Sharing and Automation for
Privacy Preserving Attack Neutralization

(H2020 833418)

D6.2 SAPPAN Dashboard, final version (M33)

Published by the SAPPAN Consortium

Dissemination Level: Public

H2020-SU-ICT-2018-2020 – Cybersecurity

Document control page

Document file: Deliverable 6.2
Document version: 1
Document owner: Robert Rapp (USTUTT)

Work package: WP6
Task: T6.1
Deliverable type: DEM
Delivery month: M33
Document status: ☒ approved by the document owner for internal review
☒ approved for submission to the EC

Document History:

Version	Author(s)	Date	Summary of changes made
0.1	Robert Rapp (USTUTT)	2022-01-25	Initial version of the document
0.2	Robert Rapp (USTUTT)	2022-01-27	Revised document in order to internal review
0.3	Robert Rapp (USTUTT)	2022-01-29	Improvements based on internal review
1	Robert Rapp (USTUTT)	2022-01-31	Final version of the document

Internal review history:

Reviewed by	Date	Summary of comments
Franziska Becker (USTUTT)	2022-01-27	Suggestions for wording improvement and technical review
Avikarsha Mandal (FIT)	2022-01-28	Grammar and Spelling check, non-technical review
Alexey Kirichenko (FSC)	2022-01-30	Consistency and high-level technical review

Executive Summary

This final description of the SAPPAN dashboard reports a web-based dashboard application that serves as the integration point of visualisation-related tasks for analysts throughout the SAPPAN project. After a short review of the task T6.1 and its relation to the other SAPPAN tasks, the deliverable illustrates the dashboard from an security analyst's perspective to demo the general function of the dashboard. This is followed by the architecture of the whole dashboard application and ends with an overview of important implementation aspects. The latter include visualisation components and a notification and provenance service. The document concludes the work that has been carried out since the submission of the initial deliverable.

Table of Contents

1 Introduction 5

2 SAPPAN Context 6

3 Related work 7

4 Prototype 9

4.1 Description of the SAPPAN Dashboard 9

4.2 Analysts workflow 9

4.3 Implementation details 10

4.3.1 Project organisation 10

4.3.2 Persistence library 10

4.3.3 Elastic Search library 10

4.3.4 MISP library 11

4.3.5 Dashboard application 11

4.3.6 SignalR notification service 12

4.4 Exemplary visualisation components 13

4.4.1 Process tree visualisation 13

4.4.2 Playbook viewer 17

4.4.3 NetFlow visualisation 17

5 Evaluation 18

6 Summary 20

7 References 21

1 Introduction

This deliverable reports on the final result of Task 6.1 "Dashboard for response and recovery awareness". First, it describes the dashboard application, which includes prototypical visualisations, user management, a notification service, and analytic provenance. Based on the D2.3 "Visualisation requirements" we created a dashboard as a web-based application that enables access to the endpoint and network data through visualisations. This visual access to the data is of great value for security operation centres (SOC) to address their problem of having to handle tons of alerts each day. The advantage of visualisations is that a view can present collected information in a way that makes the analytical tasks of exploring, comparing and testing processes visually applicable. With the visualisations we add in the SAPPAN dashboard an alternative to viewing raw log files and data collected by security monitoring sensors.

In the following, we describe the role of the dashboard in the overall context of the SAPPAN project, illustrate the functionality of the process tree visualisation from the user's point of view and elaborate on important aspects of the system design and implementation.

2 SAPPAN Context

As described in the initial deliverable, the SAPPAN concept distinguishes the local response and recovery level, which refers to the data collection, threat detection and handling within a single organisation, from the global one, which is added by means of the sharing system. While WP3 and WP4 mostly deal with the local level, WP5 adds the global level and WP6 integrates all of the aforementioned results in a demonstrator for evaluation. The SAPPAN dashboard is a user interface for visualisations developed in the project and therefore needs to address both levels. As it depends on the results and data from the other work packages and therefore requires some level of integration, we decided that it should be a part of the overall demonstration and evaluation activities in WP6. The SAPPAN dashboard is not a research result of the project on its own, but it is the foundation and testbed for the visualisation-related research performed in the other work packages.

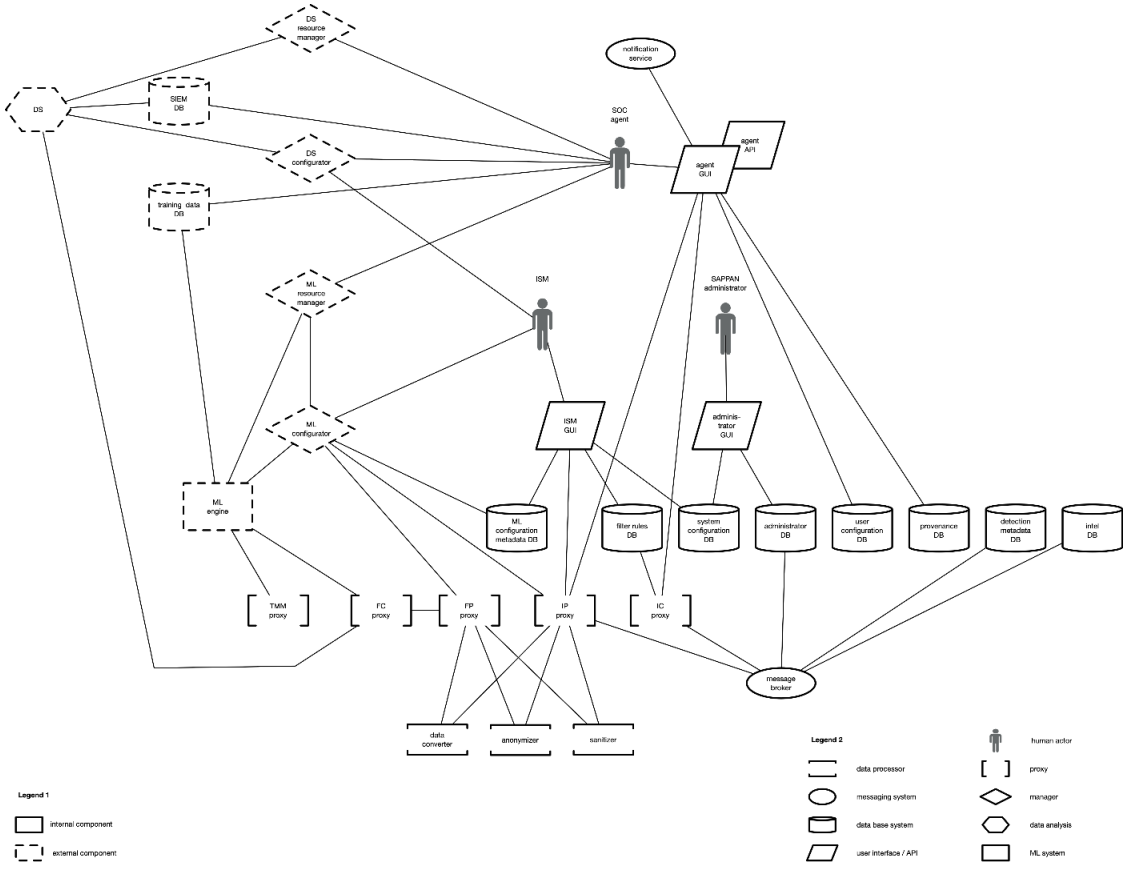

Figure 1: The SAPPAN system architecture.

In the system architecture shown in Figure 1, we deliberately chose to have three concepts for a user interface, the Agent GUI, the Information Security Manager (ISM) GUI and the Administrator GUI, without defining whether all of these roles would be fulfilled by a single interface or not. On a later point in time in D5.7, the MISP platform was chosen as the basis for the SAPPAN sharing system. The MISP platform has a user interface for managing the sharing platform itself, thereby fulfilling the functions of the ISM GUI and the administrator GUI. The SAPPAN dashboard developed in T6.1 represents only the Agent GUI, which is presented to the analysts in a security operations centre (SOC).

3 Related work

Our system is based on the idea of a dashboard [3], because this is a common user interface used by analysts to generate insights from incident-related information. For its use as an interactive tool for reporting in a management context [8], a widely used definition was proposed by Few as "a dashboard is a visual display of the most important information needed to achieve one or more objectives; consolidated and arranged on a single screen so the information can be monitored at a glance.", which he later clarified as dashboards being used to monitor whereas "faceted analytical displays" would "combine several charts [...] for the purpose of analysis" [2]. In our dashboard, the visualisations for analysis purposes are placed in cards that can be flexibly arranged and re-sized on a grid. By the development of interactive visualisations the SAPPAN dashboard comes with the advantages of a visual analytics [6] application that allows for interactive exploration of data and not just a pure display of the data. However, as the concept of dashboards is used in different domains, including software development [4], manufacturing [5], learning [6] and network security [7], and the technology to build them evolved, the term is also used for interfaces that enable interactive reasoning.

Process tree visualisation

For process trees, we developed a visualisation for analysts where they are able to view the temporal development next to the process hierarchies. While process trees and process activities are beneficial for exploring attacker operations like malware infiltration, there has been not much research on visualising them effectively. The time that a cyberattack detection software collected information may be a different time that an analyst looks at a specific alert in our dashboard, which differentiates our approach from approaches for dealing with real-time data like memory or CPU usage of endpoint processes. Tools like task managers of operating systems or more advanced process explorers like *Sysinternals Process Explorer* [9] from Microsoft or *htop* and *Process Hacker* are examples of process hierarchy representations. One notable difference between our data and the data that all of the aforementioned tools work with is that those tools always work with a snapshot of the processes currently running on the system. In contrast, our data are collected over a longer period of time, therefore including all processes that have ever been recursively spawned by the same system root process until a reboot. There are a number of established visualisation techniques that are specifically designed for hierarchical data such as node-link diagrams, indented lists, treemaps or matrix representations that are used to visualise them [4].

To view process hierarchies, an explicit representation like a node-link diagram can serve a clear view, but requires more screen space as a drawback. For two reasons, we decided to represent the hierarchical data in a left-aligned indented list. First, this representation is well known and used by all of the above-mentioned process explorers; second, it allocates space for process related details next to the hierarchy. The saved space can be used for the temporal process data next to it.

For hierarchically grouped elements involved in different transactions over time, Burch et al. [1] developed an approach to visualise sequences in so-called timeline trees. They used a tree layout for hierarchical information and a timeline to explore the events and analyse the temporal changes. Our approach shares a similar layout, but we use

the timeline to show process behaviour and the runtime of processes instead of mapping one property to it.

The approach of Trinius et al. [10] displays the behaviour of malware based on a treemap and a thread graph to show the frequency of system-level activities on threads. The thread graph shows a selected sequence of performed actions within a thread to support behaviour-based analysis similar to our approach. However, the visualisation of activities in threads has no hierarchical order like in process trees.

The approach by Hassan et al. [5] allows to view a dependency graph to show sockets, files and processes as nodes linked to a process. Based on this graph, analysts can see the flow of events including the execution of subprocesses and related activities to a source process.
Their approach aims to automatically decrease the number of alerts to reduce operators' workload. In contrast, we provide analysts with interactive visualisation to classify the behaviour on their own and to decide faster on an appropriate response. While the approach from Hassan targets the issues of *high false-positive rate* and *level of automation* mentioned in the survey paper from Kokulu et al. [7], we address the issues of *slow response speed* by displaying all process-related information at a glance in one visualisation.

4 Prototype

4.1 Description of the SAPPAN Dashboard

The SAPPAN dashboard is a visual interface for analyses of data in a SOC. Therefore, the primary function of the dashboard is to enable an initial analysis and lookup of detected alerts and incidents in the response process. After logging in, an analyst can view a detection result in the overview page and select an alert to start the investigation.

In its final version, the interface comes with a user management, a notification service, a flexible layout, visualisations, an analysis session and playbook viewer and backend sub-projects to connect to the SAPPAN sharing platform and data sources.

The initial deliverable contained the core concepts of the architecture, the requirements and – partly – implementation details of the layout and prototypical visualisations. In the following, we primarily describe the new contributions.

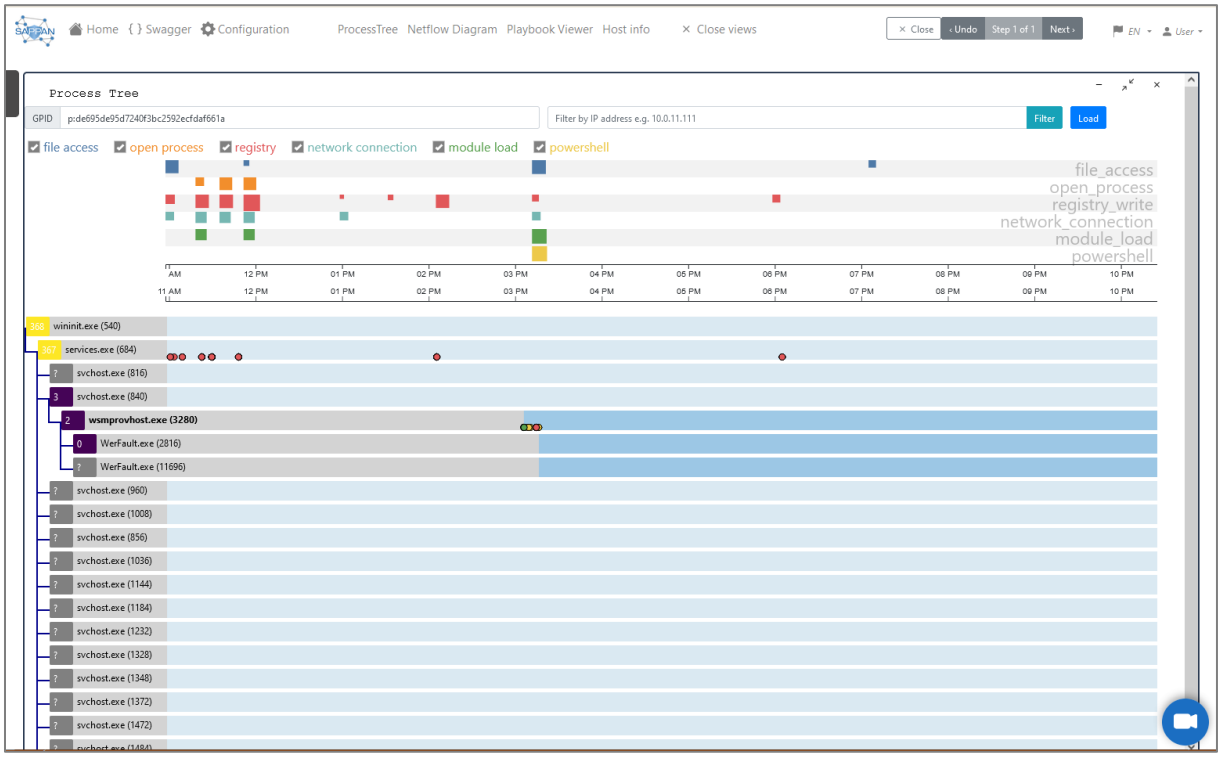

Figure 2: The SAPPAN dashboard displays a process tree visualisation in its card-based layout.

4.2 Analysts workflow

The analysis will begin in the process tree to access the process behaviour on an endpoint at the time of the alert. In the process tree, analysts can search for malicious process behaviour, like a suspicious network connection or unusual parent-child relations. With the use of the NetFlow histogram with seasonal trend decomposition, the analyst can analyse the packets or bytes transferred by the machine to identify peaks in the traffic. Based on the findings, the analyst can open the remediation tool in the overview menu. If the analyst has started a session recording via the provenance tool,

comments can be attached during analysis and different analysis steps can be viewed in the dashboard at a later point in time.

4.3 Implementation details

4.3.1 Project organisation

We split the dashboard project into several sub-projects, which are more or less self-contained libraries. A wrap up of the frameworks and libraries used for the backend and frontend functionalities is given in Figure 3. A detailed description is given in the initial deliverable D6.1.

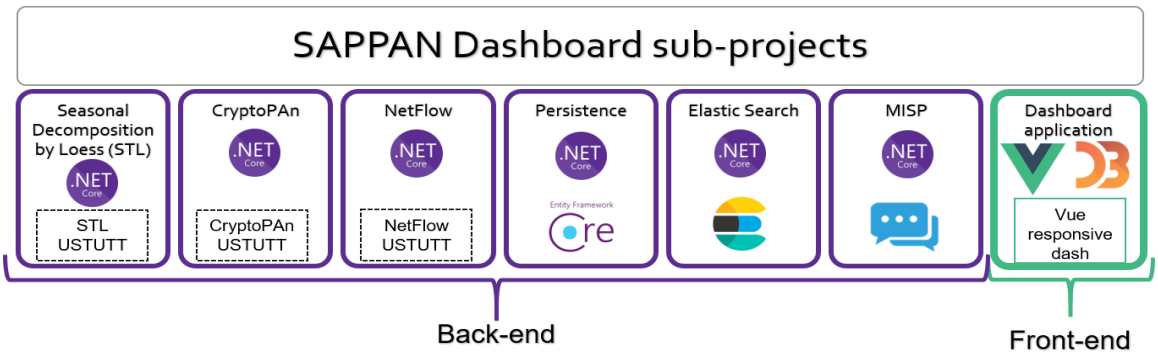

Figure 3: Sub-projects and used frameworks

4.3.2 Persistence library

The persistence library encapsulates the interface to a per-organisation relational database used to store the provenance and user identity information. We opted for a code-first approach based on the Entity Framework Core (EF Core), which defines the data model first and generates the database schema from the code and some annotations in it. The integration of the frontend's user login is based on this database to verify the user's access and provide a unique identifier of a user session in the frontend. This session identification provides analysts with the same layout status as when they closed the dashboard the last time. The SQL database schema is further described in D4.9 because one of the key concepts is the implemented command patterns that is used to persist analytic provenance steps as a sequence.

4.3.3 Elastic Search library

With the NEST (.NET client for Elastic Search)[1], an interface for the Web API of elastic search is provided. We created two minimal sets of common properties in the provided data in Elastic Search to parse JSON-based data models to the one we use in the dashboard. For the visualisations, we developed the sets that allow to parse the events generated by a host and NetFlows. The approach we use allows for quickly defining new data sources in the dashboard and having them automatically interpreted. To configure new data sources, the elastic search configuration file can be adapted to define different data sources in form of ES indices and define subtypes of data and their certain fields which then get processed. The mappings allow for a flexible configuration, by defining not only a single JSON path but an array, which allows for specifying alternate paths if the first one did not exist. This way, we prepared the dashboard for flexibly handling unexpected data.

[1] Elasticsearch .NET client: *[checked on 30.01.2022]*

4.3.4 MISP library

The MISP library provides a strongly-typed .NET Core implementation of MISP's core schema. The types are fully generated from the JSON Schema of MISP 2.4 and annotated so that their JSON serialisation meets the needs of MISP's Web API. In this library, a core schema is defined that enables access to the attributes of events shared over MISP. An event which value is a serialised SAPPAN playbook can be interpreted and pre-processed from the dashboard's backend for a visual representation of a playbook.

4.3.5 Dashboard application

The SAPPAN dashboard application is developed for experimental information visualisation. The user-interface design and in particular the layout assume that the dashboard displays just the relevant content in cards instead of a fixed allocated space per visualisation. Using the frontend framework vue.js, we built a frontend layout that can be dynamically filled by content from the data sources of the dashboard, as shown in the architecture diagram in Figure 1.

In the initial deliverable, we explained the layout grid and the flexible arrangement of the card-based functionality in detail. To create a dashboard without clutter, noise, and eye-catchers that could distract from analysis tasks, the layout has improved further, and we tried different ways.

As the dashboard will display sensitive data, we implemented the frontend functionality for login, logout and user management of Microsoft ASP.net core. To access the content, the analyst needs to log in first and is guided by an overview page shown in Figure 4. It allows for selecting between analysis, response or playbooks. The first displays an alert that can be directly opened in the process tree to start an investigation. The middle one is a link to an external response tool for remediation. The last one allows an analyst to view a visually represented incident response playbook. Analysts can always return to this point by the keyboard shortcut *shift+space.*

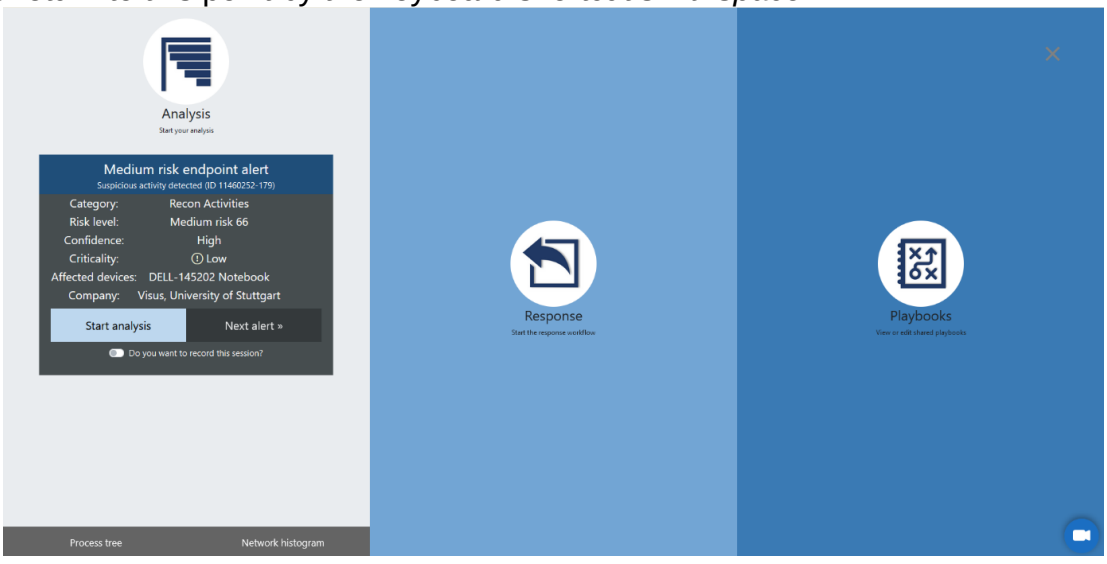

Figure 4: Entry point of the SAPPAN dashboard to guide the user to analysis, remediation or playbooks.

The initial layout was split into header navigation, analysis area to view different cards, and a views navigation and provenance bar on the bottom. We decided to remove the

bottom tools to allocate more space for the analysis area.

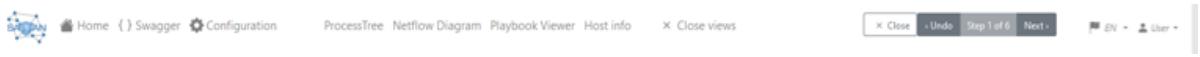

Figure 5: The new header links to development options, the visualisations, session player, language and user management.

The header area shows the logo, the home and development tools for data source configuration and API testing. This is followed by four different visual representations that were implemented and included in the dashboard. A card can be attached to the analysis area with these buttons, or all cards can be closed. A step navigation appears in the header area if a recorded analysis session is replayed by the provenance session player. The right part is for language settings and user options. The drop-down menus give access to the user management and show a selection for the language, which is not fully integrated because the content was prepared entirely in English.

To achieve this we included a menu (cf. Figure 6 on the right) that is just a small circle and can be clicked to access the provenance functionality recording or to select an analysis session. The selection of visualisations for the analysis tasks can be done via the header; between analysis sessions, a button to clear the dashboard can be used. The analyst can keep track of which analysis views are currently open by hovering over small pillows that are positioned on the left. In Figure 6, such a grey pillow can be seen on the left.

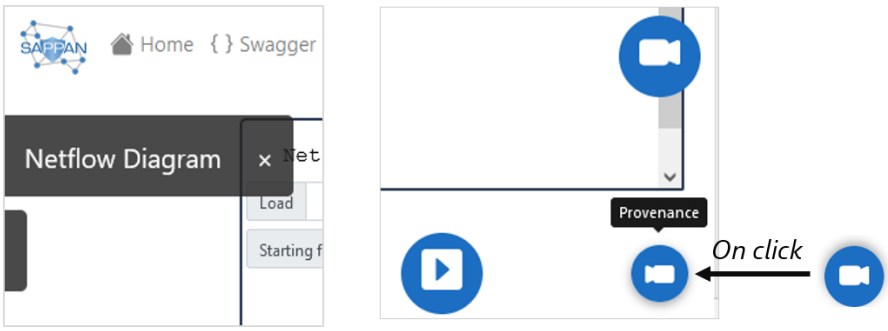

Figure 6: The current views are listed on the left and allow to maximise, minimise and close views. They show up when the mouse pointer hovers over the grey area. The provenance menu on the bottom right can be clicked to either start or view an analysis session.

The layout is handled over a JSON object. It has not changed since the initial deliverable, but we used it as well to persist the current arrangement of the views in the local storage of the browsers. When a user logs in or reloads the page, the latest layout can be loaded and the user does not need to open cards again.

4.3.6 SignalR notification service

We integrated a notification service that is envisaged in the SAPPAN architecture, which enables to push notifications to the users of the dashboard. In the current project setup, we have no notifications to show in the dashboard. For that reason, we imple-

mented the notification service on a prototypical level without a connection to an external source of notifications, such as an SIEM.

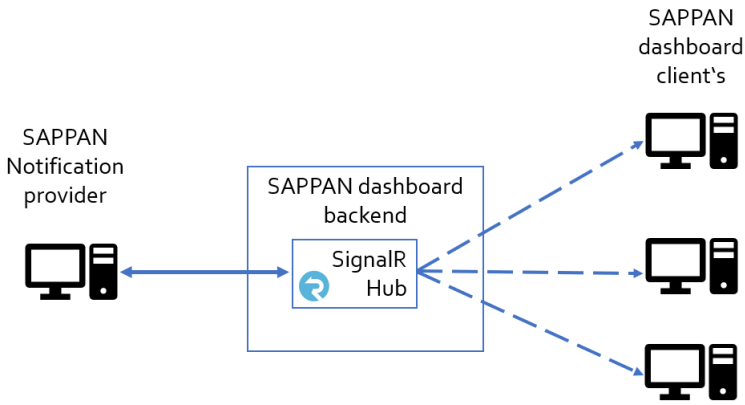

Figure 7: The abstract functionality of the SAPPAN notification service

The notification service works in a publish-subscribe manner, as Figure 7 shows. A notification provider can trigger notifications to inform all the dashboard clients at once. The notification provider can be an external system like a SIEM or a ticket system. SignalR creates a WebSocket for all clients to push notifications with almost no delay to the recipients.

4.4 Exemplary visualisation components

4.4.1 Process tree visualisation

In the first deliverable, the visualisation of the process tree showed the hierarchy as an indented tree plot, the temporal development, and the frequency of process activities in process bars. It also explained the computationally intensive process of reconstructing the hierarchy from the host data. The analyst could provide an ID to fetch a specific process tree.

A significant drawback of the initial visualisation was that processes that might be irrelevant were shown to the analyst and could not be removed. Also, the overdraw and lack of details of process activities, represented by the dots on a process life bar, avoided analysing the behaviour in detail. We address these issues by filtering, zooming, panning and layout improvements.

Filtering processes and activities

The current process tree initially begins with two input fields and six checkboxes. The first is still a mandatory field and defines which process tree will be fetched. The checkboxes enable the user to filter the displayed dots by a specific activity type. In Figure 8 no *file access* activities were shown on the life bars. By the frequency graph, an analyst still can estimate the presence and frequency of such events over the process tree.

We added a second input field to apply a filter for the y-axis where we can reduce the number of visible processes which might not be relevant for the analysis. To demonstrate the filtering mechanism, we implemented a filter by IP address. We filter the preprocessed data in the frontend to search for all processes that include a specific IP address within their network connections. If we display only the matching processes, the valuable information of parent-child relationships would be filtered out. The direct parents of the matching processes are added to the result set to avoid a loss of important information. As a suggestion for further improvements the process data could

be filtered in the same matter on multiple attributes to expand the analyst's possibilities collecting a subset of relevant processes.

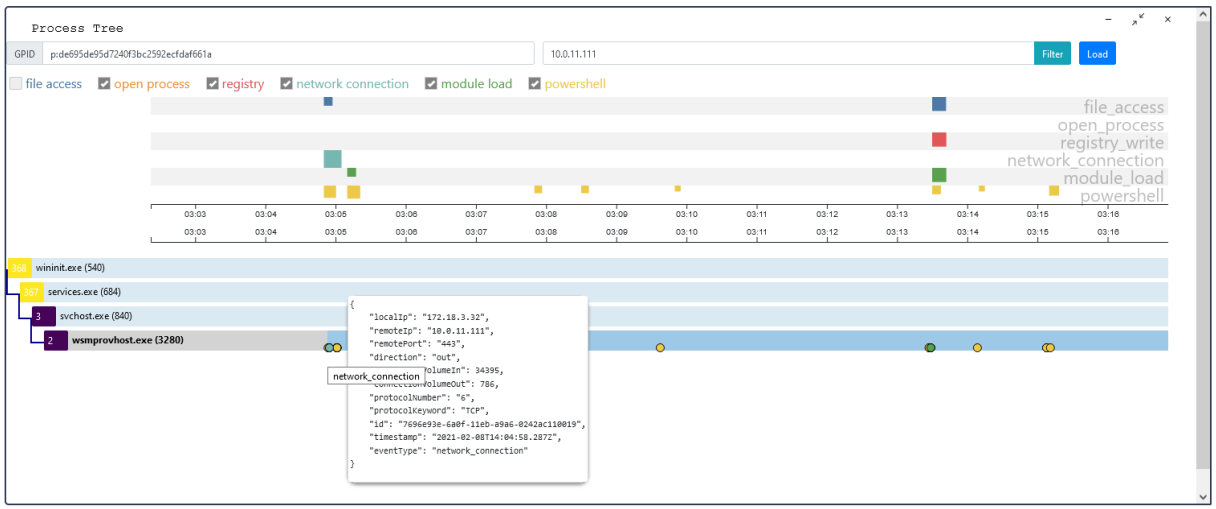

Figure 8: A filtered process tree shows only the matching process and the parents of the branch

Frequency graph

The frequency graph gives an overview of different process activities present in the process tree. In Figure 8, the graph shows each activity type as one line with rectangles distributed over the timeline. Each rectangle's size is represented on a logarithmic scale that is based on the total amount of activities per type. That representation gives the analyst an idea about the frequency of activities in the current timeframe. If the user moves the mouse over a rectangle, the number of events represented by this rectangle is shown. The timescale is coupled to the timeline of the process tree and can be rescaled by zooming in the process tree to view a specific timeframe. After the process tree is loaded the rectangle histogram summarises all events of the process tree, but an analyst can select a process to display the event frequency of a single process.

Enriched process details

A click on the process name opens an overlay with all process details for each shown process. The overlay has three different tabs and a coloured VirusTotal-indicator for a first estimation. The first shows the complete details about the process. The details can be used to inspect the process details or copy values for use in other SOC analysis tools. The second tab shows the full details of the VirusTotal API response. A request contains the sha1-value of a process and is triggered when an analyst opens the overlay. The VirusTotal-indicator summarises how many security providers have classified this value as suspicious. This demonstrates the possibility of information enrichment in such visualisation to avoid additional steps to external websites that an analyst might use. The third tab shows the activities in a collapsible list sorted by the type attribute. In **Figure 9**, is the full details of a network connection event are shown. After analysts viewed the temporal development in the process tree, this view becomes quite essential to view essential details for decision-making like the executed code of a PowerShell event.

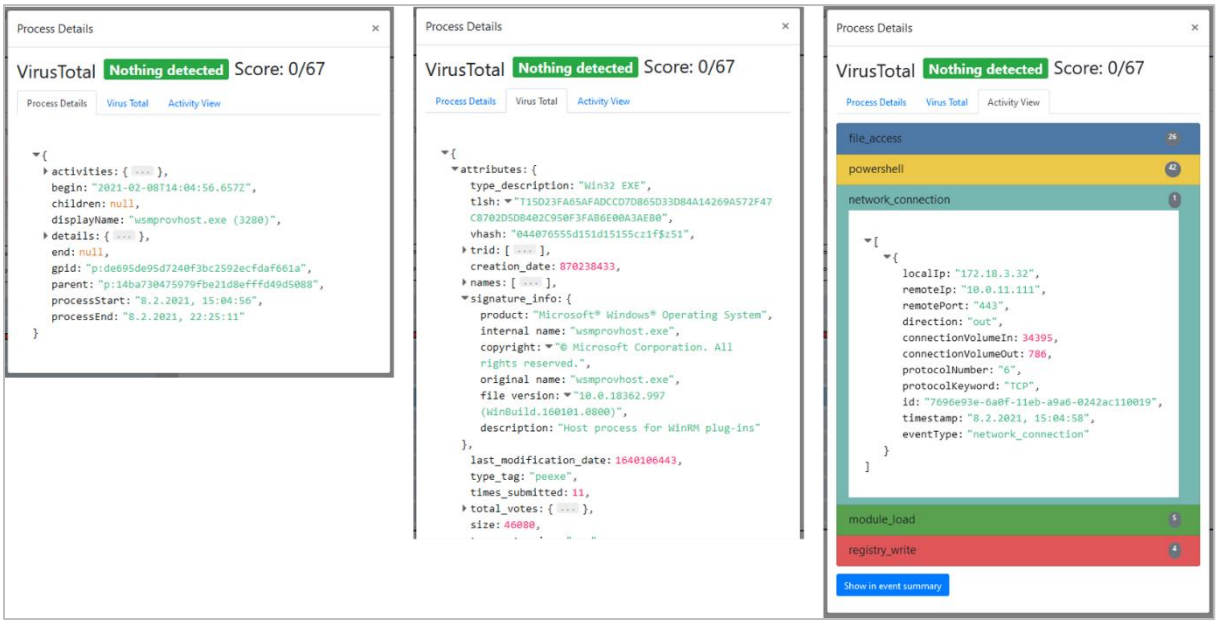

Figure 9: Process Details offer enriched information and access to all raw data for further investigation.

Layout improvements

In our earlier approach, we have started with the timeline on the same x-position as the hierarchy begun and worked with opacity to visualise. This allowed us to draw the hierarchy and runtime of processes more space efficient on top of each other, but it required more explanation for analysts. To face this, we defined a fixed area for the hierarchy before drawing the zoomable timeline area. By this change, we could apply a zoom and panning area just to the temporal data like the runtime and activities of a process and reduce misunderstanding. This implementation comes with another advantage. Since we used the runtime of the root process for the length of the time scale, the time axis sometimes showed a range of several weeks, which made it difficult to analyse a specific point in time. With the zooming implementation, we could set the initial timeframe to 10 hours to provide an overview of the process runtimes.

Special handling for event types

The process tree visualisation provides an ample space to explore the activities. Each activity can be handled special to shows details that belong to the activity type. We have implemented one of these handlings for process manipulations, i.e. events indicating that one process has opened another one via its process handle. By hovering over an *open_process* dot, the affected process is highlighted in red. Additional, a red arrow is drawn to the target process. These interactions do not need to be limited to the process tree visualisation and refer to other visualisations like host profiling e.g. to get further insights about network connections. An extensive tooltip is shown for other activities that let the analyst already get insight into the action performed.

Process activities comparison

While users can see the series of activities related to a process via the dot glyphs on the bars, it is hard to judge whether this time series is reasonable or might indicate a problem. Therefore, it is possible to compare this series of events using a specialised histogram view shown in Figure 10 that can be opened in a separate card on the dashboard. This view shows two overlayed histograms per type of event, one shows the

occurrences of events for the selected process, whereas the other displays the average of occurrences for all processes of the same process image. The end of the x-axis is similar to the current timescale of the process tree visualisation so that the analyst can pick an individual timeframe for the comparison. As the time on the x-axis is relative to the start of each individual process, these bar charts immediately show whether the selected process performed activities at similar points in its life cycle. While this might not immediately point to suspicious behaviour, particularly in the case of interactive applications whose activities are driven by user interaction, it still helps to identify spikes of activities as probably normal. A potential problem when using these histograms is that the number of activities can greatly vary over time and across processes. Therefore, a single malicious activity at an unusual point in time might become invisible as it gets mapped to a bar of almost height zero. To counter this, we let analysts switch between linear and logarithmic scaling. Furthermore, zooming on the time dimension addresses a similar problem with long-running processes like system services, for which the bars might become prohibitively thin.

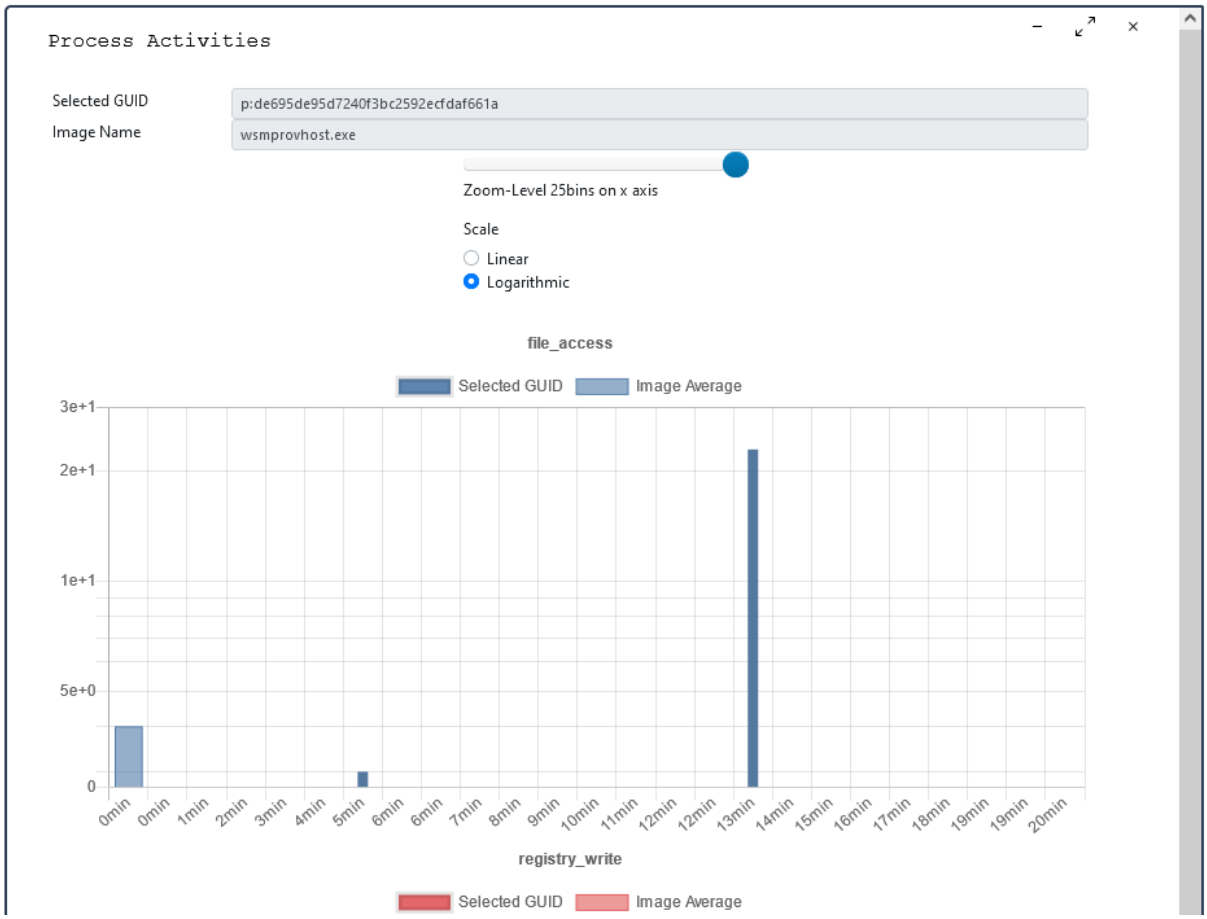

Figure 10: Analyst can compare the frequency of activities of a selected process and similar images.

4.4.2 Playbook viewer

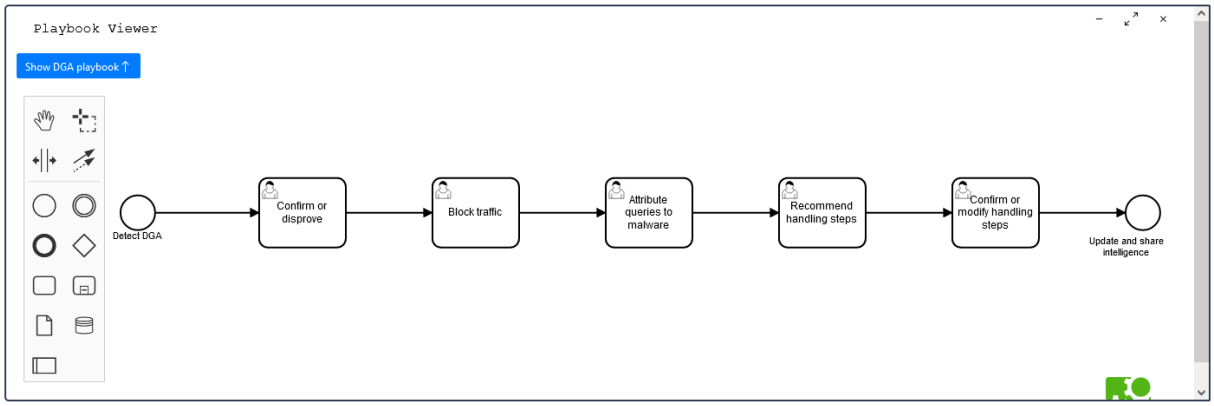

Figure 11: An exemplary playbook is translated from a JSON into BPMN shapes and edges to visualise the sequence of steps.

The dashboard has implemented a playbook viewer (cf. Figure 11) for the SAPPAN playbook standard which was developed in the D4.2. The viewer has an additional mapper to fetch a playbook in JSON format from MISP and is mapped to a graphical notation named business process model notation (BPMN). This notation was developed for analysts, technical developers and business managers. For this reason, we opted for BPMN a graphical representation of playbooks, as they should be easily communicated to colleagues and understood by the analysts without demanding advanced visualisation capabilities. The mapping for this visual representation was done before the consortium decided to change the playbook description to CACAO. The representation cannot import the current CACAO specification[2] because that would require the creation of a new mapping.

4.4.3 NetFlow visualisation

The NetFlow visualisation in Figure 12 shows the bytes or packets transferred in NetFlow data. The NetFlow data is pre-processed and an additional STL decomposition is performed by the NetFlow API controller before the data is delivered to the frontend. In the initial deliverable, this visualisation is explained in detail. However, it was slightly adapted to record the backend request for provenance reasons as well as handle a request by the session player to trigger a backend request with the parameters the user defined during recording.

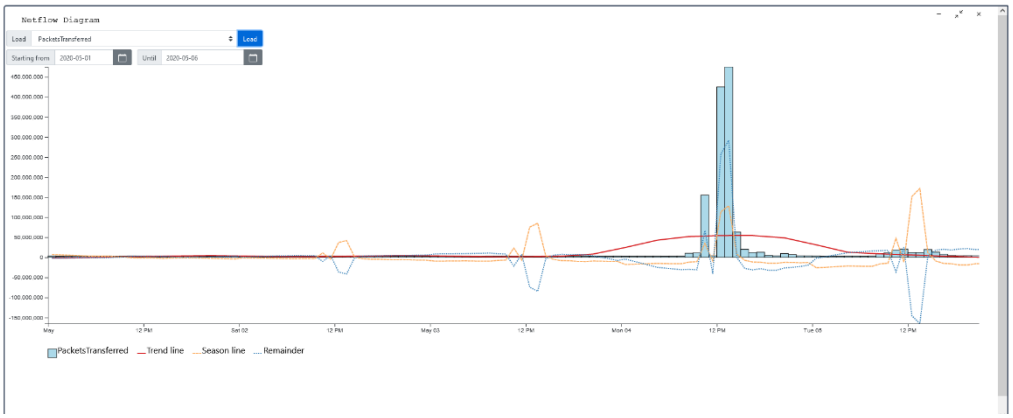

Figure 12: NetFlow Visualisation in the SAPPAN dashboard

[2] CACAO Specification: https://docs.oasis-open.org/cacao/security-playbooks/v1.0/cs02/security-playbooks-v1.0-cs02.html [checked on 30.01.2022]

5 Evaluation

The SAPPAN dashboard is not a research result of the project we evaluated. However, as a testbed for visualisation-related research, we evaluated the process tree as part of the dashboard. The process tree was evaluated as the primary visualisation to enable analysts to view endpoint sensor data in a few iterations. During development, we frequently discussed internally the visualisation approaches at VIS institute of University Stuttgart and additional collected expert feedback in peer-to-peer feedback sessions via video chat to adjust the approach. This was done to fit two purposes, first to ask an expert from security about important aspects of the visualisation for the ongoing development and to get feedback about the visual representation from cyber practitioners without a visualisation background. These peer-to-peer feedback sessions contained a demonstration of the SAPPAN dashboard within the visualisations, especially the process tree and discussing the current status.

The first session was done with a security expert in September 2021. At this point in time, initial versions of the visualisations process tree and NetFlow visualisation were available in the dashboard. Because the endpoint data in the session was locally collected and had no actual malicious processes in it, the general understanding of the process tree and the process activities could be discussed. The participant pointed out that analysts need to have access to all data on demand. However, child processes unrelated to a suspicious process lead to cluttering in the visualisation and can be removed. The participant's feedback also leads to the decision to work with an overlayed bar histogram, which can be seen in Figure 10. The use of additional tooltips that show more details on demand and apply filters to reduce overdraw of activities. Initial positive feedback regarding the process hierarchy and temporal development of processes was given here. The visual approach was easy to understand, but additional filters and details on demand are necessary to better the analyst's workflow.

The second session was done with two security experts, both with over ten years of experience in cyber defence, in November 2021. The process tree visualisation got additional tooltips and activities in it can be filtered by checkboxes, but the endpoint data was still without a malicious process and sanitised personal data. Additionally, the process information of a selected process was enriched by the VirusTotal API and in the process details. The visualisation of process activities was changed to a histogram with overlayed bars.

The session's feedback can be summarised into several statements where the participants pointed to. The participants liked to see the hierarchical dimension next to the temporal development and added several remarks during the discussions. The process tree should be able to filter on the y-axis to reduce the number of processes that are not relevant for the analysis and enable zooming on the x-axis to customise the visible timeframe. The approach to information enrichment is good and could be further developed. A further recommendation was to create a new dataset to serve the analyst a full context and the frequency of child processes based on the shown data. The sanitised data did not contain important data, like the path of a *file_access* activity, which would be needed for real analysis. However, this issue could be addressed afterwards with the SAPPAN: Advanced Threat Data dataset[3]. Regarding activities of

[3] SAPPAN: Advanced Threat Data dataset: https://zenodo.org/record/5547862 [checked on 30.01.2022]

processes, the participants recommended that the single dots not only represent the position on a timeline and the type but also lead to a detailed view.

The final evaluation sessions took place in January 2022. Within three video calls the feedback of four security analysts was collected. The sessions last around one hour and in the second session participated two analysts from the same company. During the sessions the participants were briefly asked about their expertise, role and familiarity with visualisations. The experiment leader continued with an introduction to a simple demo scenario based on the SAPPAN: Advanced Threat Data we developed in WP 3. Then the participants analysed together with us the suspicious process. We could operate the actions on the local instance of the dashboard on a shared screen and the participants could interpret the shown information and think aloud about actions and insights. An interview followed the analysis to fill in a questionnaire that collected the system usability scale (SUS) and the use for cyber security analysis tasks. The session ended with additional remarks from the participants.

In view of the fact that the number of participants is not necessarily suitable for such a quantitative measurement and the participants are more inclined to give positive feedback in order to fulfil the researcher's hypotheses, the prepared questionnaire is analysed below for the SUS and the performance questions.

Used scale	1	2	3	4	5
	strongly disagree	disagree	neutral	agree	strongly agree

System usability scale	Session 1		Session 2	Session 3	Average
I think that I would like to use this visualisation frequently.	4	4	4	5	**4,25**
I found this visualisation unnecessarily complex.	2	2	2	2	**2**
I thought this visualisation was easy to use.	4	4	4	4	**4**
I think that I would need the support of a technical person to be able to use this visualisation.	1	1	1	1	**1**
I found the various functions in this visualisation were well integrated.	4	4	4	4	**4**
I thought there was too much inconsistency in this visualisation.	1	1	1	1	**1**
I would imagine that most people would learn to use this visualisation very quickly.	4	4	4	4	**4**
I found this visualisation very cumbersome to use.	2	2	2	3	**2,25**
I felt very confident using this visualisation.	4	4	4	4	**4**
I needed to learn a lot of things before I could get going with this visualisation.	1	1	1	1	**1**

Performance	Session 1		Session 2	Session 3	Average
The visualisation enables viewing the temporal development of processes?	5	5	4	5	**4,75**
The visualisation enables inspecting the frequency of activities of processes?	4	4	4	4	**4**
The visualisation shows relations between temporal and hierarchical information?	4	4	5	4	**4,25**
The visualisation shows enough details on demand to make conclusions.	3	3	5	4	**3,75**
The visualisation enables viewing malicious behaviour of processes.	4	4	4	5	**4,25**

6 Summary

In the final version of the SAPPAN dashboard, we demonstrated the progress of the SAPPAN dashboard since the initial deliverable. This deliverable focuses primarily on the prototypical visualisations and improvements of the frontend design. The dashboard is an interface for SOC analysts to visually access host and NetFlow data and allows analysts to combine different views on the data. With the evaluation of the prototypical implementation, we could show that we meet the visual requirements of analysts by easily understandable visualisations and could identify issues in the reconstruction of hierarchical structures that need to be addressed for productive implementation.

7 References

[1] Michael Burch, Fabian Beck, and Stephan Diehl. 2008. Timeline trees. In *Proceedings of the working conference on Advanced visual interfaces*. ACM, New York, NY, 75. DOI: https://doi.org/10.1145/1385569.1385584.

[2] Stephen Few. 2009. *Now you see it*. *Simple visualisation techniques for quantitative analysis*. Analytics Press, Oakland, Calif.

[3] Stephen Few. 2013. *Information dashboard design*. *Displaying data for at-a-glance monitoring* (2 ed.). Analytics Press, Burlingame, Calif.

[4] Martin Graham and Jessie Kennedy. 2010. A Survey of Multiple Tree Visualisation. *Information Visualisation* 9, 4, 235–252. DOI: https://doi.org/10.1057/ivs.2009.29.

[5] Wajih U. Hassan, Shengjian Guo, Ding Li, Zhengzhang Chen, Kangkook Jee, Zhichun Li, and Adam Bates. 2019. NoDoze: Combatting Threat Alert Fatigue with Automated Provenance Triage. In *Proceedings 2019 Network and Distributed System Security Symposium*. Internet Society, Reston, VA. DOI: https://doi.org/10.14722/ndss.2019.23349.

[6] 2005. *Illuminating the path*. *The research and development agenda for visual analytics*. IEEE Computer Soc, Los Alamitos, Calif.

[7] Faris B. Kokulu, Ananta Soneji, Tiffany Bao, Yan Shoshitaishvili, Ziming Zhao, Adam Doupé, and Gail-Joon Ahn. 11062019. Matched and Mismatched SOCs. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. ACM, New York, NY, USA, 1955–1970. DOI: https://doi.org/10.1145/3319535.3354239.

[8] Shadan Malik. 2005. *Enterprise dashboards*. *Design and best practices for IT*. Wiley, Hoboken, NJ.

[9] Mark Russinovich. 2021. *Process Explorer - Windows Sysinternals* (June 2021). Retrieved June 9, 2021 from https://docs.microsoft.com/de-de/sysinternals/downloads/process-explorer.

[10] Philipp Trinius, Thorsten Holz, Jan Gobel, and Felix C. Freiling. 102009. Visual analysis of malware behavior using treemaps and thread graphs. In *2009 6th International Workshop on Visualisation for Cyber Security*. IEEE, 33–38. DOI: https://doi.org/10.1109/VIZSEC.2009.5375540.