrid Intrusion Detection System, Technical Documentation, http://www.prelude-ids.org. Accessed 20 July 2005.

[6] Erabacher, R., (2002): Intrusion and Misuse Detection in Large-Scale Systems. Proc. IEEE Transactions on software Engineering, Texas, USA, 38-48, IEEE Computer Press.

[7] Huang, M., and Eades, P., (2000): Navigating Clustered Graphs using Force-Directed Methods. Journal of Graph Algorithms and Applications, **4**:157-181, ISSN 1526-1719.

[8] Janet, S., Tollis, Ioannis., (2001): Effective Graph Visualization Via Node Grouping. IEEE Symposium on Information Visualization, Washington DC, USA, 51-58. IEEE Computer Society.

[9] Cuppens, F., and Miege, A., (2004): Alert Correlation in a Cooperative Intrusion detection Framework, IEEE Symposium on Research in Security and Privacy, IEEE Press.

[10] Solka, J., L., and Marchette, D., J., (2002): Functional Analysis of Computer Network Data, appeared on Proc of Interface, Tornoto CA, 180-188, University Press.

[11] Girardin, L., and Brodbeck, D., (1998): A visual approach for monitoring logs, in proc of the 12th usenix conference on system administration, Berkley CA, 299-308, USENIX Association.