

# Interactive, Visual-Aided Tools to Analyze Malware Behavior

André Ricardo Abed Grégio<sup>1,2</sup>, Alexandre Or Cansian Baruque<sup>2</sup>, Vitor Monte Afonso<sup>2</sup>, Dario Simões Fernandes Filho<sup>2</sup>, Paulo Lício de Geus<sup>2</sup>, Mario Jino<sup>2</sup>, and Rafael Duarte Coelho dos Santos<sup>3</sup>

<sup>1</sup> Renato Archer IT Research Center (CTI/MCT), Campinas, SP, Brazil  
`argregio@cti.gov.br`

<sup>2</sup> University of Campinas (Unicamp), Campinas, SP, Brazil  
{vitor, dario, paulo}@las.ic.unicamp.br, jino@dca.fee.unicamp.br,  
orcansian@gmail.com

<sup>3</sup> Brazilian Institute for Space Research (INPE/MCT), S. J. dos Campos, SP, Brazil  
`rafael.santos@lac.inpe.br`

**Abstract.** Malicious software attacks can disrupt information systems, violating security principles of availability, confidentiality and integrity. Attackers use malware to gain control, steal data, keep access and cover traces left on the compromised systems. The dynamic analysis of malware is useful to obtain an execution trace that can be used to assess the extent of an attack, to do incident response and to point to adequate counter-measures. An analysis of the captured malware can provide analysts with information about its behavior, allowing them to review the malicious actions performed during its execution on the target. The behavioral data gathered during the analysis consists of filesystem and network activity traces; a security analyst would have a hard time sieving through a maze of textual event data in search of relevant information. We present a behavioral event visualization framework that allows for an easier realization of the malicious chain of events and for quickly spotting interesting actions performed during a security compromise. Also, we analyzed more than 400 malware samples from different families and showed that they can be classified based on their visual signature. Finally, we distribute one of our tools to be freely used by the community.

**Keywords:** Security data visualization, malware analysis

## 1 Introduction

Malicious software—malware—is the main current threat to information systems security. It is usually spread through the Internet and can cause incidents with severe damage to confidentiality, integrity or availability of systems and data. Most malware are targetless, attacking as many systems as they can, so that an attacker can gain control and use of the victim’s resources or steal sensitive data. However, there are cases in which malware have specific targets and are thoroughly designed to delude the victim, talking the unsuspecting user into

supplying confidential information, as it happens in attacks directed to a government infrastructure. In both situations, severe incidents caused by malware can disrupt an entire network of systems by disseminating through headquarters and then to branch offices or can also cause irreparable damage by exposing confidential information.

When attacks succeed in breaking into a computer system, forensic procedures can be followed in order to find out:

- How the attack was perpetrated;
- Where the points of collection of information are (downloading of tools and sending of data);
- What happened during the attack to the system.

In the case of malware attacks, it is important to collect the binary that infected the system or was downloaded after the target system was compromised. This binary may then provide some clues leading to a deeper understanding of the techniques used by the attacker and the purpose of the attack. This can be done by running this malware in a controlled environment and monitoring all the filesystem and network activities to compose a specific behavioral trace.

Malicious behavioral traces are in essence a log of the events performed by a malware on a compromised system, but this amounts to large chunks of data. Such logs are difficult to analyze as we have to stress out interesting segments of behavior (main malicious actions) while simultaneously having to obtain a general overview of the extension of the damage. However, the information obtained from this analysis is paramount to provide adequate incident response and mitigation procedures. In those cases of massive amounts of textual data to analyze, we can apply visualization techniques that can greatly enhance the analysis of logs and allow us to quickly spot important actions performed by a specific malware and to better understand the chain of malicious events that led to the compromise of the target system.

The main contributions of this article are:

- We developed two visualization tools—the behavioral spiral and the malicious timeline—to aid security analysts to observe the behavior that a malicious software presents during an attack. Those tools are interactive and they allow a user to walk through the behavior while performing zoom, rotate, and gathering of detailed information for each malware action.
- We discuss visual classification of malware families and show that our tool can be used to visually identify an unknown malware sample based on its comparison to previously known malware, and that is a step towards a visual dictionary of malicious code.
- We distribute online a beta version of our prototype, so that the community can benefit from it and use it freely.

## 2 Related Work

There are several research works that use visualization tools to overcome the plenty of data provided by textual logs related to security. Some of them are not

open to the public, others are neither intuitive to use nor interactive, and there are those whose results are more difficult to be visually interpreted by security analysts than if they search manually through the log files.

Quist and Liebrock [12] applied visualization techniques to understand the behavior of compiled executables. Their VERA (Visualization of Executables for Reversing and Analysis) framework helps the analysts to have a better understanding of the execution flow of the executable, making the reverse engineering process faster.

Conti et al. [3] developed a system that helps the context-independent analysis of binary and data files, providing a quick view of the big picture context and internal structure of files through visualization. In a forensic context it is especially helpful when analyzing files with undocumented formats and searching for hidden text messages in binary files.

Trinius et al. [15] used visualization to enhance the comprehension of malicious software behavior. They used treemaps and thread graphs to display the actions of the executable and to help the analyst identify and classify malicious behavior. While their thread graphs can confuse a human analyst with lots of overlapped information and lack of interactivity, our timeline (Section 4.1) allows a walk-through over the chain of events performed by different processes created and related to the execution of a certain malware sample, as well as the magnification of interesting regions, information gathering about selected actions and annotation. Furthermore, our behavioral spiral represents temporal action, whereas their proposed treemaps consist of the actions' distribution frequency. Again, there is a lack of interactivity and excessive data, as we handle only actions that can cause changes on the target system. However, similarly to our work, it is not possible to visually classify every malware family, as variant samples can pertain to a class while presenting completely different behavior regarding the order or nature of the performed actions.

Finally, reviewing logs from intrusion detection systems is an important task to identify network attacks and understand these attacks after they happened. There are several tools that use visualization for this purpose; each one has its own approach and is better than others at specific situations. To take advantage of those tools the DEViSE (Data Exchange for Visualizing Security Events) framework [13] provides the analysts a way to pass data through different tools, obtaining a better understanding of the data by aggregating more extracted information.

### 3 Data Gathering

To visualize malware behavioral data, we first need to collect malware samples that have been currently seen in the wild and then analyze them to extract the actions they would perform in an attack to a target system. In the sections below, we discuss our approaches to malware collection and behavior extraction.

### 3.1 Malware Collection

To collect malware samples that spread through the Internet, we use the architecture described in [5], which uses mixed honeypots technology (low and medium interaction) [10] to capture malicious binaries for MS Windows systems. Honeypots are systems that are deployed to be compromised so as to lure attackers to reveal their methods and tools by the compromise of a highly controlled environment. The collection architecture consists of a **Honeyd** [11] node to forward attacks against certain vulnerable ports to a **Dionaea** [7] system, which emulates vulnerable MS Windows services in those forwarded ports and actually downloads the malware sample. During 2010 we captured more than 400 unique samples, which are used as our test dataset in this article.

### 3.2 Behavior Extraction

To extract the behavior of a malicious software, we run it in a controlled environment and monitor the actions it performs during the execution in the target system. Those actions are based on the system calls executed that are relevant to security, i.e. if they modify the status of the system or access sensitive information, such as file writing, process creation, changes in registry values or keys, network connections, mutex creation and so on. The dynamic analysis environment used for behavior extraction is BehEMOT [6], a system that produces logs in which each line means one action performed by the monitored malware sample or a child process and is in the form “*timestamp, source, operation, type, target*”. For instance, let's suppose that a malware sample “*mw.exe*” created a process entitled “*downloader.exe*”, which connects to port 80 of an Internet IP address *X.Y.W.Z* to download a file *a.jpg* to a temporary location *TEMP*. The log file produced by BehEMOT would have the following three lines:

```
ts1, mw.exe, CREATE, PROCESS, downloader.exe
ts2, downloader.exe, CONNECT, NET, X.Y.Z.W:80
ts3, downloader.exe, WRITE, FILE, TEMP/a.jpg
```

Therefore, we use the BehEMOT behavior format to input the textual data to our visualization tools, which are described in the next section. It is worth noting that logs produced from malware sample execution can be thousands of lines long, motivating the use of visual tools to aid the process of human analysis.

## 4 Interactive Visualization Tools for Behavioral Analysis

The behavior of a malicious program can be interpreted as a chain of sequential actions performed on a system (as seen in the previous section), which involves operating system interactions and network connections. The analysis of those operations can provide the steps that were performed in an attack to understand the incident as a whole, as well as detailed information about what was changed

in a system, such as libraries that were overwritten, infected files, downloaded data and even evaded information.

Usually, antivirus developers analyze unknown malware samples to create signatures or heuristics for detection. This is an overwhelming process and involves plenty of manual work, as a human analyst has to search for pieces of data that characterize the sample as part of an already known malware class or create a new identifier to it. Actually, this process is worse due to the increasing amount of new malware made from ‘do-it-yourself’ kits and variants of older ones. Sometimes, a malware is assigned to a family (and detected by an antivirus engine) based on the value of a mutex it creates, or on a specific process that it launches with a particular name, or on the kind of information it sends to the network.

Our motivation to develop visual tools is to make it easy to process and pinpoint very specialized information and then to help a human analyst to focus in the interesting actions performed by a malware sample. This way, it is possible to quickly analyze new malware by visualizing their overall behavior and expanding only those actions that an experienced analyst considers suspicious or important. Also, public available dynamic analysis systems (e.g. [8], [14], [1]) provide textual reports full of information that would be easier to be interpreted if an analyst could visually manipulate it. To fill this gap, we developed two tools that transform a textual report from a dynamic analysis system into an interactive and visual behavior, which are explained below.

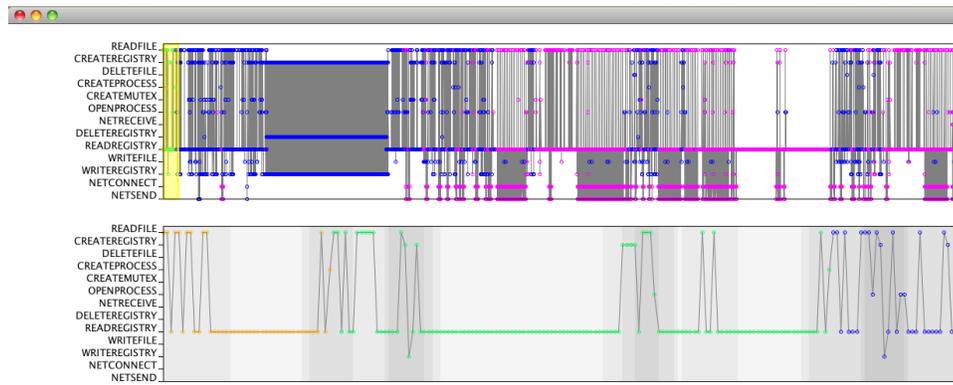
#### 4.1 Timeline and Magnifier

Malware time series events can also be visualized in simple x-y plots, where the x axis represents the time and the y axis some information about the event. The time information on the x axis can be any of the following: i) the absolute time of occurrence of an event; ii) the relative time of occurrence (counted from a particular initial value); or iii) a simple sequence that implies the order of occurrence of the events.

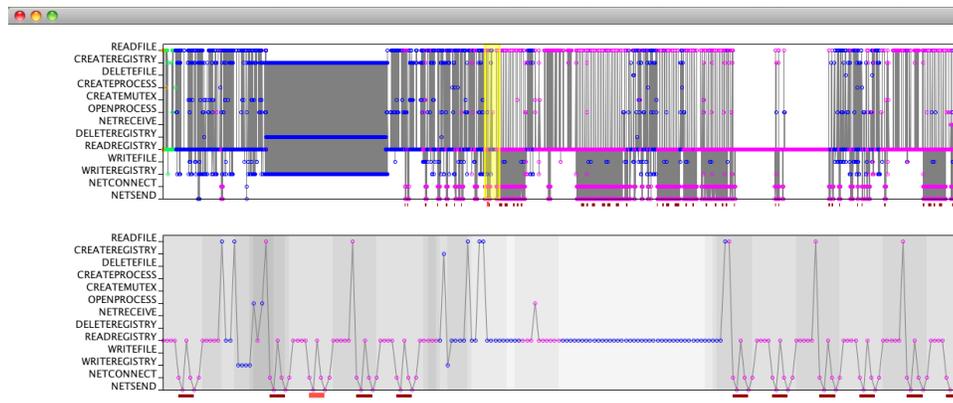
The event information can be plotted using several different methods, often specifically tailored to a particular purpose. The height on the y axis can be used to represent the frequency of occurrence, severity or intensity of a given event, if such information is known, or a discrete representation of event types. Decorations such as different marks for additional event characteristics can also be used to allow representation of more data dimensions than just two. Additional graphical elements may also be used, but one should always take care not to overload the plot with too much graphical content, which may confuse the user and hinder his/her ability to draw quick conclusions from the plot.

An example x-y plot used to represent malware time series events is given in Figures 1 and 2. They show the corresponding tool in action, as it parses a malicious behavior file with malware events ordered by action timestamp and plots the result using a simple, interactive interface. The tool draws the whole time series in two panels: on the top panel all points on the series are plotted, with the x axis representing the order in which events occurred and the y axis

representing the action associated with an event (which are not in any particular order and can be rearranged). Also, events are plotted as dots of different colors, according to the process id that performed such actions. For example, if the malware associated process created two child processes and also required service from an already running process, we would have four different colors in the graphic: malware, first child, second child and the running process). Not all possible monitored actions have to be performed by a malware, so the y axis varies accordingly to what was present in the captured behavioral trace.



**Fig. 1.** Timeline and Magnifier tool representing malicious events.



**Fig. 2.** Sequence of events selected by the user as a pattern to be searched (light red) and automatically matched events (dark red).

Plots created by this tool are also interactive. Since there are many events on the top section of the plot, it is hard to see exactly which one follows which one, so a region of interest (highlighted under a translucent yellow region on the plot) can be selected by the user. Selection is done by dragging the region with the mouse, which also causes the region of interest to be shown enlarged on the bottom panel of the plot. The x and y axes and plot colors follows the ones in the top section. The bottom part of the plot also conveys information about the diversity and variability of operation types in its gray background: darker backgrounds suggest a higher diversity, whereas lighter backgrounds point to higher similarity.

## 4.2 Malicious Spiral

The goal of this tool is to present an ordered sequence of all malicious actions of an attack in a spiral format, using an iconic representation. The spiral representation is useful to show the big picture of a malware sample behavior and also to allow quick visual comparisons among different malware samples even in the presence of variants. Instead of using a straight line broken in columns (as seen in [4]), the spiral format is less prone to confusion as small variances present in the behavior of malware from the same class usually keep the general visual appearance that models a family.

By exploiting the viewing abilities available, an investigator can zoom in and out, turn, tilt, select behavior slices, view the logged action in textual form and compare it with other behavioral data, if available. Thus, we provide not only an overview of the attack, but also the possibility of identifying certain behavioral patterns that could help in the classification of malware samples (Section 5).

The operations and types that are monitored and used to produce a behavior log are shown in Table 1, as well as all icons that are used to represent them. We divided the table lines in four groups of operations with similar purpose on each subsystem type, e.g., a CONNECT operation in a NET type has the same effect as that of a CREATE REGISTRY, i.e they prepare the environment for an active operation that will represent a registry value being written or a network connection being opened that later might send data. In the same way, a READ FILE or REGISTRY, a RECEIVE NET and a QUERY MUTEX are all passive operations, and so on.

## 5 Tests and Analysis of Results

Although we have developed the timeline and the spiral tools based on the same kind of log format, they have different usage. The timeline and magnifier tool can be used by any user to do a “behavioral walk-through”, while verifying how many processes were launched, what actions they performed and from what kind, etc. On the other hand, the spiral tool can be used to visually identify malware from the same family, while simultaneously depicting an iconic overview of the

**Table 1.** Monitored operations and types grouped by activity equivalence and differentiated by icons and colors

Action / Type	MUTEX	FILE	PROC	REG	NET
READ					
QUERY					
RECEIVE					
WRITE					
SEND					
CONNECT					
CREATE					
DISCONNECT					
DELETE					
TERMINATE					
RELEASE					

behavior that can be manipulated to show more detailed information (present in the log file).

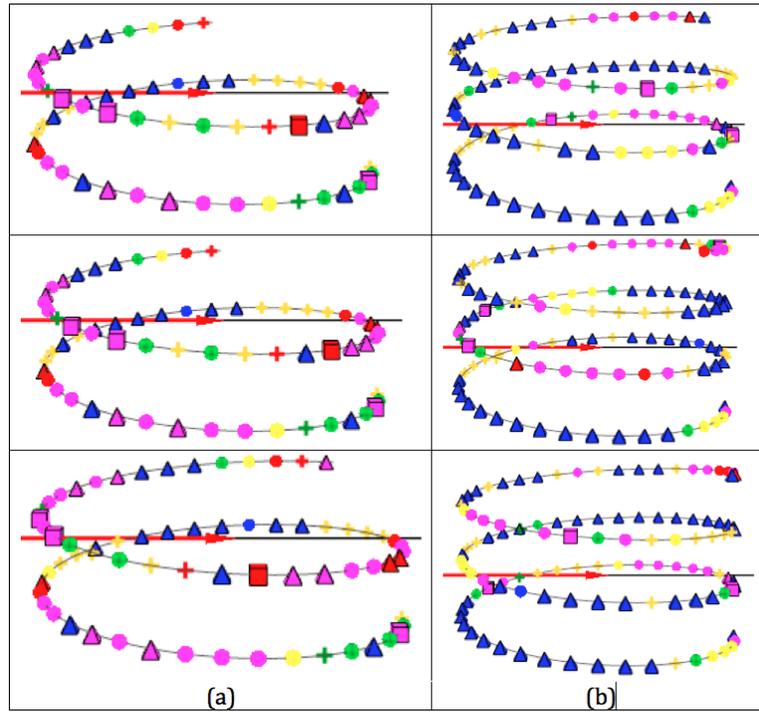
In this section, we show how the spiral tool serves as a visual dictionary and how it can help classify malware from the same family. The current prototype can be obtained from [9], together with larger screenshots. To the extent of our tests, we extracted the execution behavior from 425 malware samples collected by the system described in Section 3.1 and dynamically analyzed them with the system described in Section 3.2.

All malware samples from our dataset are currently found “in the wild” and together constitute variants from 31 families. Scanning them with the up to date ClamAV antivirus engine [2] reveals 94 unidentified samples. In [9] one can also verify the behavioral spiral pictures for each analyzed malware and its respective log.

By observing the generated spirals, we realized that it is possible to group some of them by their visual behavior. Also, malware samples from different families present similar behavioral patterns among samples from their own class, while at the same time keeping a dissimilarity from other classes’ samples. This differentiation factor present in the visual patterns is an important indicative that clustering algorithms, artificial intelligence and data mining techniques can be applied to our logs to classify malware based on behavioral similarities.

In Figure 3, we chose two trojan families—Pincav and Zbot—and selected three samples of each one to be printed side-by-side. Notice that even when the

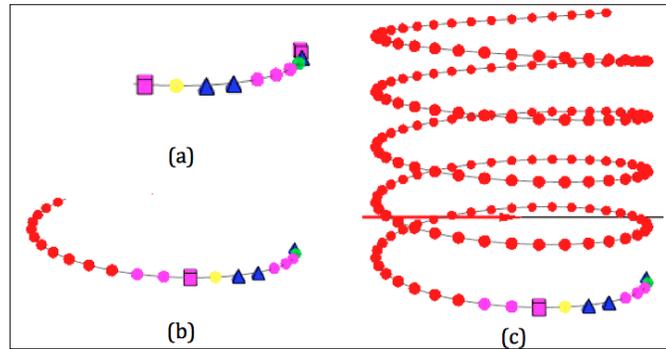
malware samples from a family perform variant operations, they still keep a behavioral pattern that can be used to characterize them in the same class.



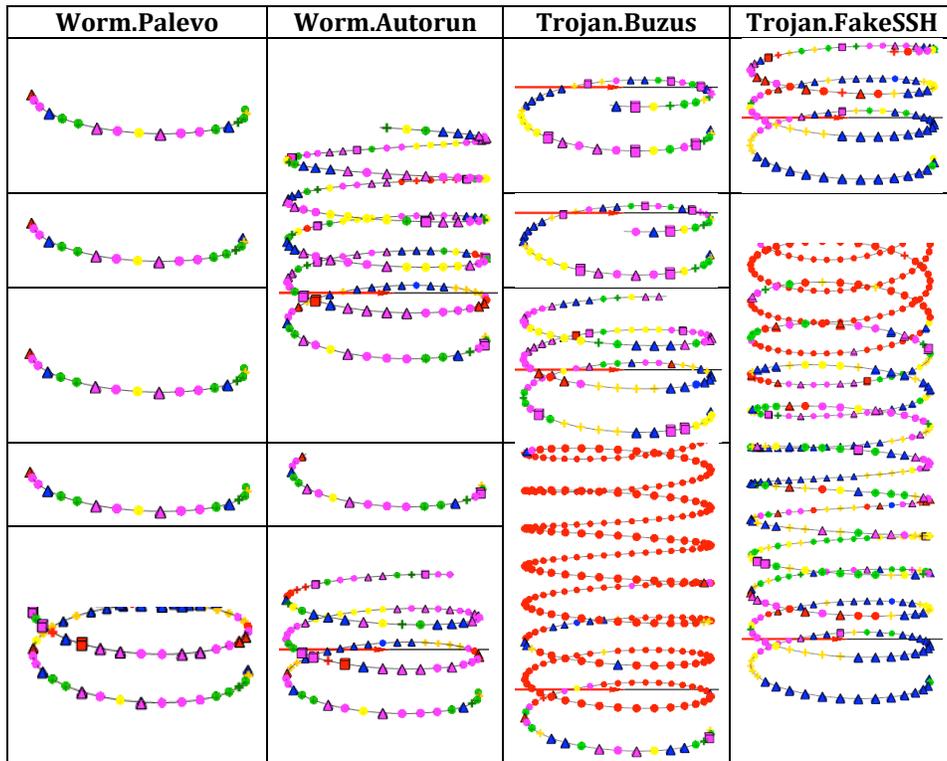
**Fig. 3.** Behavioral spirals from three samples of the trojan families ‘Pincav’ (a) and ‘Zbot’ (b).

Another interesting fact is that if a malware sample tried to do more network connections than another one from the same family (e.g., malicious scans) or if it performed fewer actions or crashed, it is possible to clearly notice the similarities between an incomplete behavior and a larger one, as shown in Figure 4, which contains three samples of the *Allapple* family.

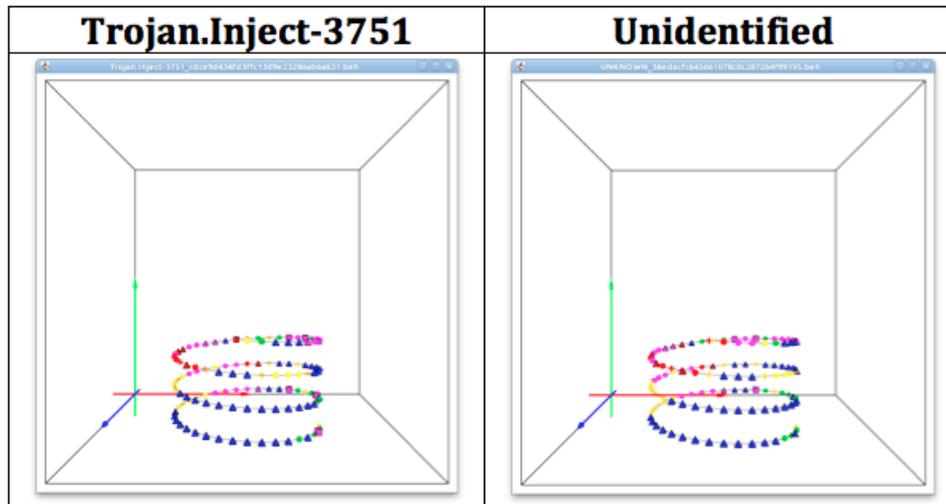
In Figure 5, we depict the behavioral spirals from four different malware families—the worms Palevo and Autorun; the trojans Buzus and FakeSSH. Notice that we can visualize the separability of the classes with minor variances among behaviors from the same family for Palevo, Autorun and FakeSSH. In the case of Buzus, one can observe that the first two behaviors significantly differ from the last ones, putting those samples apart from each other is an automated classification scheme. However, in Figure 6, one can also realize that a sample whose AV assigned label is “UNKNOWN”—i.e. an unidentified sample—presents a visual behavior that is quite similar to a sample from the trojan family “Inject”.



**Fig. 4.** Three ‘Allapple’ worm variants showing: (a) a sample that could not connect to the network and stopped its activities; (b) a sample that performed a short network scan; (c) a sample performing a massive scan, where each red sphere means a connection to a different IP address.



**Fig. 5.** Visual behavior extracted from four malware families.



**Fig. 6.** Unidentified malware (right) sample visually classified as a known threat—Inject trojan (left).

## 6 Conclusion and Future Work

In this article we propose two interactive, visual-aided tools to increase the efficiency in malware analysis, which allow an overview of malicious behaviors to security analysts. Moreover, our tools allow a walkthrough over the logs, the annotation and emphasis on interesting actions, the searching for patterns, a deep understanding of the damage performed on a target system and the visual comparison among malware samples. Hence, the possibility of visual family differentiation indicates that we can apply, in the future, an automated technique to classify, to cluster or to mine behavioral data. Also, it is possible to visualize which parts of a malware sample behavior are similar to another one's, indicating the same functionality or even code reuse. We are developing, as future works, a behavioral database that can bring some intelligence to the annotation process of the timeline/magnifier tool, the ability to load multiple logs at the same time to visualize several spirals in parallel and, finally, a classification algorithm to be integrated to the spiral tool that will allow us to automatically identify samples that share a high level of similarity and, after that, visualize their behavior.

## References

1. S. Buehlmann and C. Liebchen. Joebox: a secure sandbox application for windows to analyse the behaviour of malware. <http://www.joebox.org>.
2. Clam antivirus. <http://www.clamav.net>.
3. G. Conti, E. Dean, M. Sinda and B. Sangster. Visual Reverse Engineering of Binary and Data Files. *Proceedings of the 5th international workshop on Visualization for Computer Security (VizSec)*, 2008, pp. 1-17.

4. S. G. Eick, J. L. Steffen and E. E. Sumner, Jr. Seesoft—A Tool for Visualizing Line Oriented Software Statistics. In *IEEE Transactions on Software Engineering*, vol. 18, no. 11, pp. 957-968, 1992.
5. A. R. A. Grégio, I. L. Oliveira, R. D. C. dos Santos, A. M. Cansian and P. L. de Geus. Malware distributed collection and pre-classification system using honeypot technology. *Proceedings of SPIE*, vol. 7344, pp. 73440B-73440B-10, 2009.
6. A. R. A. Grégio, D. S. Fernandes Filho, V. M. Afonso, R. D. C. dos Santos, M. Jino and P. L. de Geus. Behavioral analysis of malicious code through network traffic and system call monitoring. *Proceedings of SPIE*, vol. 8059, pp. 80590O-80590O-10, 2011.
7. The HoneyNet Project. Dionaea. <http://dionaea.carnivore.it>.
8. C. Kruegel, E. Kirda and U. Bayer. Ttanalyze: A tool for analyzing malware. In *Proceedings of the 15th European Institute for Computer Antivirus Research (EICAR 2006) Annual Conference*, 2006.
9. MBS Tool. Malicious Behavior's Spiral - Beta version. <http://www.las.ic.unicamp.br/~gregio/mbs>
10. N. Provos and T. Holz. Virtual Honeypots: from botnet tracking to intrusion detection. *Addison-Wesley Professional*, 2007.
11. N. Provos. Honeyd - A Virtual Honeypot Daemon. In *10th DFNCERT Workshop*, 2003.
12. D. Quist and L. Liebrock. Visualizing Compiled Executables for Malware Analysis. *Proceedings of the Workshop on Visualization for Cyber Security*, 2009, pp. 27-32.
13. H. Read, K. Xynos and A. Blyth. Presenting DEViSE: Data Exchange for Visualizing Security Events. *IEEE Computer Graphics and Applications*, vol. 29, pp. 6-11, 2009.
14. ThreatExpert. <http://www.threatexpert.com>.
15. P. Trinius, T. Holz, J. Gobel and F. C. Freiling. Visual analysis of malware behavior using treemaps and thread graphs. *International Workshop on Visualization for Cyber Security (VizSec)*, 2009, pp. 33-38.