Eventpad

Citation for published version (APA):

Cappers, B. C. M., Meessen, P. N., Etalle, S., & Van Wijk, J. J. (2019). Eventpad: Rapid malware analysis and reverse engineering using visual analytics. In S. Trent, J. Kohlhammer, G. Sauer, R. Gove, D. Best, C. L. Paul, N. Prigent, & D. Staheli (Eds.), *2018 IEEE Symposium on Visualization for Cyber Security, VizSec 2018* [8709230] Institute of Electrical and Electronics Engineers. <https://doi.org/10.1109/VIZSEC.2018.8709230>

DOI:

[10.1109/VIZSEC.2018.8709230](https://doi.org/10.1109/VIZSEC.2018.8709230)

Document status and date:

Published: 09/05/2019

Document Version:

Accepted manuscript including changes made at the peer-review stage

Please check the document version of this publication:

- A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
- The final author version and the galley proof are versions of the publication after peer review.
- The final published version features the final layout of the paper including the volume, issue and page numbers.

[Link to publication](#)

General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license above, please follow below link for the End User Agreement:

www.tue.nl/taverne

Take down policy

If you believe that this document breaches copyright please contact us at:

openaccess@tue.nl

providing details and we will investigate your claim.

Eventpad: Rapid Malware Analysis and Reverse Engineering using Visual Analytics

Bram C.M. Cappers*

Paulus N. Meessen†

Sandro Etalle‡

Jarke J. van Wijk§

Figure 1: Visualizing A) Jigsaw and B) CryptX ransomware activity. C) Partitioning the data by file nesting shows high repetition of file creation and deletion patterns in the mini-map. D) Inspection of protocol data in a tabular view shows the creation of duplicate files with a .fun extension. E) Repetitive “Open-Find-Close” patterns as a result of recursive directory scanning.

ABSTRACT

Forensic analysis of malware activity in network environments is a necessary yet very costly and time consuming part of incident response. Vast amounts of data need to be screened, in a very labor-intensive process, looking for signs indicating how the malware at hand behaves inside e.g., a corporate network. We believe that data reduction and visualization techniques can assist security analysts in studying behavioral patterns in network traffic samples (e.g., PCAP). We argue that the discovery of patterns in this traffic can help us to quickly understand how intrusive behavior such as malware activity unfolds and distinguishes itself from the rest of the traffic.

In this paper we present a case study of the visual analytics tool EventPad and illustrate how it is used to gain quick insights in the analysis of PCAP traffic using rules, aggregations, and selections. We show the effectiveness of the tool on real-world data sets involving office traffic and ransomware activity.

Index Terms: Human-centered computing—Visualization—Visualization application domains—Visual analytics; Security—Software and application security—Software reverse engineering

1 INTRODUCTION

The analysis and identification of malware in computer environments is a complex and time-consuming task due to the size and variety of generated network traffic. Even if security analysts are aware of the presence of undesired activity (e.g., ransomware encryption patterns) it is still difficult to efficiently locate this behavior in large amounts of traffic using tools such as Wireshark [11] without a starting point [15, 19]. Automated analysis techniques already greatly assist analysts in

finding points of interest. In practice, however, they are often time-consuming to setup or very difficult to tune properly (e.g., managing false positives rates). We need alternative techniques to quickly explore patterns in malicious network traffic.

In this paper we present the application of a novel visualization technique Eventpad [10] to apply dynamic behavioral analysis to ransomware execution sequences. Instead of automatically trying to discover patterns of interest, we show how we can quickly gain insights in (un)desired traffic patterns by visually encoding traffic based on environmental knowledge. In this study we show how the analysis technique works, the kind of information it can reveal, and how it enables analysts to quickly study file access behavior using rules, aggregations, and selections. Specifically, our main contributions are:

- a visual analytics approach to forensic analysis of malware traffic enabling users to visually inspect and detect (un)desired patterns of interest using multivariate regular expressions.
- a use case of the EventPad system to malware analysis demonstrating how
 - visualization of execution sequences can be used to gain insight in underlying ransomware mechanics;
 - rules, automated techniques, and user interaction enables users to quickly test, discover, and compare traffic execution sequences.
- the introduction of mini-map functionality and a temporal view to study larger event collections and the frequency of sequential patterns over time.

The paper is structured as follows. First, Section 2 presents related work. Next, we discuss how our visualization techniques are beneficial for the analysis or identification of malware traffic. In Sections 4 and 5 we provide example explorations on real-world ransomware and office traffic and discuss the limitations of the approach. Finally, conclusions and future work are presented in Section 6.

*e-mail: b.c.m.cappers@tue.nl (Eindhoven University of Technology)

†e-mail: pmeessen@cs.ru.nl (Radboud Universiteit Nijmegen)

‡e-mail: s.etalles@tue.nl (Eindhoven University of Technology)

§e-mail: j.j.v.wijk@tue.nl (Eindhoven University of Technology)

Figure 2: Network packets are dissected using tshark. Parsed traffic is grouped by an attribute of choice and represented as series of blocks.

2 RELATED WORK

Malware visualization is an extensively studied topic covering a wide variation of techniques in different domains. Eventpad focuses on dynamic malware analysis using Deep Packet Inspection [27]. Analysis techniques that take the source code of the malware into account (i.e., static and hybrid malware analysis) are considered out of scope. For more information about these techniques, we refer to the handbook of Sikorski et al. [31].

The visualization literature with respect to *dynamic* malware detection can be grouped in two main categories, namely malware *discovery* versus *identification*. We start with an overview of common malware visualization techniques, discuss current limitations, and how we address these in Eventpad. Finally, we describe different visualization techniques that have been applied in the areas of Deep Packet Inspection and ransomware visualization.

2.1 Malware discovery

Malware discovery is the task of extracting samples from (network) environments that show signs of undesired/intrusive behavior. Shrivari et al. [30] made an extensive overview of different security visualization techniques that have been used for:

- *detection*: visually spot anomalies using node-link diagrams, parallel coordinate plots and pixel visualizations [14];
- *correlation*: study patterns in IDS alerts to enhance decision making [19]; and
- *assessment*: apply *root-cause* analysis through semantic zooming [12] and user-interaction [37].

Zhang et al. extended this survey [38] by studying how current visual encodings are effective for detection, correlation, and assessment tasks [23]. We believe however that systems for network monitoring and digital forensics should not be limited to a single task or visual encoding, as the set of user tasks can change during analysis [17]. In Eventpad we aim for an alternative approach where analysts can interactively define themselves which data attributes should be represented in what way to serve their task at best.

With ILAB, Beaugnon et al. [5] already illustrated the value of human interaction to incrementally label data instances for supervised intrusion detection models. Systems such as KAMAS [29, 36] also use rules to search for patterns in call sequences, but limit their sequential analysis to only this attribute. In Eventpad we show how incremental labeling combined with unsupervised clustering and alignment techniques are effective for the discovery of patterns in multivariate network traffic.

2.2 Malware Identification

Gaining insight in malware mechanics is crucial for understanding:

- what (kind of) systems are affected by the malware;
- the type of services or data the malware is interested in; and

- what countermeasures can be used to prevent this type of software from being executed in the future.

Eventpad assists analysts in the identification of (malicious) execution sequences by enabling users to define and search for patterns of interest using rules. Visual comparison of the packet details between the found sequences enable analysts to see whether the sequences show overlap and can be related to existing malware families or applications.

Wagner et al. [35] provided an extensive analysis of visualization techniques to visually compare malware samples, study samples individually, and summarize collections of malware together. Popular systems in these categories are CantorDust [13], Nataraj et al. [24], and Anders et al. [4] respectively. In their taxonomy Wagner et al. identified three main limitations of current systems and challenges for the future, namely

- Incorporating expert knowledge through interaction;
- Intertwining analytical methods with visualization; and
- Bridging gaps between forensics and classification through *rule generation*.

In Eventpad system we tackle these limitations by enabling analysts to define new patterns of interest using conditional formatting and rewrite rules. Tight coupling between automated methods and user input is achieved by interactively visual encoding packet sequences as *blocks*. The third limitation refers to the inability of malware analysis systems to test the effectiveness of new discovered rules and signatures for the analysis and classification of other malware samples. In Eventpad rules can be added and removed dynamically throughout data exploration and exported for future analysis tasks.

2.3 Deep Packet Inspection

Eventpad uses Deep Packet Inspection (DPI) to study user behavior in network traffic. Various systems have been proposed to visualize DPI data for the detection of Advanced Persistent Threats [32]. Systems like Wireshark [11], SNAPS [8], and CoNTA [9] already support the analysis of traffic at application level, but the provided search mechanisms and visualizations do not support the comparison and analysis of sequential patterns in network traffic. However, Camiña et al. [7] showed that for the detection of for instance masquerade attacks, the analysis of sequential patterns is crucial. We show how we can extend Wireshark to enable sequential analysis using Eventpad. With respect to ransomware analysis, Reuille et al. [28] analyzed the spreading of the cryptolocker virus by visualizing OpenDNS traffic. Krzysztof et al. [18] also illustrated the value of visualization to gain faster insights in larger collections of traffic.

In summary, current malware visualization techniques use static visual encodings to either focus on the detection of specific malware behavior or the identification of it. Systems that provide interaction and analytics methods provide little support for incorporating expert knowledge in the system or feeding new discovered patterns in these analytical methods. Eventpad enables interactive visual encoding

through rule generation to dynamically detect signs of new malware and to verify the presence of existing ones.

Figure 3: A) Users can visually encode sequences according to attributes of interest by constructing one or more rules. B) Rules are constructed using multivariate regular expressions. Users can design their own blocks to highlight points of interest. C) Event collection after rule rewriting.

3 EVENTPAD

Eventpad is a dynamic behavioral analysis tool designed to study sequential patterns in network traffic. Without loss of generality we assume that a network packet (or *event*) has a timestamp and belongs to a particular *sequence* (also known as a conversation, session, etc.).

In Eventpad we visualize network traffic as a list of block sequences where sequences correspond to *network conversations* and packets correspond to *blocks* (Figure 2). Initially, every packet is represented by a gray block. Users can create *rules* to replace block sequences by new ones. Rules can be used for three purposes:

- *discover* patterns by visually encoding packets according to attributes of interest;
- *test* the presence of (un)desired behavior through pattern matching; and
- *guide* automated techniques in discovering patterns by defining new (higher level) concepts.

Similar to regular expressions, operators such as sequential composition, iteration (0 or more times), and choice are provided to construct more complex replacements. Figure 3 shows an example of the rule construction interface. Double clicking a block in the interface results in a popup where users can add Boolean constraints to the chosen block (Figure 3B, similar to Wireshark). Only blocks whose attribute and values match the Boolean constraint are replaced by the rule's right hand side. For a more formal model of the query mechanism we refer to the work of Cappers et al. [10]. Multiple rules can be applied after one another to enable incremental rewriting of the traffic. This is also illustrated in Figure 4B.

The rules enable users to highlight and visual encode traffic properties that are of interest. Automated techniques such as clustering and alignment in turn can use this labeling to discover patterns between packet sequences. Clustering enables users to study pattern frequencies, whereas alignment can detect overlap between similar sequences using Multiple Sequence Alignment [6]. Figure 5 shows a schematic overview of all operations that can be applied in the Eventpad system. Eventpad consists of four main components.

Figure 4: A) Block expression operators are similar to regular expressions. B) Example where multiple rules apply to the same block.

Figure 5: Workflow scheme of Eventpad. Simplify event sequences by incrementally replacing block sequences using rules. Compare execution sequences by applying clustering and alignment techniques to the rewritten data. Inspection of attributes in overlapping sequences enable analysts to identify (un)desired behavior in the traffic. New insights can directly be incorporated in the analysis by defining new rules throughout exploration.

- The Sequence view (Figure 6A) visualizes conversations as sequences of blocks. The minimap on the right (Figure 6B) enables discovery of patterns over larger data collections;
- The Rule view (Figure 6F) shows the ordering in which rules are applied to the data. The icicle plot above (Figure 6E) shows how much of the data is affected by every rule;
- The Attribute view (Figure 6C) shows an overview of all event properties as histogram widgets. The histograms are highlighted whenever one or more blocks are selected. The histogram widgets enable users to inspect overlap in properties between two or more blocks;
- The Linechart (Figure 6G) plots event frequencies over time to detect temporal patterns such as bursts, drops, and periodicity. The colors in the chart show when and how often rules have been fired over time.

4 USE CASES

To illustrate the effectiveness of the tool for digital forensics, we applied Eventpad to the analysis of file access behavior in a real-world office network. For a better overview of the system's usage in practice, we refer to the supplementary material¹. The system² and recorded ransomware samples³ are also available for download.

4.1 Problem statement

Over the last few years ransomware attacks have become an increasing threat to society, with the primary aim to earn money by captivating resources. The largest class of these viruses use file encryption to achieve their goal [26]. Recent occurrences of WannaCry and Petya [21] affected over 300.000 computers world-wide causing millions of financial damage by encrypting network shares. The faster we can locate their underlying mechanics, the better.

Traditional behavioral analysis tools discover ransomware by analyzing entropy changes in network traffic [33]. These techniques however suffer from high false positive rates, since entropy alone

¹<https://youtu.be/g4brXOtPELI>

²<http://www.event-pad.com>

³https://security1.win.tue.nl/doku.php?id=artefacts#data_sharing

Figure 6: Graphical user interface of the implemented prototype and components: A) Sequence view represents sequences shows network packet conversations as series of blocks. B) The minimap shows patterns over larger amounts of data. C) The attribute view shows trends and patterns in protocol fields. D) Selections are stored in filters throughout exploration. E) The icicle plot shows the coverage of the applied rules. F) The Rule view shows a list of applied rules during exploration. The ordering for rewriting is controlled via drag and drop operations. G) A linechart can be enabled to study the activation of rules over time.

Figure 7: A) Highlight rules that are initially applied for the analysis of office traffic. B) Grouping the traffic by a) smb2.sesid, b) file nesting, and c) smb2.fid shows how different patterns can be discovered.

cannot distinguish between legitimate versus ransomware file encryption. Together with Mlder et al. [22] we have extended entropy-based detection in Samba shares by also incorporating file mutation patterns (e.g., read, write, open, close, delete etc.) in the analysis. A significant challenge however was to discover these patterns in network samples using tools such as Wireshark. This involved manually tracking file identifiers and stepping through captured network traffic for every connection. In this use case we demonstrate how we can use Eventpad to quickly reverse engineer file access activity using visual analytics.

4.2 Experimental setup

To study ransomware activity, we setup a Honeypot and built a detector that can protect Samba shares by passively observing the network traffic [22]. This lab consisted of four victim virtual machines, each with a different version of the Microsoft Windows operation system, which were connected to a Samba share on a virtual machine running Windows Server 2012. The Samba share contained a large collection of files, loosely based on the office type files found on the network shares at the university and the source code repositories of a number of software projects. All these machines were in a virtual network connected to the Internet, since some malware will not run if it cannot connect to specific command and control servers [31].

All network traffic in this virtual network was captured using tshark. Before and after the experiment all virtual machines were reset to a verified snapshot. The malware in the experiment was

Figure 8: A) Grouping by smb2.sesid shows repetitive behavior. B) Grouping the data based on file nesting shows repeated access of files depending on their height in the file directory structure C) Grouping by smb2.fid shows the file operations applied per file.

captured by the university, found on TheZoo [25], or found in spam e-mail. Every sample is executed on the victim machines to ensure encryption of the share. Services were stopped once the encryption phase of the samples was over.

Using our setup we captured activity traces from two common ransomware families: CryptX and Jigsaw. We also captured some traffic from the university network shares to test if these ransomware samples were present there.

Figure 9: A) Repetitive patterns in the Jigsaw traffic. B) Clustered open close patterns show high similarity. C) Inspection of read and writes shows that data is copied to files with .fun extension.

Figure 10: Discovering patterns in Jigsaw traffic by capturing repeating access patterns in rules. A) Reduction from 1429 to 49 unique patterns. B) Rule set to compress repeating find, open, and close behavior. C) Resulting traffic after rewriting.

4.3 Partition Strategies

Although the sequence of opening, reading, and closing a file in general may not be suspicious, they can be considered malicious when executed by a particular user and/or moment in time etc. The way we group packets into sequences therefore determines the type of patterns that stand out. This is also referred to as *context* [9].

Whenever a file or directory is opened, a Samba request is sent. Directories are closed when all files in that directory are closed. This results in a nesting pattern of Open and Close requests that shows the order in which files are accessed in the traffic (illustrated in Figure 7B). Naively grouping traffic by Samba's session id (Figure 7B-a) resulted into one large sequence as we discovered that the ransomware samples typically executed all behavior in one session. Although we are still able to discover patterns, this can hinder inter-sequence analysis with clustering and alignment techniques. Grouping the traffic by file access on the other hand does not preserve the order in which certain file operation are performed (Figure 7B-c). This information however is crucial for understanding how systems deal with files and data transfer.

To ensure that this ordering is preserved, we implemented a stack-based session based on file nesting. Each time a file is opened, the corresponding filename is pushed to a stack. The name is popped from the stack upon encountering its corresponding close request. Figure 8 shows the effect of traffic clustering when ransomware traffic is grouped by session, file nesting, or file id. The frequency of every pattern is shown at the end of every pattern.

4.4 Ransomware

To study the creation, deletion and file modification in the ransomware samples, we initially created rules to highlight all `smb2.cmd` open, close, read, write in yellow, pink, blue and purple blocks respectively (Figure 7A). In case of Samba files a `smb2.flags.disposition` is set true whenever a file is ready to be deleted. These blocks are marked in red. Figure 1A and 1B show some of the patterns we discovered in Jigsaw and CryptX traffic.

4.4.1 Jigsaw

We initially start exploration of approximately 1,500,000 packets (Figure 6D) by analyzing the traffic by `smb2.sesid`. This resulted into one large sequence per session. Inspection of the `smb2.create.action` attribute (Attribute view, Figure 6B) shows that there are two ways in which files are opened, namely

- `FILE_OPEN` If the file already exists, return success; otherwise, generate an error.
- `FILE_OVERWRITE_IF`: Overwrite the file if it already exists; otherwise, create the file.

Figure 11: A) The icicle plot shows that some CryptX packets are both Open and Find requests. B) Inspection with Wireshark. The minimap (C) and line chart (D) show that compound requests are sent before directory traversal and outside the encryption phase. E) Grouping the traffic by `smb2.filename` shows that files are overwritten. F) During encryption files are opened and closed sequentially.

These are marked by a yellow “Overwrite IF” and “O” blocks respectively. The `smb2.nt_status` widget in Figure 6C shows that the Jigsaw virus did not produce any errors when sending the requests.

After applying the highlight rules in Figure 7A, the minimap shows repetitive patterns in the traffic (Figure 1A). Closer inspection of these patterns shows that at some point files are repeatedly deleted (Figure 9A, indicated by the “X” block). Inspection of the line diagram shows that the pattern repetition started around 23:15 (Figure 9A).

To study the frequency of the discovered pattern we constructed a rule that “cuts” the data upon encountering file deletion blocks (scissors, Figure 9). This resulted into 1,195,817 sequences as illustrated in Figure 9D. Clustering the sequences however shows a lot of similarity between the different sequences (Figure 9B).

Figure 12: Access patterns to Microsoft Policy files (C) in (A) university traffic versus (B) access patterns of a user backup (D). Note that files are accessed in parallel (not sequentially). E) Policy files are accessed on multiple periods in time. F) Backup burst happened on January 22nd.

Hovering the mouse over the “read” block of the clustered sequence shows a list of all files that have been accessed in the exact same way (Figure 9C). Grouping and clustering the data with our stack-based session approach reveals three main patterns in the traffic (Figure 1C):

- The creation of a new encrypted file X.fun,
- The deletion of the original file X, and
- Directory traversal using repeated Open Find Close patterns

This enables us to see that the Jigsaw virus first creates a copy of the target file before its starts encryption. After the encryption, the original file is deleted instead of overwritten. The attack is flawed in the sense that the original file could still be recovered from disk using Disk Recovery tools [16].

In order to reverse engineer the mechanics of the virus, we construct rules to capture repeating block patterns to higher level concepts. Figure 10B shows the rules that are applied to simplify the traffic. The first two rules compress repetitive occurrences of open and close patterns. The frequency of these patterns depends on the file depth in the directory structure. Since the number of sequential read and write patterns depends on the size of the requested file, we also create a rule that compressed these sequences in a single block. After compression, the number in the upper right corner of the blocks shows how many blocks are contained in the new block. Applying alignment on the remaining patterns revealed that overlap between these file access behavior is large (Figure 10C).

4.4.2 CryptX

For the analysis of CryptX we start with the rule set as constructed for the Jigsaw virus (Figure 7A). Inspection of the line diagram shows that read/write bursts started between 20:19 and 20:35. In contrast to Jigsaw we do not see any delete requests.

After applying the rules, the icicle plot shows that for some Samba packets multiple rules apply at the same time (Figure 11A). Closer inspection in Wireshark showed that these packets consisted of multiple Samba headers (Figure 11B). Figure 11C shows examples of *compound* requests that represent the opening of a directory and finding files in that directory encoded in one packet.

Compound Samba requests in general are valid with respect to the protocol specification as they have been introduced for efficiency reasons. There are however known bugs in practice with these constructs [1–3]. In addition, Samba intrusion detectors without DPI may actually be unaware of multiple headers in Samba traffic.

Figure 13: Studying sequential Cryptx file access patterns (A). B) Example of a recursive directory scan before encryption C) Wireshark detailed view. D) Detailed overview patterns E) Tabular view shows files are accessed alphabetically.

The evaluation of viruses that use compound requests to masquerade file reads and writes in practice is left for future work. After searching for compound requests in the data, the line chart shows that these commands were sent before and after the burst period (Figure 11D).

To study the overall duration of sequences, users can enable an arc diagram, drawing an arc from the start to the end of every sequence (Figure 11D). The large arcs correspond to the opening and closure of directories whereas the small arcs correspond to file access.

Grouping the traffic by `smb2.file_name` shows that repeated read and write activity only happened during the burst period (Figure 11E). Zooming in on the linechart shows smaller arcs indicating that files were opened and closed in short periods in time (Figure 11F). Inspection of the burst period in the minimap shows that the burst consists of a large number of find requests followed by repetitive file read and write patterns. The find requests in Wireshark show that the virus, in contrast to Jigsaw, first recursively searches directories for files before the encryption burst starts (Figure 13A). Inspection of the burst shows that files are encrypted sequentially (Figure 13D) and are traversed alphabetically (Figure 13E).

Figure 14: Searching for Cryptx virus in University traffic. A) Search query for compound requests. B) Discovery of compound requests. C) Requests are originating from one source. D) All blocks access Microsoft Network Policy files.

Figure 15: Extraction of policy files. A) Overview policy access patterns B) Searching for file modification requests and grouping the traffic by ip.src. Discovery of unauthorized IP addresses modifying policy files. C) File modification happened throughout January. D) All blocks access Microsoft Network Policy files.

4.5 University Traffic

We studied the patterns in the ransomware samples and compared them to recorded traffic from a university that has been struck earlier by ransomware. The main question was whether there was still ransomware activity inside the network.

For the analysis of university traffic we recorded internal network traffic of 20 hosts over a period of a month. Out of this 94GB of traffic we extracted all smb2 meta-data using the tshark protocol dissector. This results in the analysis of approximately 14,000,000 packets with over 100 protocol fields. In order to reduce the large packet count, response values were merged in the corresponding requests based on their smb2.msg_id and session. In addition, file print services such as spoolss were excluded from the analysis.

We initially study the traffic by loading the rule set that we constructed with the ransomware analysis. The line chart in Figure 12 shows the presence of a traffic burst at January 22nd (Figure 12F). Grouping the traffic by smb2.sesid and inspecting the traffic shows that a user is reading files from his backup directory (Figure 12D). In contrast to ransomware traffic we see that Open and Close packets are not alternating, but occur in bursts showing that in the university traffic files are accessed in parallel rather than sequentially. Figure 12E also shows this for traffic outside the traffic burst.

To check for signs of the Cryptx virus we start searching for compound requests using Eventpad's find dialog (Figure 14A). This

revealed a burst of Open-Find compound requests similar to Cryptx (Figure 14B). The main difference was that files after the burst were only read. Inspecting the Attribute view shows that all compound requests were sent after the burst by the same IP address (Figure 14C). Inspection of the file names revealed that they were all related to the reading of Microsoft Network Policy files (Figure 14D).

To study the access patterns to these files in greater detail, we search for all packets that involve policy files. Figure 12A highlights the packets as a result of the search. Figure 12E shows that the pattern occurred several times in the data set. Filtering out the rest of the traffic, we can see that these policy files have similar access patterns (Figure 15A).

Although the reading of Microsoft Network Policy files is necessary to determine authorization, we know that the hosts involved in the recording are not authorized to change these policies. To verify this hypothesis, we search for conversations containing write requests (Figure 15A). Grouping the traffic by ip.src shows that several IP addresses that were violating this constraint (Figure 15B). Figure 15D shows that these users were modifying RemoteInstall, Logoff scripts and Registry files to gain access to the share from a virtual machine.

5 DISCUSSION

The use cases illustrate how we can use Eventpad to quickly gain insight in user activity by analyzing file access behavior in PCAP traffic. Users are enabled to simplify, focus on, and detect patterns inside network conversations by defining rules. Automated techniques in turn use the labeling to assist users in discovering patterns between different conversations. Inspection of packet attributes enable users to see whether these patterns share overlap in other attributes after which rule sets can be refined. There are several reasons for the rapid discovery of patterns in Eventpad:

- Constructed rules sets are efficient to evaluate as they are based on regular expressions.
- Users do not need to have a full specification of what "good or bad" behavior looks like, but can incrementally obtain results by visually encoding knowledge that they have about their data.
- Unsupervised automated methods in the background use the constructed rules sets to provide nontrivial insights.

Besides the creation of traditional black or white listing signatures, analysts create rules to assist them discovering unknown patterns

and focusing on parts that are relevant for their investigation. The constructed rule set can be evaluated fully automatically on new (incoming) data. By saving and loading rule sets analysts can quickly verify the presence of malware in other data sets.

Although the tool quickly enables analysts to isolate areas of interest in large network samples, there are threats to validity. First, we analysed ransomware activity by extracting file access behavior from network traffic. Ransomware viruses that do not focus on repeated (network) file encryption (e.g., UIWIX [20]) cannot be detected with this approach. Although Eventpad in general works on any event data, other data sources are required to make the technique useful for these types of malware.

Second, based on the Samba signaling alone we cannot distinguish between user initiated file encryption versus ransomware encryption. Although the patterns in the use cases showed clear differences between ransomware and university traffic (sequential versus parallel activity), this does not hold in general. Additional (automated) anomaly techniques are required to verify if the observed patterns indeed correspond to file encryption.

Third, the Eventpad system suffers from a “cold start” problem in the sense that users must already be aware of properties that are of interest. Naively applying clustering or alignment without a rule set results in no insights (underfitting) whereas creating too many rules can result into “noisy” patterns (overfitting). Also to deal with phenomena such as concept drift [34], rule sets need to be maintained by users to ensure the discovery of new viruses in the future.

6 CONCLUSIONS AND FUTURE WORK

In this paper we demonstrated a visualization tool to support rapid and cost-effective analysis of network traffic analysis and malware activity. We have shown the effectiveness of the system in real-world ransomware and office traffic. Our tool shows how visualization can be used to quickly gain insight in malware and network activity by combining data reduction and automated techniques in one interface using rules, aggregations, and selections.

ACKNOWLEDGMENTS

This work is funded by SpySpot, a project in the Cyber Security program of Netherlands Organisation for Scientific Research (NWO).

REFERENCES

- [1] Bugzilla “https://bugzilla.samba.org/show_bug.cgi?id=7331”.
- [2] Bugzilla “https://bugzilla.samba.org/show_bug.cgi?id=8407”.
- [3] Bugzilla “https://bugzilla.samba.org/show_bug.cgi?id=9173”.
- [4] B. Anderson, C. Storlie, and T. Lane. Improving Malware Classification: Bridging the Static/Dynamic gap. In *Proceedings of the ACM Workshop on Security and Artificial Intelligence*, pp. 3–14, 2012.
- [5] A. Beaugnon, P. Chifflier, and F. Bach. ILAB: An Interactive Labelling Strategy for Intrusion Detection. In *International Symposium on Research in Attacks, Intrusions, and Defenses*, pp. 120–140, 2017.
- [6] R. Bose and W. van der Aalst. Trace alignment in process mining: opportunities for process diagnostics. In *International Conference on Business Process Management*, pp. 227–242. Springer, 2010.
- [7] J. Camiña, J. Rodríguez, and R. Monroy. Towards a Masquerade Detection System based on Users Tasks. In *International Workshop on Recent Advances in Intrusion Detection*, pp. 447–465. Springer, 2014.
- [8] B. C. M. Cappers and J. J. van Wijk. SNAPS: Semantic Network traffic Analysis through Projection and Selection. In *Visualization for Cyber Security (VizSec), 2015 IEEE Symposium on*, pp. 1–8. IEEE, 2015.
- [9] B. C. M. Cappers and J. J. van Wijk. Understanding the Context of Network Traffic Alerts. In *Visualization for Cyber Security (VizSec), 2016 IEEE Symposium on*, pp. 1–8. IEEE, 2016.
- [10] B. C. M. Cappers and J. J. van Wijk. Exploring Multivariate Event Sequences using Rules, Aggregations, and Selections. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):532–541, 2018.
- [11] G. Combs et al. Wireshark-network protocol analyzer: <http://www.wireshark.org/>. Version 0.99, 5, 2008.
- [12] G. Conti, J. Grizzard, M. Ahamad, and H. Owen. Visual Exploration of Malicious Network Objects using Semantic Zoom, Interactive Encoding and Dynamic Queries. In *IEEE Workshop on Visualization for Computer Security*, pp. 83–90. IEEE, 2005.
- [13] C. Domas. Cantordust (derbycon 2012). “<https://sites.google.com/site/xxcantorxdustxx/>”, 2012.
- [14] J. Donahue, A. Paturi, and S. Mukkamala. Visualization Techniques for Efficient Malware Detection. In *IEEE International Conference on Intelligence and Security Informatics*, pp. 289–291, 2013.
- [15] S. Etalle. From Intrusion Detection to Software Design. In *European Symposium on Research in Computer Security*, pp. 1–10, 2017.
- [16] S. Garfinkel. Digital Forensics Research: The next 10 Years. *Digital Investigation*, 7:S64–S73, 2010.
- [17] L. Hao, C. G. Healey, and S. E. Hutchinson. Flexible web visualization for alert-based network security analytics. In *Proceedings of the Tenth Workshop on Visualization for Cyber Security*, pp. 33–40. ACM, 2013.
- [18] C. Krzysztof. Visualization as support for web honeypot data analysis. *Information Systems in Management*, 4(1):14–25, 2015.
- [19] Y. Livnat, J. Agutter, S. Moon, R. Erbacher, and S. Foresti. A Visualization Paradigm for Network Intrusion Detection. In *Proceedings from the IEEE Workshop on Information Assurance*, pp. 92–99, 2005.
- [20] T. Micro. Trend micro. “<https://blog.trendmicro.com/trendlabs-security-intelligence/wannacry-uiwix-ransomware-monero-mining-malware-follow-suit/>”.
- [21] Microsoft. Wannacry and petya: <https://www.microsoft.com/en-us/wdsi/threats/ransomware>. Microsoft, 0, 2017.
- [22] D. Mülders. Network based Ransomware Detection on the Samba Protocol, Mar 2017.
- [23] T. Munzner. A Nested Model for Visualization Design and Validation. *Transactions on Visualization and Computer Graphics*, 15(6), 2009.
- [24] L. Nataraj, S. Karthikeyan, G. Jacob, and B. Manjunath. Malware Images: Visualization and Automatic Classification. In *International Symposium on Visualization for Cyber Security*, p. 4, 2011.
- [25] Y. Nativ and S. Shalev. theZoo. <https://github.com/ytisf/theZoo>, 2016-2017.
- [26] S. Popoola, U. Iyekepolo, S. Ojewande, F. Sweetwilliams, and A. Atayero. Ransomware: Current Trend, Challenges, and Research Directions. In *Proceedings of the World Congress on Engineering and Computer Science*, vol. 1, pp. 169–174, 2017.
- [27] T. Porter. The Perils of Deep Packet Inspection. *Security Focus*, 2005.
- [28] T. Reuille, D. Mahjoub, and P. Yan. A Survey of Visualization Systems for Malware Analysis. In *Virus Bulletin OpenDNS*, 2014.
- [29] J. Schick, M. Wagner, N. Thür, C. Niederer, G. Rottermann, P. Tavalato, and W. Aigner. Rule Ceation in a Knowledge-assisted Visual Analytics Prototype for Malware Analysis.
- [30] H. Shiravi, A. Shiravi, and A. Ghorbani. A survey of visualization systems for network security. *IEEE Transactions on visualization and computer graphics*, 18(8):1313–1329, 2012.
- [31] M. Sikorski and A. Honig. *Practical Malware Analysis: the Hands-on Guide to Dissecting Malicious Software*. no starch press, 2012.
- [32] R. Sloan. Advanced Persistent Threat. *Engineering & Technology Reference*, 2014.
- [33] M. Stokkel. Ransomware detection with bro. Talk at Bro-Con ‘16, Austin, https://www.bro.org/brocon2016/slides/stokkel_ransomware.pdf, 2016.
- [34] A. Tsymbal. The Problem of Concept Drift: Definitions and Related Work. *Computer Science Dep., Trinity College Dublin*, 106(2), 2004.
- [35] M. Wagner, F. Fischer, R. Luh, A. Haberson, A. Rind, D. Keim, and W. Aigner. A Survey of Visualization Systems for Malware Analysis. In R. Borgo, F. Ganovelli, and I. Viola, eds., *Eurographics Conference on Visualization - STARs*. The Eurographics Association, 2015.
- [36] M. Wagner, A. Rind, N. Thür, and W. Aigner. A Knowledge-assisted Visual Malware Analysis System: Design, Validation, and Reflection of KAMAS. *Computers & Security*, 67:1–15, 2017.
- [37] J. S. Yi, Y. A. H. Kang, and J. Stasko. Toward a Deeper Understanding of the Role of Interaction in Information Visualization. *Transactions on Visualization and Computer Graphics*, 13(6):1224–1231, 2007.
- [38] T. Zhang, X. Wang, Z. Li, F. Guo, Y. Ma, and W. Chen. A Survey of Network Anomaly Visualization. *Science China Information Sciences*, 60(12):121101, 2017.