An experimental tool for visual data mining

Leen van Dalen and Willem G. Knoop

{leen,w.g.knoop}@xirion.nl

Introduction

In this paper, we discuss a visualization tool and how it can be applied to analyze log data. The intended audience is system administrators, who need to deal with computer log files, analyze resource utilization, network traffic, security issues and performance problems, and people with a general interest in information visualization.
At Xirion, we have been studying the results of research in the field of information visualization and we have done some additional research, which has resulted in a number of prototype tools. Some of these tools are particularly suited to analyze hierarchical data, like file system data or configuration data provided by directory services. In this paper, we will focus on a tool that is more suited to analyze event data. The source of the data can be arbitrary. The only assumption we will make is that each event has an accurate timestamp, which provides us with a dimension in which the events can be ordered.
Though our primary focus has always been to apply these tools in order to enhance our own skills, we have discovered that their use is certainly not limited to our field. For instance, when we were using our tool to investigate patterns in usage of internet services at a provider's site, we discovered that besides pinpointing a few bottlenecks we could also gather interesting marketing information.
Such experimental usage however, posed additional requirements on our tool since we were dealing with sensitive information. For example, a log of a chat service had users' email addresses as one of its fields. So we needed to be able to anonymize certain attributes of events, in order be able to trace the activities of individuals but without the possibility to reveal their identity (an email address). In general, an authorization scheme which controls access on a per-attribute basis is desirable.
Before we continue to describe our tool, we will make some remarks regarding log file management and give a brief introduction to information visualization in general.

Log file management

This section describes some of the problems associated with traditional log file management (or the lack of it) and identifies a number of criteria we think should be addressed by a policy regarding log file management. Our goal is to achieve easy, uniform and secure access to log data from different sources covering a large time span because it provides us with a valuable source of information. This holds for event log data in a broad sense: error conditions, state transitions, authentication failure/success,

etc. Some of our remarks may also apply to transaction log files, which can sometimes be used in other ways as originally intended.

Applications often have a non-standard logging strategy: centralized logging services (if available) are often ignored, the format of the log messages varies widely between applications and the locations where the log files are stored on the file system may not be configurable. So in practice, log files are distributed over an entire computing infrastructure and even within a single machine they are scattered over the entire file system. Moreover, the diverse formats of log files are often rather primitive: not very compact, inconsistent and insecure.

The size of log files varies, but often comprises large amounts of data. Sometimes the amount of logging an application generates can be tuned. In many cases, only events indicating failure are logged. When examining a typical firewall log, only rejected or dropped connections are logged. There is no information concerning accepted connections. When faced with such a log as the only source of information, valuable context information is missing. However, when maximum logging is chosen, a considerable performance penalty may be incurred. In each case, it should be investigated which amount of information is desirable and feasible. It should also be decided which logs should be archived and how long they should be retained.

The information in log files may be sensitive and may be protected by access restrictions on the file. Care should be taken when transferring the information in the file to other systems; the original access restrictions should be migrated along with the data. With system logs, access is at most times restricted to a system account. It should be noted that the people who can authenticate as such, might be different people on different systems. Therefore, in general, more advanced authentication methods are needed.

Log files are generated because the information they contain may be useful. However, often, log files are not actively scanned at all. Only when a problem occurs, there may be an incidental ad hoc inspection of log files. In this situation, the often-employed log file policy of purging log entries with a certain age or when too much space is occupied may seem quite adequate. However, purging recent records because the log is getting too large may leave you empty-handed.

Another strategy, which is also in common use, is to automatically analyze log files using filters that detect events that require action. This can be very useful, e.g., when a file system is at 90% of its capacity, problems can be expected. A lot of routine checking can be captured in such tests. However, this approach is limited to problems that can be anticipated. Unexpected events (unfortunately often the majority) are filtered out. Such automatic analysis of log files may even lead to a false sense of security.

This paper shows that an unfiltered look at all the events in a log can be beneficial. Unfortunately, the large variation in existing log file formats makes constructing a tool that handles them all difficult. Even if a widely accepted standard log file format was to emerge, we couldn't expect existing applications to be using it in the near future. Therefore, we took the approach of writing converters to convert log files to our own standard format. When we are interested in a new type of log file, we will have to start by

writing a converter for it. Fortunately, for the majority of the log file formats we have coped with, this has been a trivial task.
The standard log file format we have chosen, is the same as which is being used by XiAudit, Xirion's audit product. This is a fairly general, extensible, self-describing format with support for encryption and which efficiently employs compact storage structures. We have several more or less conventional tools to explore data in this format. They can be used to make selections using regular expressions or by using a powerful query language that operates on attributes of the events. Although very useful, these tools suffer from the same problems as filtering the event data or taking an entire log in a text editor: they focus on what is expected; when log files are sufficiently large, interesting or unexpected events will only be discovered by accident.
To recapitulate, a policy regarding log file management should at least cover the following points: it should be determined what information should be logged, how long it should be retained and whether it should be archived. Access to the log data may need to be restricted because the data contains sensitive information and/or to ensure its integrity. In addition, in order to enhance the accessibility of the data, it may be desirable to convert the data to a more efficient, more secure, self-describing format.

Information Visualization

The techniques used in the field of information visualization are meant to graphically present information for which no appropriate analytical model is known. It can be contrasted with the more classical scientific visualizations that are based on a known model and which provide a graphical presentation of a certain phenomenon in order to provide a better insight, to perform simulations, etc.
Information visualization starts with a more chaotic view of the world. It attempts to exploit the human eye to recognize patterns in a graphic presentation of data. Many different presentations are possible, and some will more clearly show certain patterns than others. The key problem is to select the right presentation for a given data set.
In order to enable the user to discover things, a good user interface is a vital requirement for a visualization tool.
A tool must enable a user to switch between different presentations and when faced with a selected view, it must be possible to navigate through the data and focus on phenomena of interest while keeping some feedback on the context of the details being inspected. Since the amount of space on a computer screen is limited, for an overall view the information needs to be compressed. When embedded in the right tool, straightforward visualization techniques have been proven to be quite useful [1,2]. An example of a compressing visualization technique is the information mural [3]. With the mural technique, a large image is squeezed to an image that fits. Each pixel in the destination image has an associated weight. Each pixel from the source image is mapped to one or more pixels in the destination image and contributes proportionally to the weight of the destination pixel. The weight then needs to be translated to a visual attribute (e.g.,

brightness). A space efficient presentation of hierarchical organized data can be achieved using treemaps [4]. Here one attribute of the data is mapped to the size of nested squares that are hierarchically organized.
We have been experimenting with symmetrical histograms [5]. Here the data is compressed using a linear or logarithmic scale. Events that match selected criteria are organized in bands. One of them is put in the center; the others are wrapped around it. Events in the center appear much more prominent, by changing the order of presentation the focus of interest can be changed.

The Xirion Chronoscope Prototype

This section describes the prototype application we have built. The most significant initial requirements to be satisfied by this prototype were:

- It should be possible to display more than 100000 events in one graph.
- No events are allowed to be filtered out.
- The user should be able to zoom into details while keeping a visual reference to the context of these details.
- The user should be able to map attributes of the events to aspects of the visualization.
- The tool should be able to handle arbitrary event data, the only limitation being that each event should have a timestamp.
- It should run on a normal workstation.

As mentioned before, we handle access to events from different sources by a preprocessing phase in which they are converted to a format which our tool can handle. A line from a log file, such as:

```
Jun  8 10:24:54 vanadium ftpd[16613]: FTP LOGIN FROM hafnium.xirion.nl, leen
```

is stored as an event matching the template:

```
<timestamp> <hostname>ftpd[<pid>]: FTP LOGIN FROM <hostname>, <user>
```

Such a template is stored as an *event type* in our log format. Each attribute of the event type also has an associated type, e.g., in this case ‘user’ is of primitive type string. Our tool can retrieve these types from a log and enable the user to manipulate aspects of the presentation of the data for those events that match certain conditions, e.g., assign a color to the ftp login events in general and a different color to login events for a specific user. (Incidentally, note that the example log doesn’t bother to store a year in its timestamp) Figure 1 is an example of a graph where events are colored using a condition on a single attribute of the event data. It provides two views on a firewall log. In the top view, connections coming from one specific source have a different color. In the bottom view,

different protocols are assigned different colors. A peek at ICMP traffic shows up at the same spot as the traffic from the highlighted source in the top graph. In this visualization, each event is represented by a single pixel and events are presented in the order in which they occurred. This simple visualization technique can be very useful to discover certain patterns in event data. However, devoting an entire pixel to each event limits the number of events that can be presented on the screen.

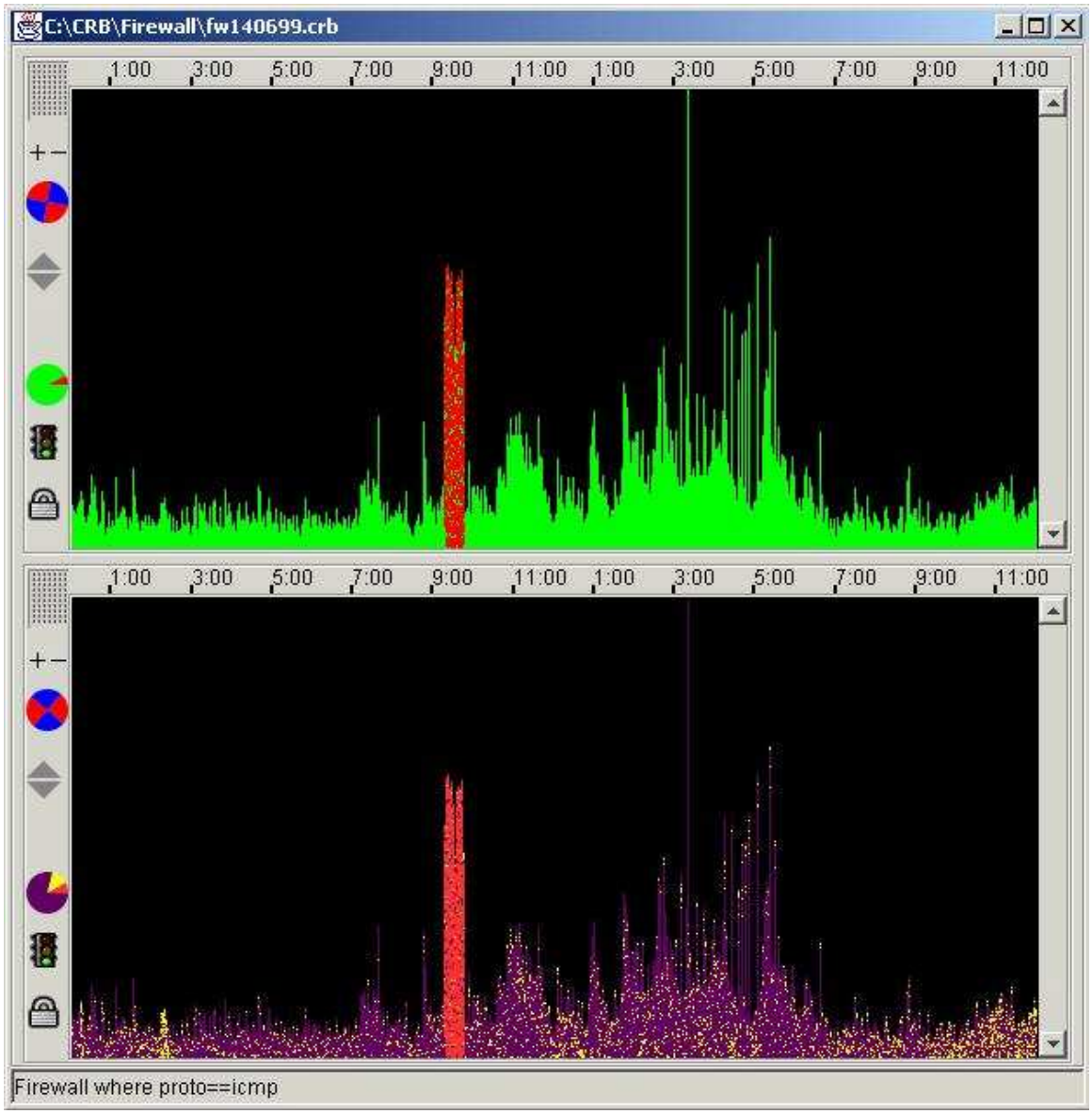

Figure 1. A firewall log

Using this technique, it's impossible to display 2 million events at once using a screen resolution of 1280*1024 = 1310720 pixels, so often compressing visualisation techniques are desirable. On the other hand, when using *only* a single pixel for each event, a single event can easily be missed. Although this simple visualisation technique may not be appropriate to display millions of events at once, it can of course always be used on a subset of the data.
As is shown, each graph is accompanied by a time ruler. When multiple graphs are shown, the interval that is selected can be synchronized by dragging the time ruler from one of the graphs onto another one. This makes it possible to easily identify correlations between events in logs from different sources.

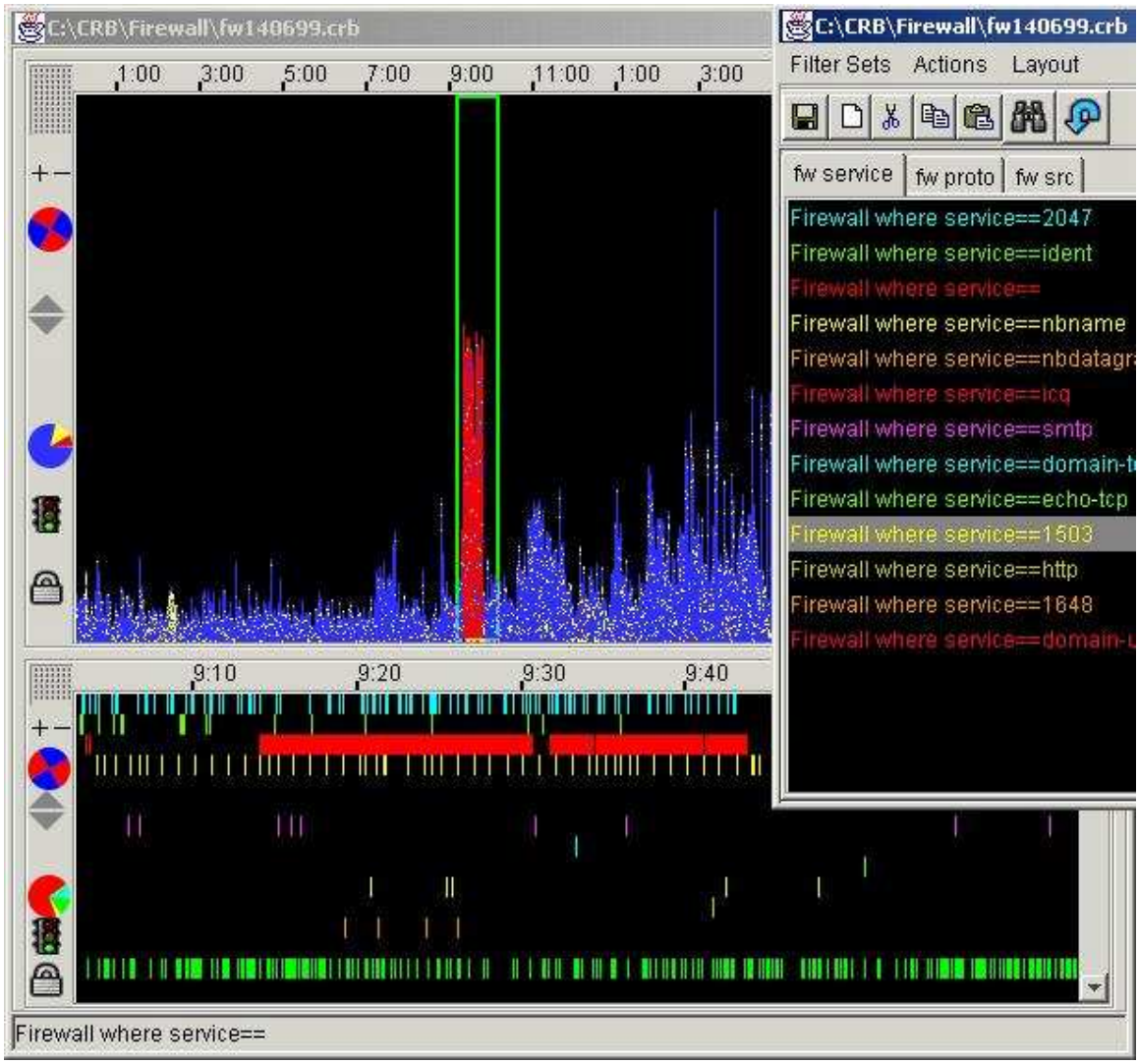

Figure 2. Zooming into details

A user can select a time interval of interest to be shown in a separate graph. The selected interval is indicated in the source graph and it can be interactively moved or changed in size. Multiple levels of subselections are possible. Figure 2 shows such a selection. The interval containing the peek at ICMP trafic is selected in the top graph and shown again in the bottom one. In the bottom graph events are sorted on service type. Events for the 13 most frequent services are shown and the fourteenth band of events on the bottom shows the events for all other services. The ICMP traffic is visible as two bursts having an empty service field.
Note that the order in which the events are shown in the bottom half of figure 2 is the same as the order of the rules shown in the window on top of it. Sometimes it can be useful to arange the rules in a specific order so you can more easily see possible relations between different events that may have triggered each other or which are in some way related.

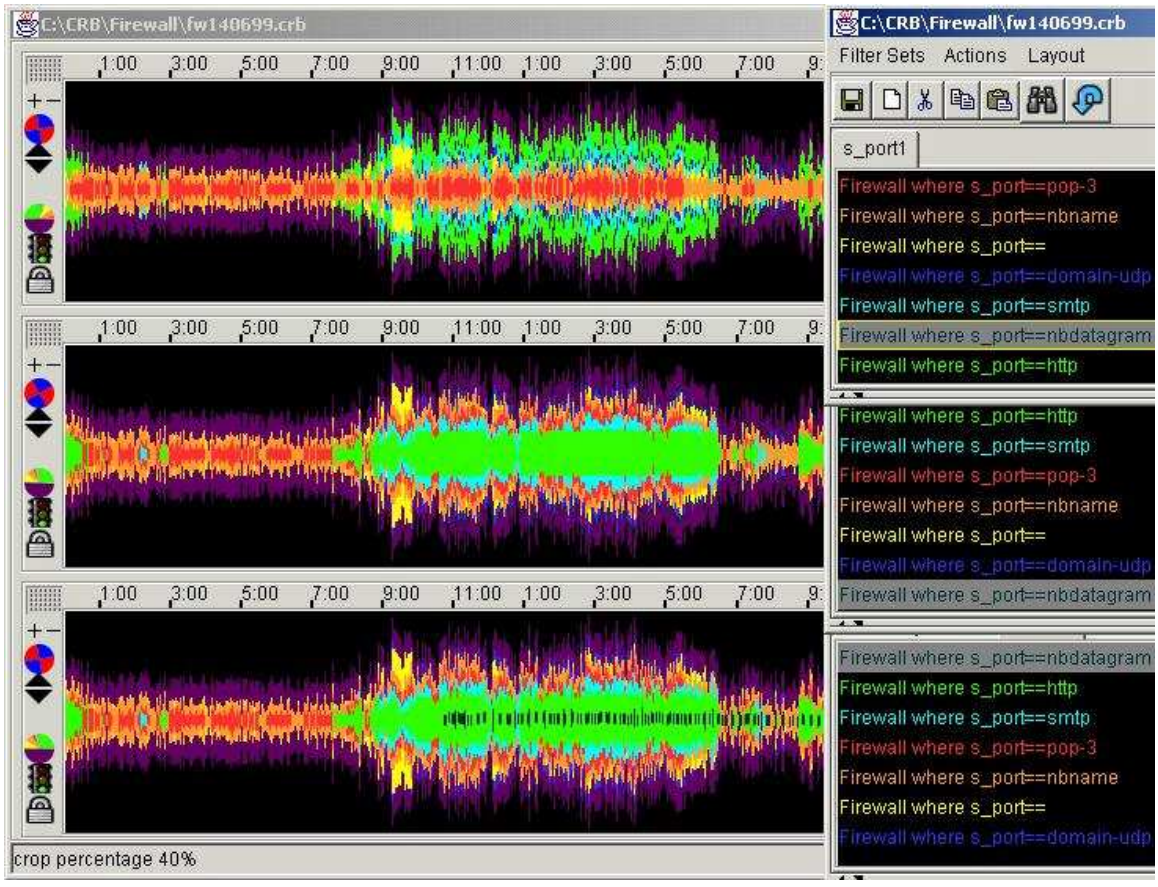

Figure 3. A symmetrical histogram

A visualization where the order of the rules really matters is shown in figure 3. It shows the same log 3 times, colored using the same rules but with the rules each time in a

slightly different order. This is a symmetric histogram with events matching the first rule displayed in the center and events matching the other rules around it in the order of the rules they match. Events not matching any rule are displayed on the outside. The height of the histogram is a logarithmic function of the number of events that have occurred in the time interval with the width of one pixel. The events displayed in the center appear much more prominent.

Each graph has a set of rules associated with it. Both the contents of a rule set and the association of a rule set with a graph can be interactively changed. Rule sets can be given names, making it easier to retrieve them later. In order to comfortably construct sensible rules, it is possible to scan the log for attribute values annotated with their frequency, after which one or more specific attribute value can be selected to construct a rule for events matching these values.

Although it can be useful to have some standard sets of rules to apply to logs of a certain type using this tool, we think it is vital that the user experiments and tries to construct some new rules in order to discover interesting unexpected relations in the data. Our prototype is not limited to a specific type of data, but, in general, you need to be a domain expert to correctly interpret these visualizations, to decide which patterns are interesting or which rules may result in interesting patterns.

The data we have been investigating includes authentication logs (wtmp/btmp, Radius, ACS), generic event logs (syslog, Windows NT event log), audit logs, firewall logs and network traces obtained by a sniffer. Although our prototype is in some regards still rather limited, experimental usage has shown that it is very powerful in several ways. Our initial results using it on client data have been very promising.

Conclusions

In practice, a log file policy may not be given as much thought as it deserves. One of the reasons log files often do not attract much attention is that powerful tools to disclose valuable information contained in them have been missing.

Information visualization provides us with means to discover connections between events that do not adhere to a known, well defined and, for our purposes, useful model. Log data as normally encountered in system administration tasks is an example of event data lacking such an underlying model.

We have presented a prototype tool which employs visualization techniques and enables us to interactively explore event data. Before we can analyze data with the tool, it has to be converted to an efficient storage format. For typical log data, this is a trivial procedure. The only (rather arbitrary) demand we currently require data to conform to in order to be amenable to inspection, is that each event should be accompanied by a timestamp. The current version of our tool insists on ordering the events in time, but this could easily be relaxed or at least the ordering could be done in another (user selectable) dimension.

The initial results we have achieved by experimentally employing our tool have been very promising. There have been many suggestions from our users to improve our

prototype by using novel techniques and by enhancing the user interface. We hope to develop it further into a fully-fledged, efficient and robust data analysis tool.

References

[1] Stephen G. Eick, Michael C. Nilson, Jeffery D. Schmid, *Graphical Analysis of computer logfiles*, CACM, Vol. 37, No. 12 (December 1994) 50-56.

[2] Stephen G. Eick, *A visualisation tool for Y2K*, IEEE Computer, Vol 31, No 10 (October 1998) 63-69.

[3] Dean F. Jerding, John T. Stasko, *The Information Mural: A Technique for Displaying and Navigating Large Information Spaces*, IEEE Transactions on Visualization and Computer Graphics, Vol. 4, No. 3, July-September 1998.

[4] Brian Johnson, Ben Schneiderman, *Treemaps: a space-filling approach to the visualization of hierarchical information structures*, Proc. of the 2nd International IEEE Visualization Conference, San Diego, October 1991, 184-291.

[5] Theo de Ridder, *Informatievisualisatie als beheerinstrument*, IT Beheer Jaarboek 1999, ten Hagen & Stam uitgevers, 221-227.