Visualizations to Improve Reactivity Towards Security Incidents Inside Corporate Networks

Patrick Hertzog
NEXThink S.A.
Scientific Park, PSE-B
CH-1015 Lausanne, Switzerland
patrick.hertzog@nexthink.com

ABSTRACT

Corporations are usually not able to react quickly enough toward security incidents because their security staffs are flooded by information difficult to interpret. To help them in their task, we propose a new approach to build efficient visualizations based on more pertinent information. Fewer but higher-level parameters are collected on the endpoints and then centralized on the network. We also present an interactive grouping method to overcome the problem of the amount of data to display. Finally, two complementary visualizations based on that approach are described along with usage scenarios that illustrate their usefulness.

Categories and Subject Descriptors

H.5.2 [**Information Systems**]: Information Interfaces and Presentation—*User Interfaces*; C.2.0 [**Computer - Communication Networks**]: General—*Security and Protection*; C.2.3 [**Computer - Communication Networks**]: Network Operations—*Network Monitoring*

General Terms

Security

Keywords

reactivity, reaction time, security, connection visualization, alarm visualization, parallel-coordinates, starfield

1. INTRODUCTION

Security incidents impact not only the corporate network where they occurred but, even worse, the impacted network can be in its turn the source for attacking another target (*i.e.*, another network) and cause harm there. The legal responsibility of the corporation can then be claimed unless it can be shown that sufficient measures have been taken to be able to report such incidents to authorities within an acceptable time period. The reaction time towards security incidents is therefore an important factor in risk management and interests greatly people in corporations up to the executive level. From a more technical point of view, one can observe that major incidents are usually preceded by some heralding signs. Being able to react soon enough may enable security staffs to take early measures preventing the major incident. The reaction time towards security incidents can actually be separated in three distinct phases:

detection Time period between the moment when the incident occurred and when it is actually detected.

decision Time needed to evaluate the incident and its characteristics (*e.g.*, severity, involved resources) and to decide, if necessary, which action to undertake. It cannot be carried out autonomously by a security system because a lot of technical as well as non-technical parameters have to be taken into account.

action Time needed take the action (*e.g.*, put in place remediation, report to authorities) chosen in the *decision* phase.

Today, both the *detection* and the *decision* phase lengthen considerably the reaction time because the security administrator is usually flooded by non-pertinent information. We claim that adequate visualizations can help her/him to take quicker and better decisions.

Several tools based on information visualization have appeared during the last years; they can be divided in different groups based on their information source. Tools such as visFlowConnect [12], nVisionIP [9], RUMINT [4] or TNV [5] use network flows or packet inspection. Applications like MieLog [10] or Tudumi [11] use logs collected directly on the endpoints. Based on raw information, those two approaches try to help the administrator during the *detection* phase. On the other hand, RainStorm [1], SnortView [8] or STARMINE [6] visualize alarms generated by traditional security systems (*e.g.*, IDS[1]): they are thus useful during the *decision* phase.

Unfortunately, computer or network logs provide unmanageable amounts of low-level data which are too far from the decision to be exploitable: human operators reason more in terms of users and applications than, for example, in terms of bits in the payload of network packets. On the other hand, traditional security systems fail to provide pertinent alarms due to either the amount of generated alarms or their ratio of false alarms.

[1]Intrusion Detection System

In the following sections of this paper, we will describe the approach we propose to overcome the challenges highlighted above in order to build effective visualization-based tools and then propose two complementary visualizations based on that approach. The presented work is currently commercialized by NEXThink[2] in its *REFLEX* solution.

2. APPROACH

In order to reduce the reaction time, our goal is to shorten the *decision* phase by improving the knowledge about the network in general and about what is going on inside the network in particular. However, as seen in Section 1, we are facing two major challenges: the lack of pertinent information and the amount of data to display.

2.1 Focusing on pertinent information

Current systems are based either on logs (computer logs or network flows) or on alarms generated by traditional security systems. In both cases, collected information is made of a high number of low-level parameters that are difficult to interpret by the security administrator: she/he has to do a lot of inferences to transform those low-level parameters into information useful to take a decision. This lengthens considerably the process and moreover it introduces some incertitude because assumptions have to be done. For instance, one can have a HTTP flow on port 80 with some headers indicating that it may be *Internet Explorer* but no one can be sure that it is not a malicious application trying to pass itself off as *Internet Explorer*.

Our approach is to obtain only a small set of certain high-level parameters for each connection on the network.

Definition 1. *A* Connection *is the tuple* <time*,* user*,* source host*,* application*,* destination port*,* destination host> *where*

- time *is the time stamp at the beginning of the connection,*
- user *is the user who triggered the connection (*e.g.*, the user can be identified by the SID on Windows or the UID on UNIX-based operating systems),*
- application *is the application (we mean the binary,* e.g.*, firefox.exe, not the protocol,* e.g.*, HTTP) used to initiate the connection,*
- port *is the port used on the target computer and*
- source / destination hosts *are the identification of the source and destination computers of the connection.*

We collect those parameters on the endpoints and we centralize them on the network where they are correlated and analyzed by an engine based on artificial intelligence to produce, if necessary, comprehensive alarms. The techniques used for the analysis go beyond the scope of that paper and will therefore not be explained here.

[2]NEXThink develops and markets *REFLEX* Endpoint Behavior Manager based on research initially conducted at the Artificial Intelligence Lab of the Swiss Federal Institute of Technology (EPFL) in Lausanne, Switzerland. http://www.nexthink.com

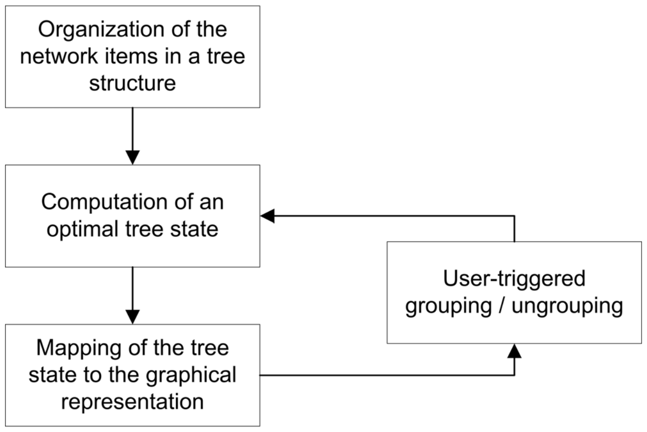

Organization of the network items in a tree structure
Computation of an optimal tree state
Mapping of the tree state to the graphical representation
User-triggered grouping / ungrouping

Figure 1: Process to display grouped network items in visualizations

2.2 Visualizing big amounts of data

Even if we are collecting a small set of high-level information (as described in Section 2.1), we still have a huge amount of information to display: a corporate network with thousands of users using several hundreds applications can generate more than a million connections per day. It is obvious that it is not feasible to display all the information with a fine granularity and therefore we need to be able to group similar information to display information with a coarser granularity. Moreover, one still needs to be able to interactively drill down to get more precise information.

Our proposed grouping technique for visualizations enables to display individual network items (*i.e.*, users, applications, hosts and ports) or groups of network items. Network items can be interactively grouped (and inversely) according to similar characteristics. For instance, all applications of the same category (*e.g.*, browsers) can be grouped to form a meta-application (*e.g.*, representing all browsers). Meta-applications can be grouped together to form a higher order group (*e.g.*, browsers and e-mail clients grouped into internet applications) and so on.

The whole process is described in Figure 1. First, the different network items have to be organized in a tree structure. Each node of the tree corresponds to a similar characteristic of its children elements and the leaves contain the actual network items. Each node also has an attribute *state* indicating if it is collapsed (*i.e.*, its children will be displayed as a group) or expanded (*i.e.*, its children will be displayed individually). One can choose any characteristics for the nodes as long as one network item is only represented by one leaf of the tree. However, the more balanced the tree is, the better the result will be. Then, an optimal tree state is computed. Computing a tree state means setting the *state* attribute of each node in order to respect some constraints:

- the screen real estate is limited and each displayed network item (or group of network items) needs a minimal amount of space;
- the maximum number of network items (or groups of network items) should be displayed;
- the actions taken by the user (*i.e.*, manual grouping / ungrouping) have to be respected.

Once an optimal tree state has been computed, it can be mapped to the actual graphical representation. It means that each leaf whose *state* of its parent node is *expanded* will be displayed as an individual network item. On the other hand, each *collapsed* node whose *state* of its parent node is *expanded* will be displayed as a group of network items.

3. VISUALIZING CONNECTIONS

Security systems can generate alarms and to be able to take the right decision, the security administrator has first to understand them, *i.e.*, understand why the alarm has been generated and which items (*i.e.*, users, applications, hosts and ports) are involved. In most systems, the generation of alarms can be either rule-based or anomaly-based. Rule-based alarms should be easy to understand as they are triggered when a connection (or a set of connections) respects a precise rule (*e.g.*, the signature of a known attack). Understanding anomaly-based alarms is another story: security systems often use sophisticated algorithms taking into account a large number of parameters whose output cannot be directly interpreted by the human operator. In that situation, we claim that if one wants to understand what is abnormal, one needs first to understand what is normal. We need then to be able to visualize the normal behavior (*i.e.*, normal connections) along with the alarms.

3.1 Description

As described in Definition 1, connections are composed of different independent parameters. A good way to display such data is to use a parallel-coordinates visualization [7]. This kind of visual representation has already been used in the field of security, for example in [12]. It is composed of a set of parallel axes, each one representing one parameter of the connection. Their order can be chosen according to the situation and some of them may be omitted if required. On each axis, the possible values of the corresponding parameter are displayed. Each connection is represented by a polygonal line linking the values on the different axes. Although they seem to be an ideal solution, parallel-coordinates visualizations have several challenges that need to be addressed.

3.2 Challenges and Solutions

When displaying a lot of information, parallel-coordinates visualizations become quickly unreadable and therefore unusable. They are also an easy target for deception attacks based on occlusion and jamming as explained in [3]. Those issues are due to the number of lines concurrently drawn on the graph and to reduce that number, we use the grouping method we explained in Section 2.2. The second technique is to draw semi-transparent lines instead of solid lines. That way, lines cannot be completely hidden by other lines.

The second challenge is more technical: in a network composed of thousands of users, applications and computers, the amount of connections is so tremendous that it is very difficult to handle them individually: it poses problems in terms of memory and processing time. By looking closer to those connections, we can however notice that a lot of them differ only by the time. We can therefore make some abstractions:

Definition 2. *A* Signature *is an abstraction of connections sharing the same parameters except the time. It can be seen as the tuple* <user, source host, application, port, destination host>.

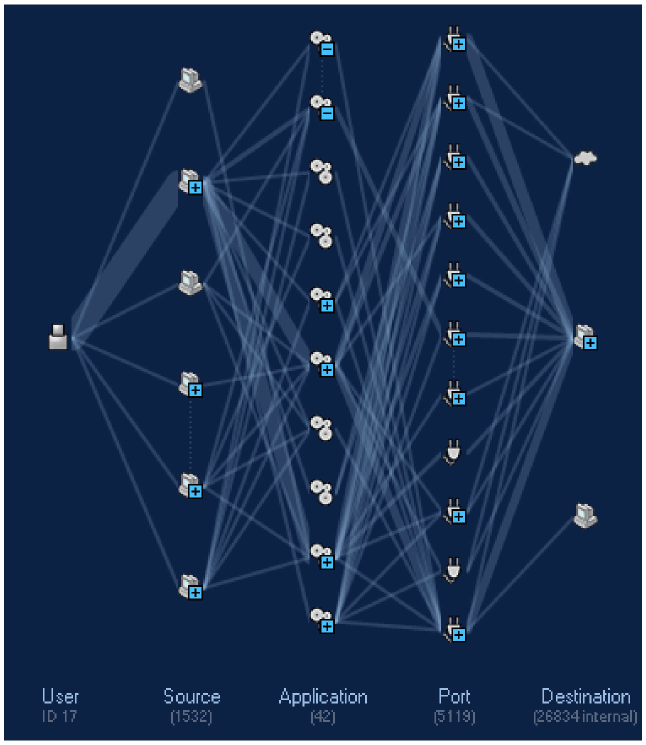


Figure 2: Parallel-coordinates visualization displaying 340,853 different signatures corresponding to more than 17 millions connections.

Definition 3. *A* Signature Activity *is the tuple* <signature, nbConnections> *where* nbConnections *is the number of connections associated to the corresponding signatures over a period of time.*

To reduce the amount of information to deal with, we focus on signature activities instead of connections. Therefore, in our parallel-coordinates visualization, the signature is displayed as a polygonal line and the corresponding number of connections is represented by the thickness of the line.

Thanks to these techniques, we are able to build parallel-coordinates able to display a big number of signatures as shown in Figure 2.

3.3 Interaction

Displaying a visualization is not enough: to take full advantage of it, one needs to be able to interact with it. The first way to interact with our parallel-coordinates is to move the mouse cursor over the different items to highlight their associated connections. For instance, if one places the cursor over an application, all connections made with that application are instantly highlighted. If one places the cursor over a segment between a source host and an application, only connections made with that application from that host are highlighted. In order to summarize what is being highlighted, the number of corresponding items are displayed under the axis labels. For example, in Figure 3 (middle image), one moved the cursor over the *office applications* group and one can directly see that there are five such applications used by only one user from a single computer and that they are using four ports on four internal destinations.

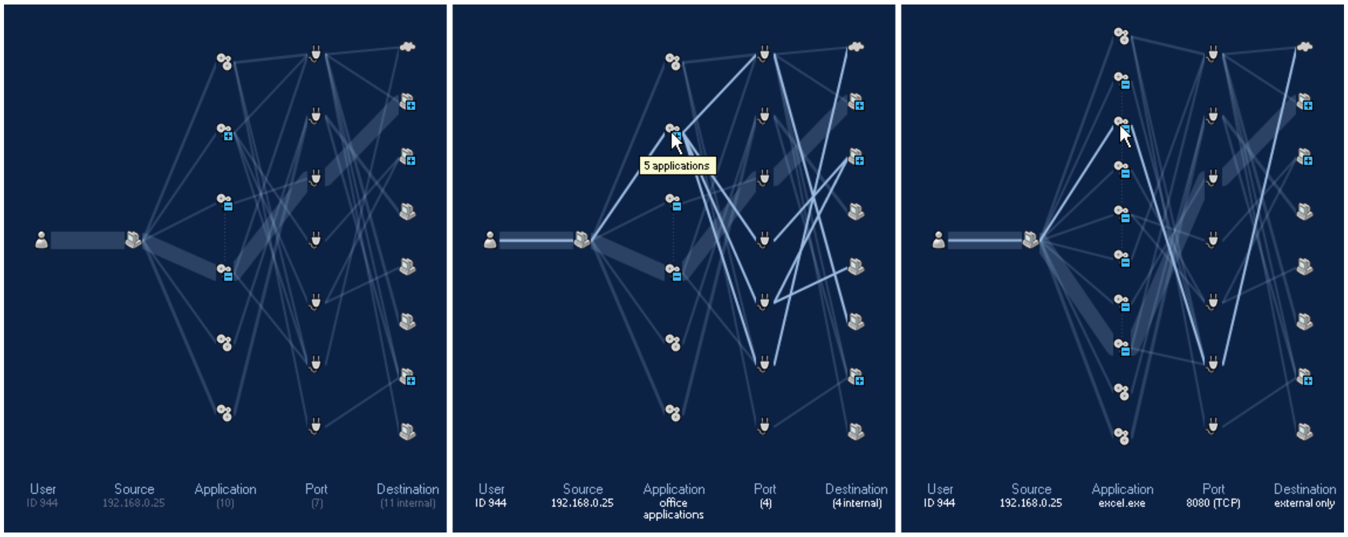


Figure 3: A set of connections are visualized on the left image. The mouse cursor is moved on the group of office applications and the corresponding connections are highlighted (middle image). On the right image, the office applications are ungrouped to provide a more detailed view.

In Section 2.2, we described a way to group similar items in a visualization. Implemented in the parallel-coordinates, this feature offers the possibility to interactiveley drill down (*i.e.*, ungroup clusters of items) to have more precise information or, on the opposite side, to zoom out (*i.e.*, group items) to have a better overview and a less crowded visualization. This is illustrated in Figure 3 (right image).

Finally, one can also select a specific item by clicking on its corresponding icon or a specific set of connections by clicking on the corresponding path. Doing so allows the security administrator to get more information and access other parts of the product. As it goes beyond the scope of that paper, we will not give further details.

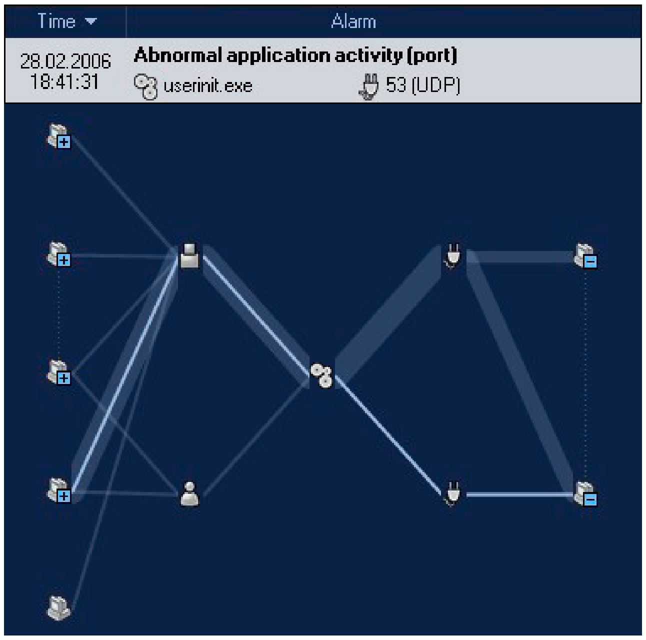
3.4 Usage Scenarios

3.4.1 *Why was this alarm generated?*

A part of the duties of security administrators is to analyze alarms generated by security systems to assess if a remedial action is needed or not (this is part of the *decision* phase described in Section 1). This is a particularly tedious work especially with behavior-based security systems that generate alarms when an anomaly is detected. To take the proper decision, the adminstrator has to understand why the event has been considered as abnormal. Let us imagine the situation where the administrator receives an alarm stating that a particular application has a strange behavior due to the port it used (see Figure 4). With the help of the visualization, she/he can quickly and easily understand the alarm and see which resources have been involved.

3.4.2 *Who is using the port 22?*

Firewalls protect corporate networks but to be able to communicate with the outside, security administrators have to make holes inside them (typically by opening ports). However they would like to know what is going through those holes to control that no malicious activity is taking advantage of them. The parallel-coordinates visualization can help the security administrator in that task as well. For



Time	Alarm
28.02.2006 18:41:31	**Abnormal application activity (port)** userinit.exe 53 (UDP)

Figure 4: One can clearly see that this application usually makes connections through another port.

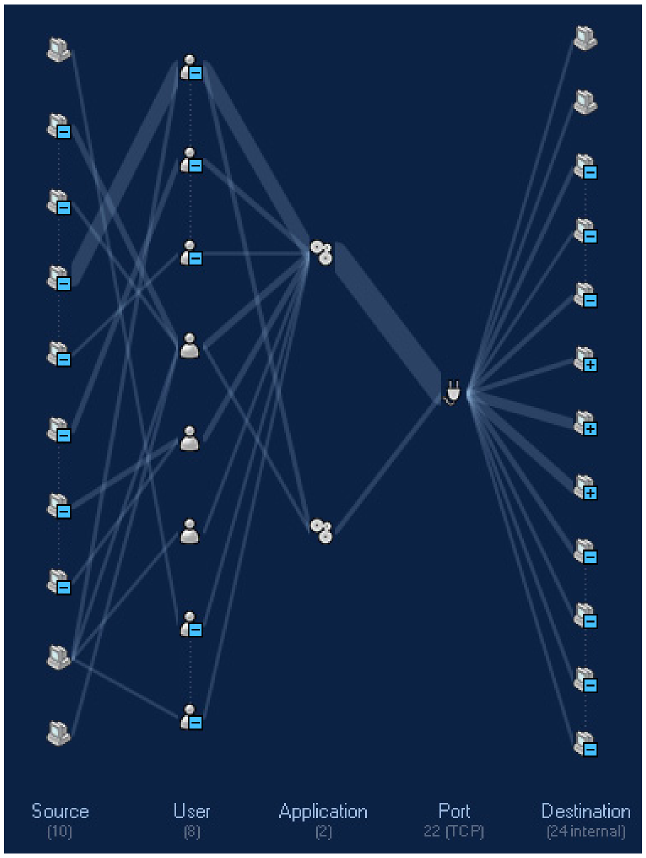


Figure 5: The port 22 is often open for SSH communication but who is using it, with which application and to connect to which computers? If one wants to close that port, who will be affected in the network?

instance, the port 22 is often open for SSH communication but who is using it, with which application and to connect to which computers? If one wants to close that port, who will be affected in the network? All those questions can be easily answered with the visualization depicted in Figure 5.

4. VISUALIZING ACTIVITY AND ALARMS OVER TIME

Parallel-coordinates visualizations described in Section 3 can display a set a connections. They can therefore visualize the activity of one or several network items during a certain period of time but, as time is not represented in parallel-coordinates, it is not possible to assess this activity over time. They are also very useful to help the administrator understand single alarms generated by security systems. However correlations may exist between alarms (*e.g.*, the same alarm occurring every 2 hours or several alarms involving different browsers) and it is quite difficult to discover them just by displaying alarms in a textual list. An adequate visualization can help to overcome these issues.

4.1 Description

We based this second visualization on a starfield (see [2]) that we modified to accommodate our needs. In our visualization, time is one axis of the chart, the other being the network items of the same type (*i.e.*, either applications, users or sources). The time axis is sampled with a certain granularity: if T is the time period represented on the axis and n the number of samples over that period, it means that the axis is divided in n portions, each one representing a duration of T/n (it is a case of linear granularity but other kinds of granularity may be defined). Therefore, for each network item, a set of n rectangles is defined, each rectangle representing the activity level of the corresponding item during the related time sample by the mean of a color gradient (*i.e.*, the brighter the rectangle is, the more active the item has been). If an alarm occurred during that time sample, the corresponding rectangle reflects it by changing its color to a color corresponding to the severity of the triggered alarm. Moreover, the chart has been augmented with additional bar charts summarizing the alarms either by network item(s) (at the right) or by time sample (at the top). These two additional graphs allow the security administrator to see in an eye-blink if there has been any peak of alarms either for a particular item or at a certain time.

In this visualization, we also have to deal with a lot of data: in a network with thousands of users, applications and computer, we cannot afford to dedicate one row of rectangles per individual item. We can therefore use our grouping approach explained in Section 2.2. That way, one row can not only represent one single item (*e.g.*, one application such as *Firefox*) but also a group of items (*e.g.*, one group of applications such as *browsers*).

With such a visualization, one can easily detect correlations between alarms. Let us see an example: in Figure 6, the activity of one day is displayed from the application perspective (*i.e.*, rows represent applications). The size of the time sample is 15 minutes thus each row is divided in 96 rectangles. Ten alarms have been generated during that day and they do not seem correlated: they are spread out over the graph and no sensible peak is visible in the bar charts. However, if one display the same day of activity but from the source perspective as in Figure 7, one can clearly see a correlation: most of the alarms generated that day were due to the same source. It is then evident that full attention has to be focused on that source and that some action will be required.

As it displays activity over time, this visualization can also help the security administrator to detect suspicious events. For instance, the activity of one particular user is displayed in Figure 8; each row corresponds to one application she/he used during that day. One can see that the application *proxy.exe* has been used very early in the morning. Was the user already working at that time? Was someone else running that application under the user's identity? To assess the situation, the security administrator has to investigate, for example, by displaying the connections done by *proxy.exe* in parallel-coordinates visualization to see from which computer it was used and to connect to which destination hosts.

4.2 Interaction

The first way of interaction is to move the mouse cursor over the visualization. When the cursor hovers over a rectangle, a tooltip is displayed showing an activity summary of the network item during the corresponding time sample and when it hovers over the additional bar charts, a summary of the alarms is shown. For example, Figure 9 (left image) illustrates one day of activity for applications on a

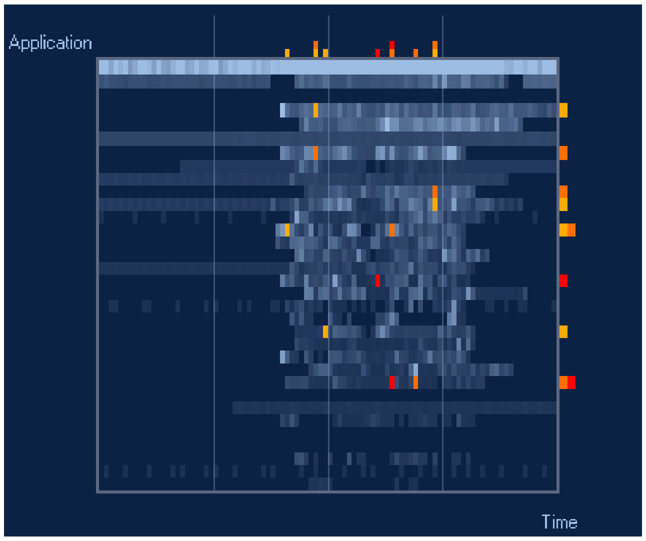


Figure 6: The activity of one day is display from an application perspective. It seems that there is no correlation between alarms.

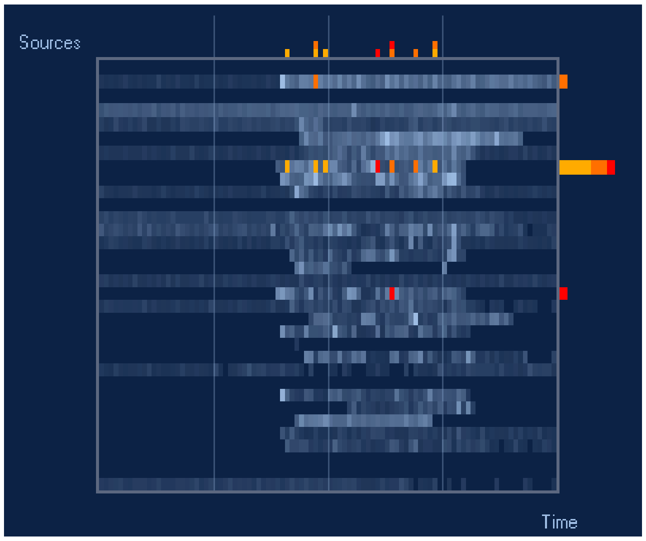


Figure 7: The same day of activity as in Figure 6 but from a source perspective: a clear correlation appears.

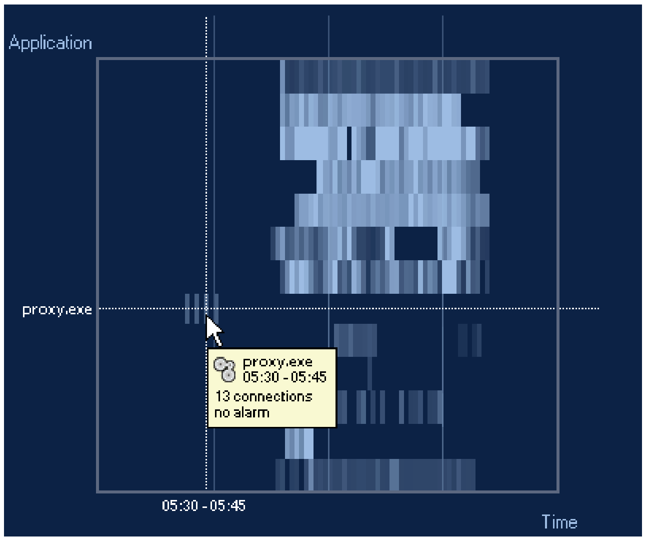


Figure 8: The activity of one particular user is displayed from an application perspective (*i.e.*, each row corresponds to one application used on a particular day). Is this user working so early in the morning? Or is it a suspicious activity?

network. The mouse cursor has been placed on the rectangle representing the activity of the application *scan.exe* from 6:30 to 6:45 in the morning. The tooltip indicates that this particular application has generated an alarm of *type silent port scanning* during that period of time. To confirm that alarm, one can directly open a parallel-coordinates visualization displaying all connections of *scan.exe* (see Figure 9, right image).

As in the parallel-coordinates visualization, one can interactively group and ungroup items in order to either have a better overview or to have more precise information.

5. CONCLUSION

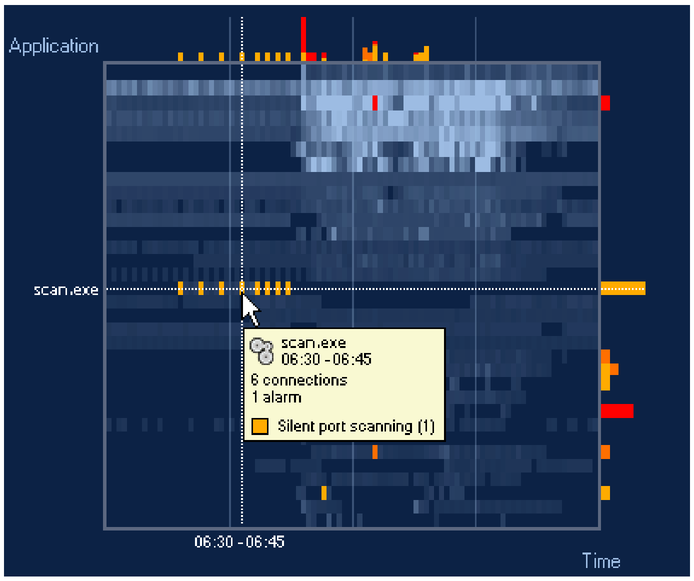
In this paper, we proposed a new approach to build visualizations based on focusing on pertinent information and a grouping method. We also described two different and complementary visualizations that are currently commercialized by NEXThink in its *REFLEX* suite. According to customers' testimonies, those visualizations turn out to be effective in helping security staffs to better understand what is going on in their networks and to react quicker and in a more appropriate way.

6. REFERENCES

[1] K. Abdullah, C. Lee, G. Conti, J. A. Copeland, and J. Stasko. Ids rainstorm: Visualizing ids alarms. In *Proceedings of the IEEE Workshop on Visualization for Computer Security (VizSEC 2005)*, Minneapolis, MN, USA, October 2005.

[2] C. Ahlberg and B. Shneiderman. Visual information seeking: Tight coupling of dynamic query filters with starfield displays. In *Human Factors in Computing Systems. Conference Proceedings CHI'94*, New York, NY, USA, 1994.

[3] G. Conti, M. Ahamad, and J. Stasko. Attacking information visualization system usability:



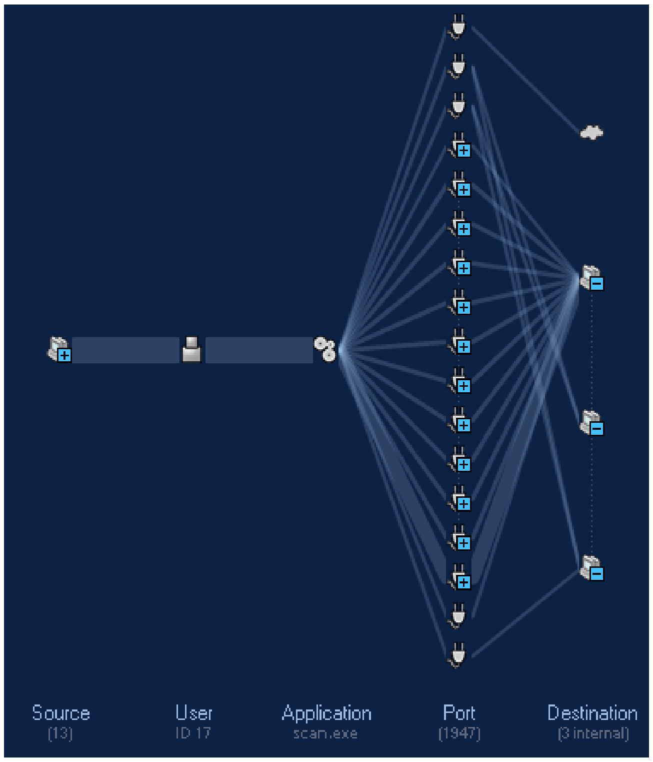


Figure 9: Moving the mouse cursor over a rectangle provides additional details in a tooltip (left image). The selected application can be visualized using a parallel-coordinated visualization and we can clearly recognize the port scanning (right image).

Overloading and deceiving the human. In *Proceedings of the Symposium On Usable Privacy and Security (SOUPS 2005)*, Pittsburgh, PA, USA, July 2005.

[4] G. Conti, J. Grizzard, M. Ahamad, and H. Owen. Visual exploration of malicious network objects using semantic zoom, interactive encoding and dynamic queries. In *Proceedings of the IEEE Workshop on Visualization for Computer Security (VizSEC 2005)*, Minneapolis, MN, USA, October 2005.

[5] J. R. Goodall, W. G. Lutters, P. Rheingans, and A. Komlodi. Preserving the big picture: Visual network traffic analysis with tnv. In *Proceedings of the IEEE Workshop on Visualization for Computer Security (VizSEC 2005)*, Minneapolis, MN, USA, October 2005.

[6] Y. Hideshima and H. Koike. Starmine : A visualization system for cyber attacks. In *Proceedings of Asia-Pacific Symposium on Information Visualization (APVIS 2006)*, Tokyo, Japan, February 2006.

[7] A. Inselberg. The plane with parallel coordinates. In *The Visual Computer*, pages 69–91, 1985.

[8] H. Koike and K. Ohno. Snortview: Visualization system of snort logs. In *Proceedings of the ACM Workshop on Visualization and Data Mining for Computer Security (VizSEC/DMSEC '04)*, Washington, DC, USA, October 2004.

[9] K. Lakkaraju, W. Yurcik, and A. J. Lee. Nvisionip: Netflow visualizations of system state for security situational awareness. In *Proceedings of the ACM Workshop on Visualization and Data Mining for Computer Security (VizSEC/DMSEC '04)*, Washington, DC, USA, October 2004.

[10] T. Takada and H. Koike. Mielog: A highly interactive visual log browser using information visualization and statistical analysis. In *Proceedings of the Sixteenth Systems Administration Conference (LISA '02)*, Berkeley, CA, USA, November 2002.

[11] T. Takada and H. Koike. Tudumi: Information visualization system for monitoring and auditing computer logs. In *Proceedings of the Sixth International Conference on Information Visualisation (IV'02)*, London, England, UK, July 2002.

[12] W. Yurcik. Visflowconnect-ip: A link-based visualization of netflows for security monitoring. In *Proceedings of the Eighteenth Annual FIRST Conference on Computer Security Incident Handling*, Baltimore, MD, USA, June 2006.