Weaving a Carpet from Log Entries: A Network Security Visualization Built with Co-Creation

Johannes Landstorfer* Ivo Herrmann* Jan-Erik Stange* Marian Dörk* Reto Wettach*

Department of Design at the University of Applied Sciences Potsdam, Germany

Abstract

We created a pixel map for multivariate data based on an analysis of the needs of network security engineers. Parameters of a log record are shown as pixels and these pixels are stacked to represent a record. This allows a broad view of a data set on one screen while staying very close to the raw data and to expose common and rare patterns of user behavior through the visualization itself (the "Carpet"). Visualizations that immediately point to areas of suspicious activity without requiring extensive filtering, help network engineers investigating unknown computer security incidents. Most of them, however, have limited knowledge of advanced visualization techniques, while many designers and data scientists are unfamiliar with computer security topics. To bridge this gap, we developed visualizations together with engineers, following a co-creative process. We will show how we explored the scope of the engineers' tasks and how we jointly developed ideas and designs. Our expert evaluation indicates that this visualization helps to scan large parts of log files quickly and to define areas of interest for closer inspection.

Keywords: Pixel-oriented techniques, task and requirements analysis, multidimensional data, network security and intrusion.

Index Terms: • Human-centered computing~Information visualization • Human-centered computing~Participatory design • Human-centered computing~Visualization design and evaluation methods

1 Intro

Network security engineers frequently inspect large log files, for example, to check automatically generated alerts, configure automatic systems, or investigate an unfamiliar incident that might be a new type of attack. Automated scanning and alerting systems perform the bulk of the routine detection work on log files. However, for issues that cannot be solved satisfactorily and to look for undetected malicious activity, engineers need (and want) to explore raw data themselves.

Data visualization can pose effective means to help with these tasks by supporting both automatic processing and human involvement, as a substantial body of research shows [6]. However, security engineers are often not familiar with advanced visualization techniques or the research has not yet found its way into widely available products [1], [24]. They mostly filter manually and with little cognitive support through vast amounts of data because they are well-versed in command line-based text mining tools (and because these tools are often lightweight yet robust). This is especially ineffective when less clear incidents need a wider scan and a more open form of exploration. Commercially available tools that feature rich graphical user interfaces (GUIs) started only recently to include more specific visualizations, beyond high-level line graphs, histograms, and pie charts [13].

Then again, for data scientists and visualization experts the field of network security is usually unfamiliar and hard to enter: log files are not self-explanatory but require substantial background knowledge, such as which server response code refers to which situation. Therefore, A deep immersion into the topic is required. Our intention with this research is to bring network security engineers and visualization designers together so that we can develop visual analytics tools that take the expertise of both groups into account.

The contributions that we want to present in this paper are twofold: to explore a structured way of collaboration between security and visualization experts and to find a customized form of visualization with this process. More concretely, we contribute

1 insights and requirements from security experts for early phase, explorative visualizations,

2 experiences and findings from applying a co-creative process to the development of data visualizations,

3 an adapted form of a pixel map that we call the "Pixel Carpet" which allows to display multivariate datasets (for low numbers of variables),

4 a visual highlighting mechanism based on value frequency and similarity.

According to Munzner's nested model for visualization evaluation [23], we contribute a domain problem characterization for network security visualization and a data/operation abstraction for visually analyzing network log files. Before we get to the visualization, we describe our co-creation process that we tried to apply across Munzner's model and our results. We discuss issues of both aspects towards the end of the paper.

2 A Co-Creative Approach to Data Visualization

In the following section, we briefly elaborate on specifics differentiating our approach from other work, then report about individual steps of our process, and finally describe the user goals and design requirements extracted together with the stakeholders. We provide general guidelines based on the reflection of our experiences in section 8.

2.1 Motivation

Innovation methods that involve the intended users early on are an established part of today's product development as well as visualization research [24], [29]. We wanted to extend the "classic set" of ethnographic methods and feedback sessions (as in e.g. [5], [21]) by elements from co-creation. The core idea of co-creation is to not only research about and design *for* a target group but to make them part of the team and design together *with* them [27]. Co-creation techniques typically focus on creating/making something together and aim at gaining insights by discussing the resulting artifacts. The intended benefits are a better understanding be-

*E-mail:
{landstorfer, ivo.herrmann, stange, doerk, wettach}@fh-potsdam.de

tween groups with different traditions and "languages", namely network security practitioners, security research, data visualization, and interface design. Furthermore, an even deeper involvement of all stakeholders, a broader spectrum of ideas, and a "built-in" validation: if the users take part in the design process, the designs are more likely to meet their needs.

2.2 Process and Methods

To build up domain understanding, extract insights, and jointly create new ideas we conducted interviews, made observations, and an ideation workshop. We describe further steps later, such as visualization design (section 4) and feedback sessions (section 5).

2.2.1 Interviews and Observations

Our core group consisted of 7 different people from 6, mostly scientific, institutions. Professional backgrounds were network security engineer, network administrator, security researcher, in operative as well as managing roles. We enlarged the team later, especially for the workshop session.

Our questions referred to the workplace and tool setup, important data types, typical tools and processes for incident management, and the role of visualizations in their work. Examples are: "What data do you usually work with during a usual day? What role do raw data (such as NetFlow) play in contrast to generated alerts?" "What kind of visualization are you familiar with in your professional environment? Where do you use it? Where do you prefer a raw/text view on your data?" We tried to avoid direct questions about which visualization they wanted, as the answer relies very much on the existing knowledge of the interviewees about visualizations. Instead, we deducted common tasks, implicit and explicit needs, and the role of visualization from their statements and our observations.

We could combine only half of the interviews with observations on location due to the limited availability of the busy experts and the sensitive nature of the domain. We were particularly interested in the physical setup of a workplace (such as number and arrangement of screens, personal and "public" screens), location of and relation to colleagues, communication devices, analog means such as white boards, and further details that the interviewees might take for granted.

We summarized the findings from our interviews and reflected the identified needs with the interviewees to avoid misinterpretations. We also included a reflection on requirements later on in our ideation workshop.

2.2.2 Ideation Workshop

As input for our one day workshop, we used needs and statements from the interviews, market research on security data visualization, and further material contributed by the participants. The goal was to come up with a visualization idea for an individually selected problem. The participants sketched out their ideas (Fig. 1), presented, commented, and refined, in single and team sessions. We also clustered the workshop results to identify common directions (see 2.3 for details). Creating ideas for selected challenges in groups helped to build up mutual understanding as a side effect.

Half of the day, a group of participants worked directly with their computers on data sets that we asked them to bring to the workshop (we called this session the "data picnic"). The idea was to create sketchy visualizations or at least an early analysis tool for one of the data sets. Besides, the participants would share tools and strategies which would facilitate later collaboration.

2.3 High-Level User Goals

In many larger computer networks, intrusion detection systems (short: IDS) scan for signatures of known attacks and issues. In various situations, the security experts told us that they were skeptical about automated detection systems. Some said they were too resource heavy and expensive, for some "they keep admins busy with false positives" (also in [1]). In general, they did not like that the internal mechanics of these systems are often hard to understand, thus alerting decisions difficult to reconstruct, resulting in a loss of control. "If it's not transparent, it might be better to stay with the raw data." While all had automated systems in place, they wished to combine them with tools dedicated to human pattern recognition and decision making. This discussion was particularly intense during the "data picnic" at the ideation workshop.

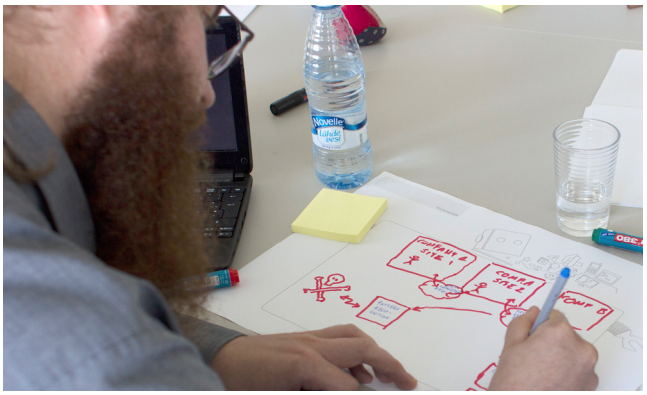

Fig. 1: A security researcher mapping his ideas for a cross-institutional monitoring system during the ideation workshop.

The security engineers might not know exactly what they are looking for when they start an inspection: "Analysis tools should encourage the use of gut feelings, e.g. through highlighting anomalies," as one interview partner put it. It is their experience and intuition during an explorative inspection that often guides them to suspicious incidents. Additionally, log files can come from any place and various devices in a network during an investigation. This causes a high variety of data structures that need to be investigated. "The complicated part is: how to choose the right filter settings? How to find the right characteristics? This needs some trial and error, an iterative approach."

Considering their workflow, the network engineers were more focused on ex-post incident analysis than on, e.g. live traffic monitoring. This relates to the wish for exploration which can hardly be done in real-time and aims more for detecting e.g., hidden root causes. Based on this use case, we can also assume that some "framing" information can be used to limit the amount of data (such as a time range, IP ranges, etc.). For the sake of completeness, we also want to mention that long term correlations or otherwise long ranging searches still require huge amounts of data to be parsed.

Discussions and output of the needs and requirements analysis touched many topics, from raw data information to e.g. team collaboration. We bundled goals to form different coherent threads of conversations and research efforts but the full scope is beyond the scope of this paper. In the following, we want to focus on a tool that supports user-driven, exploratory, post-incident analysis. It would complement the automatic systems that scan for the more well known attacks and do the bulk of routine checks.

2.4 Requirements for Our Visualization

Based on the high-level goals discussed before, we worked out key design requirements (DR). It is a synthesis from smaller ideas that the workshop participants came up with and refined.

DR1: Display raw data. Security experts had a strong preference for retaining easy access to unprocessed or hardly processed log data that could complement automatic warning tools. With the

help of a compact visualization, security experts want to inspect the data on their own in the hope to find patterns that the algorithms missed. They wished for raw data to be readily available as context information to assess automatically generated alerts.

DR2: Visualize extrema. Either the exception or the flood of data are suspicious. While many current tools highlight mass effects, the security engineers were also looking for solutions that support the investigation of targeted attacks with "low profile", i.e. few events. A consequence for visualizations is to highlight rare events over frequent ones.

DR3: Encourage explorative analysis. Stumbling across anomalies besides the known and obvious can be a strength of human perception [32]. The visualization must strive to show a large range of events that the human eyes can scan. It should present events by their parameters and avoid classifying – the goal is to leave decisions about classification to the human operator. The tools should further offer elements for flexible manipulation of views and user definable filters.

DR4: Combine overview and quick filtering. Visual tools should help to stay in context, correlate events, find patterns. Our participants wanted to have a fluid interaction when investigating an incident, for filtering out items they are interested in and switching back to an overview when needed. For a visualization, this means as much log records on one screen as possible to find patterns at large but also quick access to single records to inspect and filter them whenever necessary.

DR5: Support various log file types. The use case in focus can include a diverse set of log files of various data types and data structures. Well labeled and structured data sets allow for meaningful semantics in the visualization. As the security engineers frequently face unlabeled data sets, they recommended a straightforward tool instead that would not require them to structure the data before they could "see something".

3 Test datasets and Use Cases

When we went into more concrete design activities, we relied on our collaborating security experts to pick a relevant data set. They chose an ssh log file because it is frequently inspected after incidents and ssh servers are very important access points into networks [34]. Besides, it was good to start with because of its fewer fields and clearly defined use cases. As we discussed data sets and use cases throughout the project, we later added a web server (Apache) access log [31]. Web server logs can help understand security breaches involving the application layer which our experts reported has become more relevant in recent years.

3.1 SSH Log

Our ssh log captures the login activity via the ssh protocol to a server of a large scientific institution. Our specific set was from a previous incident analysis and we knew already that it contained two successful breaches. Stripped down to the entries relevant for login violations, it contains 13,199 lines (7 days). From the seven parameters of each record, we selected *time stamp, log message* (authentication method and whether it was successful), *source IP*, and *user name* for our visualization as they are most relevant for detecting login violations. *Source port* and *destination port numbers*, and *protocol* were not relevant or constant.

Together with the file, we got two "challenges" from the security engineer working with us on this phase. One was to find a "brute force attack", i.e. an attacker tries a series of passwords (usually via a computer program) and when she guesses the password, she will be granted access. In the log, we would see a series of failures, followed by an "accepted password" indicating the success. The difference between failures from typos of a legitimate user and a (usually scripted) brute force attack are not always clearly assessable and then require inspection by a security expert. The other was a suspected public key theft. Public keys are usually stored on a computer and used instead of a password. They are too complex to be guessed. We had to look for an "accepted public key", where the other parameters or the context were unusual.

This log file was also meant to be a "training" or "test" file: while it would be possible to solve the use case itself automatically, we wanted to develop our visualization with the help of a well known challenge.

To find evidence about malicious activity, we needed to inspect complex data: multi-variate data, where one record has several parameters, and also multi-event data, where only a combination of records indicates malicious behavior.

3.2 Apache Access Log

The Apache (web server) access log records requests from clients, most commonly when someone accesses a web page, with images, stylesheets, etc. We used logs from the web presence of a medium sized company (146,655 lines spanning seven days) and a private website (4,481 lines, one day; 33,060 lines, seven days). With the standard logging settings, there were eight different variables: *source IP/host*, *time stamp*, *URL* (i.e. requested resource), *response code* (which tells whether the request was acknowledged or produced some kind of error), *bytes sent*, and *user agent*. For most of our work, we chose *host*, *response code, URL*, and the *time stamp*. With these parameters, we can answer who did what with which result. We added the *source country* for each IP via a IP-to-geolocation service.

Recorded activities in Apache logs are more diverse and the communication is less restricted. We worked with a security engineer and a web master, who wanted to investigate the activities on their web server. A deducted use case is to find attackers via traces in this log when they try to get access to restricted files. This will be a rare event as regular visitors only access public files. Another example are calls of scripts with unusual parameters, such as SQL injections to break into an application running on the web-server.

4 Pixel Carpets for Multivariate Data

Based on the data and the design requirements elaborated before, we developed a visualization in the form of a pixel map. Next, we discuss our design decisions regarding visual mapping and interaction.

4.1 Reasons For a Pixel Map

Early ideas for a tool that puts raw data visualizations next to automatically filtered alerts already arose during the workshop (called the "split screen", Fig. 2). Building on this and the strong requirement for a transparent tool (regarding the processing of raw data, DR1) that invites for exploration (DR3), we were looking for a way to bring lots of data records onto one screen (DR4). We found pixel maps [14] particularly appealing as we deal with very large datasets, require decent overview, and did not want to conceal details by binning the data. They offer the best "data to space ratios" by using the smallest building block on a computer screen (1 pixel) per data record, at least in the extreme. For high density displays (>100 pixels per inch), the minimum size has to be adjusted.

Overview is important when the security engineer needs to explore the dataset first and does not know what to look for initially (DR3). With several hundred thousand lines recorded per day, traditional displaying techniques fail this task [20].

Binning would be a common technique to visually compress a dataset: aggregating several data records and representing them just with their average value (the most common form for this is a histogram). But when looking for exceptions and outliers (DR2), this technique can blur away the rare and hence important records

quickly. Additionally, binning works best for numeric values. But in network security, most values are categorical in nature (e.g. IPs) which have no average value and are therefore hard to bin.

4.2 Carpet Layout and Multi-Pixel Structure

We decided to give three parameters of a record their own pixels, creating columns of "multi-pixels" per record (see Fig. 3). For the ssh analysis, three parameters are sufficient and it is easy to read. An example for such an analysis is the detection of a dictionary attack. *Source country* and *user name* would stay the same but *log message* changes from "failed password" to "accepted password". We built the multi-pixels as vertical columns and arranged them from left to right. As Fig. 3 shows, each record is lined up after the other, each pixel representing the value by color, with line breaks as the screen layout or the application requires. This allows an intuitive flow of reading from old (top left) to new (bottom right, for western reading habits). The result of this layout, together with the coloring, is what we call the "Pixel Carpet".

Triple pixels decrease our "data record to space ratio" from 1/1 to 1/3 but we gain a much better insight into details (we need to balance DR1 and DR4). For the Apache log, we also tested five parameters per record. To enhance readability and to better separate the multi-pixels, we added a padding of one pixel horizontally and four pixels between the lines. We usually worked with 16 screen pixels per data item (4×4) because pixels are better readable and easier accessible with a mouse for additional info on hover (Fig. 3). This results in a "data/space ratio" of 3/80 including the padding. On a medium sized screen with a resolution of 1600×1200 (1.920.000 Pixels), we could show 24.000 entries at a "three parameter-resolution" simultaneously. For larger datasets, this can be reduced down to one pixel for data points and paddings (two to separate lines), yielding a ratio of 3/10. A magnifying tool for the cursor will then be necessary, such as a fish eye (more on scalability in section 7).

4.3 Color Mapping

While the focus on pixels is space efficient, it limits the parameters available for visual mark-up. Pixel color is the most important option. Outlines/borders and patterns, as two examples, require their own pixels just for making them visible. A distinct color for each value would be a good option, but human perception is quite limited in freely discerning colors, ruling out this option [32].

We want rare values to stand out and frequent values to fade into the background (DR2). For this, we colorize pixels based on the value frequency in the dataset, similar to a heat map. We focus on parameters, i.e. frequency is calculated for each parameter in relation to all other values of this parameter in the dataset, independently of the record it belongs to (also independently of the occurrences of other parameters). Example: if "accepted password" can be found five times in all *log message* fields of the dataset, this is its frequency and determining its color, regardless of the values for *source IP* or *URL* in the corresponding records.

To map frequencies onto color values, we started with counting the occurrence of each parameter across the data set. We then segmented this frequency distribution into ten slices of varying "width": the first group consists of the rarest 0.5% of frequencies, the next color for frequencies up to 2%, then in increasing steps from 5%, 10%, 20%, 32%, 46%, 62%, 80% to the most frequent parameters. We did so to "sharpen" the rare groups that we are most interested in and have a more coarse representation of frequent values. We then created 10 different colors between bright red and dark blue by segmenting the hue and luminance "distances" between these colors in a HSL (hue, saturation, luminance) color model into equal steps. Finally, we mapped the groups from the value frequency to the corresponding color, with red for rare and blue for frequent (as in Fig. 3).

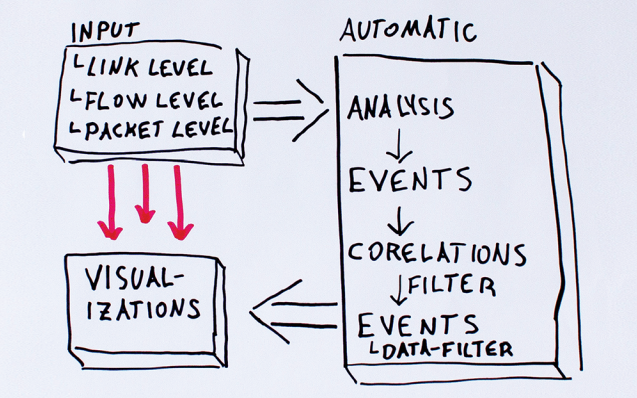

Fig. 2: One of the sketches where the workshop participant wanted to have a direct visualization of raw data (red arrows), complementing an automated analysis.

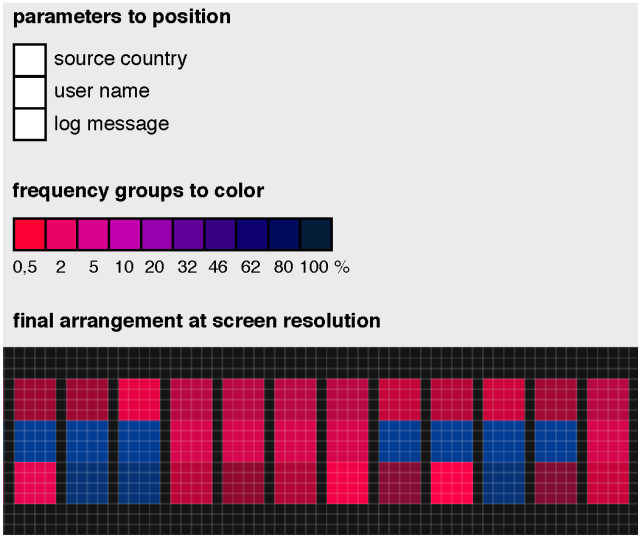

Fig. 3: A look at the construction of the Pixel Carpet with multi-pixels and the "implementation" at screen resolution. One "column" is representing one data record.

Coloring by value frequency can result in two different parameter values, such as two different *user names*, having the same color because they occur equally often. Especially when these pixels appear next to each other, this misleads the user to (intuitively) think the two values were identical. To avoid this impression, we vary colors for different parameter values by small random changes in the luminance. Two pixels that stand for values of equal frequency will thus vary slightly but noticeably. This is also possible because the exact frequencies are not important for the security expert whose main goal is getting a general understanding of the dataset and finding regions of interest.

We implemented a straightforward algorithm for our experiments with the aim to make its mechanics transparent and gain the users' trust. We do not rely on fields that only occur in specific logs so that the color mapping works for quite different types of log files (DR5).

4.4 Exploration and Filtering

Complementing the visual representation of the Pixel Carpet, we now describe its interaction techniques designed with the aim to support open exploration of log files (DR3 and DR4).

Fig. 4: Video showing the main interactive techniques currently implemented for the Pixel Carpet: highlighting of identical records, tool tips and clear text display, and filtering/removal on click. The numbers in the Carpet refer to the hours from the time stamp. Two consecutive numbers mean that all records in between got filtered away. (Data from ssh log file, IP addresses replaced for publication).

4.4.1 Highlighting Identical/Similar Records

Hovering over a multi-pixel representing a single record, all other records with identical parameters are highlighted (Fig. 4). This dynamic effect complements the static visualization: where the visualization makes regional similarities catch attention, such as several rare records forming a red block, the highlighting tool finds matches across the whole set. This reveals activity over time.

Via checkboxes in the interface, users can choose which parameters are taken into account for the matching: maybe they want to inspect a series of login attempts and want to see whether there were any successful guesses. They then switch off matching on *log message* and the tool will highlight all entries with the same *source country* (or *source IP*), same *user name*, but any kind of *log message*. With this highlighting tool, users can inspect elements and verify hypotheses that they built based on their visual impression.

4.4.2 Clear Text Display

Besides highlighting similar records, the hover operation also shows a clear text display of the corresponding log record (DR1), as a tool tip and in a dedicated log file window below the visualization (Fig. 4). The log file window also shows the neighboring log lines to provide more context and allow for faster reading. The users could even scroll through the log file via the visualization, but the power of the visual representation is a non-linear reading. In effect, the users get a quick overview over the contents of the dataset. Experienced network engineers know many *IP addresses* and *user names* in their network. They can put the values they find in context and thus classify them swiftly and precisely.

4.4.3 Filtering records

Clicking on a multi-pixel will remove all entries with identical parameters from the visualization. The users can get rid of, e.g., very frequent and thus uninteresting log records. The colors for the dataset will be recomputed based on the frequencies of all data remaining on display (Fig. 4). With some of the values removed, the range of frequencies (we could also say "dynamic range") gets smaller. The limited set of (discernible) colors can then be spread across a narrower band of frequencies, letting smaller differences show up. This makes the filter actually a sort of zoom into the data.

4.4.4 Additional Binning

For some log files (especially our Apache ones), highlighting just identical values can be too strict. For example, logs of dynamically generated websites will contain records of many slightly varying requests. A strict coloring would render all of these requests as very rare and red (Fig. 7 top), while in fact, they are all part of the published online resources. For this reason, the visualization allows to "summarize" variations via text matching. All of the affected parameters will be considered "the same", represented by the same color (Fig. 7 bottom). In effect, this bins the parameters. As mentioned in the previous section, this helps to find the abnormal activity and separate it from just individual but legitimate requests. A simple text filter as in our demonstrator, however, is hardly capable of reliably differentiating trusted *URLs* from malicious requests in practice.

4.5 Color patterns encoding activity

Our idea is that different "classes" of records become easily visible to the human eye when we colorize the individual parameters (i.e. pixels) of each record (i.e. multi-pixel). Certain combinations of colors within a multi-pixel would then indicate certain activity. These combinations can be found quickly when visually scanning the Pixel Carpet (DR1).

This is clearly visible in the ssh log file: there are two types of "entirely frequent" entries with three blue pixels (Fig. 5). Closer inspection (details in 4.4 Exploration and Filtering) reveals that one is a service the institution installed, the other is a brute force attack on "root". While we initially thought that the attack should look more alarming, our security expert agreed with the modest appearance as the attack obviously did not succeed (otherwise it would have a red bottom pixel for an "accepted password"). We can also see a record in Fig. 5 with a pink middle pixel, which comes from an attack that tries out user names. Another example shows the same intention but originating from a computer from a different country.

Besides single record classification, the Pixel Carpet also allows to view records in context (DR4). Especially in the quite homogenous ssh log file example, we can observe time spans "full" of regular behavior and then one record standing out because of a colored pixel. Or we have series of malicious activities that are visible as reddish blocks or even stretches of color (also in Fig. 5).

For an Apache log file, the initial appearance is less clear. To a large part, this is due to the huge variety of normally available resources such as web pages and images. While this might hide smaller items and single records, we can see high frequency activity and repeatedly occurring events very clearly in Fig. 6. This way, patterns evolve from a log file visually that would be hard to recognize with other techniques, especially with the traditional "plain text and grep" method. To better separate truly suspicious activity from the diversity of legitimate activities such as the case for *URLs* in web-server logs, we also created a special binning mechanic, by which the security expert can influence the coloring (details in 4.4).

Finding "classes" of activity from the visual appearance already works to a certain extent but also has its limits. The algorithm we apply for color mapping is straightforward but also rather basic. Evaluating a "group" of records that come from the same IP/user,would allow a second pass for fine tuning colors or intensities. If, in an ssh-example, "root" would "normally" log in via key-file but now logged in (successfully) via password, this "accepted password" is very abnormal, while other users might login via password regularly. In "7 Discussion", we discuss how colorizing by value frequency could be replaced with more sophisticated mechanics to (pre)discover anomalies.

5 Evaluation

The co-creation process comprised a continuous feedback loop, throughout which our collaborators informed the design of the visualization with their observations and feedback. In the following, we summarize the most important findings we gained with regard to the Pixel Carpet visualization.

5.1 Expert Feedback on the Pixel Carpet

5.1.1 Setup of Feedback Sessions

In this phase, we worked with people or groups from four different organizations. Two took part in our interviews and workshop: one person was responsible for network planning and computer security at a scientific supercomputing centre (#1) and the others were product managers from a security appliance vendor (#2). Two people joined later: a security manager in a company for electronic payment (#3) and a system and security administrator of a medium sized company (#4). Each evaluation session was one to one, on location or web based with screen sharing. We briefly explained how the visualization got generated and how the main tools worked and then let the experts explore the visualization and the datasets. All participants investigated ssh- and Apache-examples. Each session took between one and two hours and got audio recorded if allowed by the interviewee. We also worked in a team of two and took notes. Later, we analyzed our findings in categories such as general security strategies at this organization, feedback on the visualization, and feedback on the interaction and the software interface.

With interviewees #1 and #3, we also discussed different approaches to visualizing the ssh log file that also evolved from the same co-creative process: parallel coordinates and a combination of scatter plot and slope graphs.

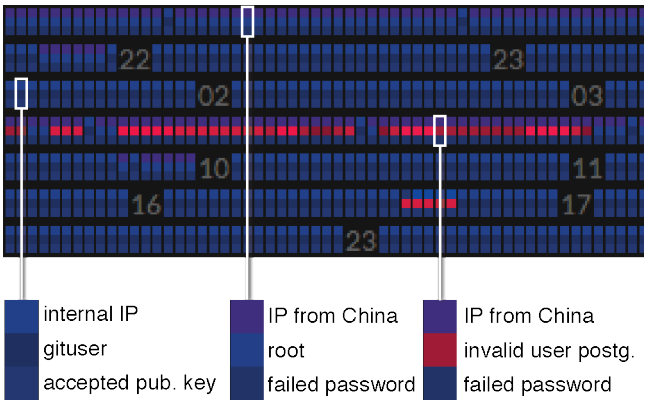

Fig. 5: Specific color combinations that represent specific activities on an ssh server: regular access (left), password guessing (middle), user name guessing (right)

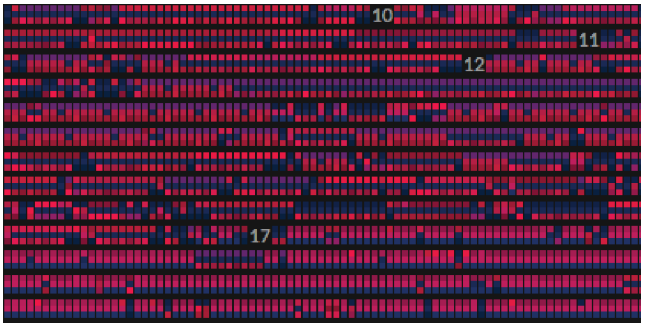

Fig. 6: Detail view on a large Apache access log file. Note the bands of similar activity. In the upper right corner, close to hour 10, there is a block of rare records.

Fig. 7: Effects of "binning" different values of a parameter together. In this case, we instructed the color mapping to treat all jpeg images (as part of the *URL*) as if it was one and the same. While each image on its own is rarely accessed (indicated by red pixels in the top band), jpegs in general are a usual request (blue pixels in the bottom band).

5.1.2 Important Overarching Findings

In case of the ssh log file, the experts could quickly check wide spread, automated login attempts with the help of pixel colors and the highlighting tool. These are usually turned away by the authentication system (”failed password”) but in one case, the attempt was successful. The change in the colors of the multi-pixels at the end of one of the series caught the attention of expert #1 early on in the investigation : “The 'accepted password'-pixel stands out clearly.” (Fig. 8) The login with the (most likely) stolen key file was initially harder to see. It came to light when the expert filtered away all records he knew were legitimate and from trusted machines. It gave itself away by its rare *source IP*.

Considering the color mapping algorithm, experts #2 and #4 wanted to have a better “semantic” differentiation, i.e. a “failed password” should not resemble an “accepted password”. They would prefer two clearly distinct color hues.

Investigating and interpreting the Apache log files turned out to be more complicated to start with. As mentioned earlier, it is quite diverse in regards of visitors (*source countries*), web-server response codes, and resources (*URLs*) accessed. Binning trustworthy entries together and/or removing them from display was highly efficient, although the experts missed regular expressions and boolean search to better specify their filter criteria. After some filter iterations, the visualizations cleared up. The log from the private web-server showed huge amounts of requests that were obviously checking for flawed plugins and potential misconfigurations: a typical preparation for attacks on applications. Several experts suggested to save their manually constructed filters to apply them automatically in future sessions. Even more, They liked the idea to construct a filter this way with visual and interactive support, a feature they were missing in their IDS rules creation tools.

The web master #4 commenting on this use case wished for an option to automatically classify “legitimate visits”, “search engine robots”, and everything else. The multi-pixels themselves were ambiguous in this area. The two experts #1 and #3 pointed out that they liked the visualization because they stayed in control of classification, and that no automated system had filtered the data by some hidden algorithms. But they also stated that they preferred a tool that shows more and requires less clicks. This topic needs careful balancing. For the approach described here, we focused not on making the root causes of incidents immediately clear from the visualization but to find areas of interest and get a “feeling” for the dataset. Our aim was guiding the focus of the experts so that they find the items to inspect more quickly, not classifying the dataset automatically.

Due to space constraints, we do not discuss the alternative visualizations (parallel coordinates and scatter plot variant) in greater detail Two key differences are: In contrast to the Pixel Carpet, they do not show events in chronological order which we found important to understand activities. The parallel coordinates need at least user input in the form of brushing to detect patterns. Especially with the huge number of different values in the Apache log, both approaches suffered from overplotting.

5.1.3 Additional Findings

It turned out that the experts could comment best on Pixel Carpets based on data from their own networks (the dataset itself, however, could be unknown to them or previously uninspected). Data from other systems still revealed patterns to our test participants but it was harder to classify them: activity that is normal in one system can be exceptional in another. Regularly repeating patterns can originate from a script that the administrators implemented themselves or from an automated attack from the outside.

The first time the participants saw the Pixel Carpet, it was quite unfamiliar to all experts. As the highlighting tool reacts on hover, they were drawn towards exploring how the visualization works

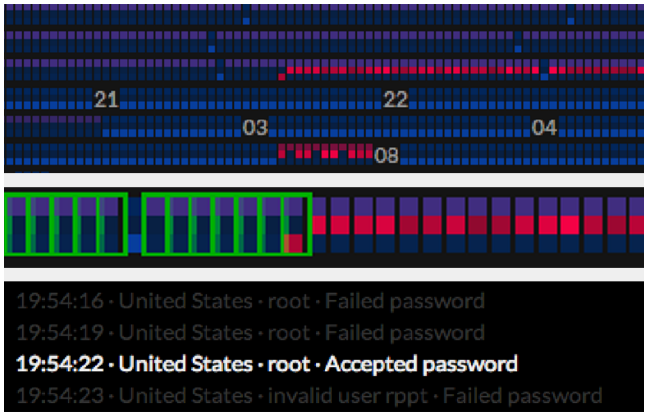

Fig. 8: Detecting a successful brute force attack in an ssh log: the red pixel in the lowest row of a multi pixel indicates a rare value, while the other parameters appear to stay the same (top). The highlighting tool reveals a series of attempts (middle, enlarged), confirmed by the clear text output (bottom).

and what they can actually see. Generally, the participants responded interested and favorably to the interface: “I have never seen something like this, which I mean in a positive way.” (#2) The mechanics of filtering and binning needed some explanations from our side.

All experts agreed that the visualization was particularly suited for the inspections of log files after an incident (”post mortem” analysis) and “when you don't know (yet) what you are looking for” (#3). It would even help understand unfamiliar datasets because of the clear and easy to understand visualization mechanics. They admitted that log file inspection was known as an important measure that they should perform more often but see as too tedious. “Continuous monitoring should still be done by automated scanners. But if I had a tool like this, I would certainly look at logs more often.” (#4)

6 Related Work

After we have laid out the key idea of the Pixel Carpet and the co-creative approach that led us to it, we want to put it into context with existing approaches.

Livnat et al. combine several visualization techniques (circular layout, network graph, geographical maps) to reveal various kinds of relations and anomalies [19]. Their main goal is situational awareness, i.e. the big picture, but they also allow access to traffic details. The remarkable effectiveness of their tool comes from a design tailored to flow records (e.g. network topology) that would not easily translate to web server logs (our DR5). In contrast to that, Humphries et al. follow a log type agnostic approach with ELVIS [10], which features smart mechanics to determine the type of data fields and to propose adequate (basic) visualizations. These basic mini visualizations already provide some statistical overview on a per field level. They can be easily combined (drag and drop) to form new visualizations. While the field extraction and statistics features are promising, the Pixel Carpet brings overview capabilities and a chronological view that are less developed in ELVIS.

Phan et al. [25] rely on external triggers to start an investigation with their system “Isis”. Once the investigation is narrowed down to a single IP, they offer a smart matrix view to analyze and reconstruct the course of events. While their display of time in an “ordinal space” is similar to the Pixel Carpet, it needs significant filtering before it shows up and details are revealed.

Phan et al. [25] and even more Xiao et al. [35] describe in detail how the analyst is involved in the classification of events. Ro-

gowitz and Goodman have the notion of a "human in the loop" [26] that describes a man *and* machine system that combines the pattern recognition abilities of both and includes a feedback mechanism for iterative analysis. This resonates with the requirements that our interview partners had (DR1, DR3). Shneiderman emphasizes that the system's operations should be transparent to the people so they can trust the results and take on responsibility for the conclusions they draw [28].

Conti et al. [3] and Weseloh [33] propose pixel maps directly for computer security visualizations. They use a simple form with one (data) dimension and a single color. Especially Conti et al. point out, how this solution lets patterns emerge visually and thus supports human pattern recognition. Both visualizations are embedded into analysis tools that at least in parts are used in productive workflows. Interaction with these visualizations, however, appears to be limited to viewing results and retrieving clear text/context information. Direct manipulation in the sense of visual analytics seems not to be implemented (DR4), such as filtering from within the visualization or dynamically displaying filter matches, something that our user group found very valuable. Chung et al. [2] show pixel oriented techniques for big security data sets, using a very large physical screen for overview with smaller, individual screens for detail inspection. They do not discuss the option of stacking pixels and instead distribute different aspects over different views.

Pixel maps are fairly straightforward visualizations and have been around for a long time. This enabled us to build on a stack of work, such as Keim et al. [14] and Lammarsch et al. [17]. Keim et al. have thoroughly researched display techniques and applications and offer sophisticated ways to map data to various visualization properties (x, y coordinates, display sections, color, intensity). They usually distribute parameters of a record into different "sub windows" of their display. It is easier to read but makes it harder to find differences in combinations of parameters.

In our approach, we integrated those parameters, which brings us closer to the idea of Levkowitz' Color Icons [18]. This concept suggests to create matrices of pixels, with one matrix representing one entry of a multi-dimensional dataset. The advantage is that a whole data record is shown close together. As Levkowitz points out, this improves revealing patterns and relationships when combined with the matrices of the rest of the data, without the need for user activity, such as brushing. We found huge amounts of matrices visually confusing, however, and limited ourselves to a single "column" of pixels per dataset entry.

Working with categorical data (such as *source countries*) in pixel maps needs some transformations as pixel color is easiest mapped to numerical values. Keim et al. have shown an implementation in [15] but it is quite limited in the number of categories it can hold so that we saw room for improvement.

While pixel maps are mostly used because of their overview capabilities, Janetzko et al. put their focus on highlighting anomalies [11]. The results from their expert evaluation indicate that analysts can find anomalies quickly and easily (DR2) while maintaining overview and their orientation in time (DR4). We also share the chronological pixel arrangement with their work but extend it towards stacked pixels and the work with categorical data.

Visualization projects and research follows user centered design principles on a regular basis [5], [16], [20]. Meyer et al. have defined a profound framework for the process and the validation of important steps [22]. They have also defined a set of roles that a larger data visualization project can involve and that need to collaborate for optimal results [16]. It involves the domain experts and visualization experts, of course, but also specialists for high-throughput computing. Truly participatory processes are still quite rare in this field. The occurrences we found in security visualization (e.g. [9]), described the techniques and results quite briefly. We provide a case study that implements participatory elements, something that Meyer et al. wish to happen more in this domain.

7 Discussion

7.1 Moving to Larger Data Sets

For most of our exploration, we worked with excerpts containing less than 10,000 lines which is over a magnitude below the intended target application at larger data centers. We worked with datasets exceeding the available screen size and one general finding is that the coloring algorithm works better the more data it covers (the larger the mass, the more the exceptions stand out). We did not investigate "big data" issues in practice, mainly because of the considerable engineering efforts required, instead our interest was in rapid prototyping of visualizations. We do not propose the Pixel Carpet as "first line of defense", e.g. in live monitoring. We rather think that it is used to explore alerts with vague indications, which means that the range of time windows or machines (IPs) is already somewhat limited. The advantage (e.g., over the previously discussed "Isis" [25]) is that the indicator can be quite imprecise because of the overview capabilities of the Pixel Carpet. Beyond that, we see several options for visual compression techniques that try to preserve small values, such as anomalies (e.g. [12]). The Pixel Carpet can also be combined into other visualizations without giving up its advantages (Fisher et al. provide ideas in this direction [8]). Assuming that most security engineers work in dedicated offices, we also see the option to increase the number of physically available pixels by working with a wall of screens (building on Chung et al. [2])

7.2 Human and Machine Based Detection

As laid out before, the system should prepare and present the data with the goal to harness human pattern detection. Major decisions should be left to the user to provide room for exploration and intervention. User-driven does not mean purely manual: We have an early and fairly simple algorithm already in place to "preprocess" the data in our current proposal. Researching more powerful algorithms, including cluster analysis and anomaly scoring (as, e.g. in [11]) and selecting one that fits with our goals is an important area for future improvements. The Pixel Carpet is meant to complement highly automated systems (such as IDS) in a future security workbench as it has been suggested in on of the workshops (Fig. 2). We need to carefully balance automated processing, visualization, and user control.

Based on our experience with ssh- and apache-log files, we see testing with a broader user base and a wider set of use cases as a useful next step. A structured comparison, e.g. with the "sub window" approach of Keim [14], is necessary for assessing specific features of the Pixel Carpet. A formalized evaluation process also complements the more empathic co-creation very well and corresponds to the "validation" requirements of Meyer et al. [22]. While continuous feedback is a part of co-creation anyways, the distinct levels of their "Nested Model" will help to create focused sessions at the right moment.

7.3 Insights into Co-Creation in Data Visualization

From our work as designers, we had experience with several of co-creative methods. Some more general principles of co-creation held true also for the area of data visualization, such as profiting from highly diverse teams, fast exchange of information, no hierarchies, and an encouraging and inspiring atmosphere. Asking everyone to work visually with pen and paper (instead of, e.g., writing down requirements) also helped to output first ideas and to get to their essence.

Then again , we had not put enough effort in integrating data into these "analog" exercises, which lead to overly optimistic as-

sumptions in the sketches about data structures or values. For evaluation (and refinement), a "quick and dirty" tool or method to check an idea would be necessary. So far, we are still looking for a method that is quick and accessible enough to fit into a fast paced, trial and error-based ideation workshop.

This became also apparent in the subgroup directly focusing on software tools and trying to put together analysis code during the workshop. We had underestimated the amount of time it takes for getting a common understanding of the structure and contents of the data. Starting with programming more or less from scratch under these conditions and within few hours turned out to be impossible. The participants reported that the intensity of fruitful exchange in this session was extremely high. Without the "crazy endeavor" of live coding, it might not have happened. An improvement for future meetings would be to work with a single dataset, that everyone can analyze beforehand, possibly already write code fragments for it , and also to have more modular data analysis and visualization tools in place. In many cases, some features of a dataset must be extracted algorithmically to create a really meaningful visualization on top of it.

Data can be considered the original material a visualization is crafted from, data comes first, visualizations should match its conditions. We have to pay attention, however, so that the rather abstract data sets do not damp the generation of concrete ideas in the early phases. Distilling data structure, some special features or defining challenges could be a way to get it integrated productively. It is also worth mentioning that getting hands on datasets of reasonable size and quality from the domain in question needs enough time to solve organizational, technical, or privacy questions. Putting datasets right into the early ideation processes is a highly promising strategy for adapting the co-creative approach.

8 Principles for Co-Creation in Data Visualization

Based on our experiences and the reflection of what worked and what did not (s. section 7.3), we formulate tentative guidelines for a co-creative approach to data visualization projects. They are organized roughly by project phases but loops and iterations are a typical feature of this approach.

Recruiting. Try to establish a team of volunteers (stakeholders) that can commit enough time to participate in the process. You do not want to "touch and go", obtaining an interview and continuing on your own. McLachlan et al. have a good description of challenges in recruiting highly busy experts and in iteratively adding new stakeholders [21].

Domain understanding. Get immersed into the topic as deeply as possible. We used the ethnographic methods of interviews and observations as described by D'Amico et al. [5]. An even better approach is to aim for a shared understanding on both sides (security and visualization) through joint, explorative, hands-on creation sessions (D'Amico et al. propose "hypothetical scenario construction", Sanders has a more in-depth description and argument in [7]). Discuss how your collaborators work with data, and try to pin down the results precisely as use cases. It forces you to be very clear about the individual steps and details, and structures the process into operational modules (D'Amico proposes Cognitive Task Analysis [5], Cooper has tasks as part of his Goal Directed Design [4]).

Requirement definition. Define and refine challenges with all stakeholders, ideally in a face-to-face workshop. Achieving a common understanding is extremely valuable to focus further efforts. Converging on a few essential requirements with a diverse team, however, is not always possible. Defining the goals together also strengthens commitment for the process ahead (also reported in [9]). It might be worth noting that this co-creative way of requirement distillation builds on communication rather than quantitative evaluation. Why and how something is relevant is given more weight than how many find it relevant (group sizes are comparatively small, too).

Data acquisition and inspection. Try to get real-world data sets from your target users that are typical and relevant for them. Take care of this as early as possible as privacy and the sensitive information contained may pose considerable and time-consuming hurdles. Investigate the data with the tools of your choice to get an understanding of its structure and quality (McLachlan et al. even set up their own monitoring tools in their network [21]).

Ideation. Brainstorm and sketch ideas together. Pen and paper are quick and easily accessible. They are well suited to frame ideas and describe the goals for a tool. Ideation tasks for these sessions must be well defined and "entry barriers" kept low to encourage all participants (example schedules and tools in [29]).

Ideas from data. Work and sketch with data as a "material" early on and incorporate it into creative sessions. Characteristics and challenges of real data are valuable input for new solutions. As it takes considerable amounts of time to familiarize with a data set, workshop participants should get the (same) data sets beforehand and inspect them as a "homework". This requires digital tools which bring their own challenges for joint creative work. Software that accepts a wide range of data types and offers a large number of presets seems promising, such as Tableau [30].

Prototyping and validation. Focus on the aspects (such as design requirements) you want to validate first with early versions of the visualization. Adapt your questions and evaluation methods to the state and focus of your sketches (s. Munzner et al. for adequate methods [23]). Rapid prototyping methods have been described at length so that you can choose what fits your needs and taste. Depending on the skill set of your stakeholders, they might be even able and willing to contribute their own code modules.

9 Conclusion

We have shown how the visualization approach of a pixel map variant can turn unlabeled data into an image that makes this data easier to survey and inspect. With different multivariate datasets we explored how to better show their structures and the activity patterns they contain. Our proposal also encompasses interactive tools that allow users to analyze their data visually.

The intended audience of network security engineers welcomed this visual approach to "their" log files because it allows them to understand large amounts of logs more quickly and find patterns they could not find with their current tools. The early prototype that we describe in this paper led to valuable suggestions for improvements and new extensions.

Throughout the process, we involved security experts as user group and experienced how valuable continuously available consultants and feedback partners are. As visual analytics becomes even more important with large datasets ("Big Data"), new expert tools need to be developed. They can only be effective (and accepted) when visualization specialists understand the domain experts' needs so that they can design appropriate tools.

ACKNOWLEDGMENTS

We want to thank our interview participants from the computer security domain for sharing their knowledge with us. Special thanks to Leibniz Super Computing Centre for helping with test datasets and intense discussion and support. We are also grateful to the students and teachers at Potsdam University of Applied sciences for their help and reviews, and to the conference reviewers for their detailed and valuable feedback. This work has been supported by the German Federal Ministry of Education and Research (BMBF) under support code 01BP12303; EUREKA--Project SASER.

REFERENCEs

[1] R. Bejtlich, *Tao of Network Security Monitoring, The: Beyond Intrusion Detection*, 1st ed. Addison-Wesley Professional., 2004.

[2] H. Chung, Y. J. Cho, J. Self, and C. North, "Pixel-oriented treemap for multiple displays," in *2012 IEEE Conference on Visual Analytics Science and Technology (VAST)*, 2012, pp. 289–290.

[3] G. Conti, E. Dean, M. Sinda, and B. Sangster, "Visual Reverse Engineering of Binary and Data Files," in *Proceedings of the 5th International Workshop on Visualization for Computer Security*, Berlin, Heidelberg, 2008, pp. 1–17.

[4] A. Cooper, R. Reimann, and D. Cronin, *About Face 3: The Essentials of Interaction Design*. Wiley, 2007.

[5] A. D'Amico, K. Whitley, D. Tesone, B. O'Brien, and E. Roth, "Achieving Cyber Defense Situational Awareness: A Cognitive Task Analysis of Information Assurance Analysts," *Proc. Hum. Factors Ergon. Soc. Annu. Meet.*, vol. 49, no. 3, pp. 229–233, Sep. 2005.

[6] T. K. Dang and T. T. Dang, "A survey on security visualization techniques for web information systems," *Int. J. Web Inf. Syst.*, vol. 9, no. 1, pp. 6–31, 2013.

[7] Elisabeth Sanders, "Virtuosos of the Experience Domain," in *Proceedings of the 2001 IDSA Education Conference*, 2001.

[8] F. Fischer, J. Fuchs, F. Mansmann, and D. A. Keim, "BANKSAFE: A visual situational awareness tool for large-scale computer networks: VAST 2012 challenge award: Outstanding comprehensive submission, including multiple vizes," in *Visual Analytics Science and Technology (VAST), 2012 IEEE Conference on*, 2012, pp. 257–258.

[9] N. Henry and J.-D. Fekete, "Matrixexplorer: a dual-representation system to explore social networks," *Vis. Comput. Graph. IEEE Trans. On*, vol. 12, no. 5, pp. 677–684, 2006.

[10] C. Humphries, N. Prigent, C. Bidan, and F. Majorczyk, "ELVIS: Extensible Log VISualization," in *Proceedings of the Tenth Workshop on Visualization for Cyber Security*, 2013, pp. 9–16.

[11] H. Janetzko, F. Stoffel, S. Mittelstädt, and D. A. Keim, "Anomaly detection for visual analytics of power consumption data," *Comput. Graph.*, vol. 38, pp. 27–37, 2014.

[12] D. F. Jerding and J. T. Stasko, "The Information Mural: a technique for displaying and navigating large information spaces," *IEEE Trans. Vis. Comput. Graph.*, vol. 4, no. 3, pp. 257–271, Jul. 1998.

[13] M. Kalra, "Announcing the Release of Splunk Enterprise 6," *Splunk Blogs*, 01-Oct-2013. .

[14] D. A. Keim, "Designing pixel-oriented visualization techniques: Theory and applications," *Vis. Comput. Graph. IEEE Trans. On*, vol. 6, no. 1, pp. 59–78, 2000.

[15] D. A. Keim, M. C. Hao, U. Dayal, and M. Hsu, "Pixel Bar Charts: A Visualization Technique for Very Large Multi-Attribute Data Sets†," *Inf. Vis.*, vol. 1, no. 1, pp. 20–34, 2002.

[16] R. M. Kirby and M. Meyer, "Visualization Collaborations: What Works and Why," *Comput. Graph. Appl. IEEE*, vol. 33, no. 6, pp. 82–88, 2013.

[17] T. Lammarsch, W. Aigner, A. Bertone, J. Gartner, E. Mayr, S. Miksch, and M. Smuc, "Hierarchical temporal patterns and interactive aggregated views for pixel-based visualizations," in *Information Visualisation, 2009 13th International Conference*, 2009, pp. 44–50.

[18] H. Levkowitz, "Color Icons: Merging Color and Texture Perception for Integrated Visualization of Multiple Parameters," in *Proceedings of the 2Nd Conference on Visualization '91*, Los Alamitos, CA, USA, 1991, pp. 164–170.

[19] Y. Livnat, J. Agutter, S. Moon, and S. Foresti, "Visual correlation for situational awareness," in *Information Visualization, 2005. INFOVIS 2005. IEEE Symposium on*, 2005, pp. 95–102.

[20] R. Marty, *Applied Security Visualization*. Pearson Education, 2009.

[21] P. McLachlan, T. Munzner, E. Koutsofios, and S. North, "LiveRAC: interactive visual exploration of system management time-series data," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2008, pp. 1483–1492.

[22] M. Meyer, M. Sedlmair, and T. Munzner, "The four-level nested model revisited: blocks and guidelines," in *Proceedings of the 2012 BELIV Workshop: Beyond Time and Errors-Novel Evaluation Methods for Visualization*, 2012, p. 11.

[23] T. Munzner, "A nested model for visualization design and validation," *Vis. Comput. Graph. IEEE Trans. On*, vol. 15, no. 6, pp. 921–928, 2009.

[24] B. F. O'Brien, A. D'Amico, and M. E. Larkin, "Technology transition of network defense visual analytics: Lessons learned from case studies," in *Technologies for Homeland Security (HST), 2011 IEEE International Conference on*, 2011, pp. 481–486.

[25] D. Phan, J. Gerth, M. Lee, A. Paepcke, and T. Winograd, "Visual Analysis of Network Flow Data with Timelines and Event Plots," in *VizSEC 2007*, J. Goodall, G. Conti, and K.-L. Ma, Eds. Springer Berlin Heidelberg, 2008, pp. 85–99.

[26] B. E. Rogowitz and A. Goodman, "Integrating human- and computer-based approaches to feature extraction and analysis," 2012, vol. 8291, p. 82910W–82910W–11.

[27] E. B.-N. Sanders and P. J. Stappers, "Co-creation and the new landscapes of design," *CoDesign*, vol. 4, no. 1, pp. 5–18, 2008.

[28] B. Shneiderman, "Inventing discovery tools: combining information visualization with data mining1," *Inf. Vis.*, vol. 1, no. 1, pp. 5–12, 2002.

[29] F. Sleeswijk-Visser, P. J. Stappers, R. V. der Lugt, and E. B.-N. Sanders, "Contextmapping: Experiences from practice," *CoDesign*, vol. 1, pp. 119–49, 2005.

[30] Tableau Software, *Tableau*. Seattle, Washington, United States: Tableau Software.

[31] The Apache Software Foundation, "Log Files - Access Log," *Apache HTTP Server*, 2014. [Online]. Available: http://httpd.apache.org/docs/current/logs.html#accesslog. [Accessed: 25-Feb-2014].

[32] C. Ware, *Information Visualization: Perception for Design*. Elsevier, 2012.

[33] M. Weseloh, "Network Security Visualization Techniques in Early Warning Systems," presented at the 1 st European Workshop on Internet Early Warning and Network Intelligence, Hamburg, 27-Jan-2010.

[34] WikiBooks, "OpenSSH/Logging," *OpenSSH*, 2013. [Online]. Available: http://en.wikibooks.org/wiki/OpenSSH/Logging. [Accessed: 25-Feb-2014].

[35] L. Xiao, J. Gerth, and P. Hanrahan, "Enhancing Visual Analysis of Network Traffic Using Knowledge Representation," in *Visual Analytics Science And Technology, 2006 IEEE Symposium On*, Baltimore, MD, 2006, pp. 107 – 114.