Sharing and Automation for
Privacy Preserving Attack Neutralization

(H2020 833418)

D4.2 Vocabulary for incident data and response and recovery actions (M21)

Published by the SAPPAN Consortium

Dissemination Level: Public

H2020-SU-ICT-2018-2020 – Cybersecurity

Document control page

Document file: D4.2 Vocabulary for incident data and response and recovery actions
Document version: 1.0
Document owner: Lasse Nitz (FIT), Mehdi Akbari Gurabi (FIT)

Work package: WP4
Task: T4.1 Develop a methodology for formalising and modelling response and recovery actions and their triggers
Deliverable type: Report
Delivery month: M21
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	Lasse Nitz (FIT), Mehdi Akbari Gurabi (FIT)	2021-01-11	Document outline
0.2	Lasse Nitz (FIT), Mehdi Akbari Gurabi (FIT)	2021-01-24	First draft
0.3	Lasse Nitz (FIT), Mehdi Akbari Gurabi (FIT)	2021-01-27	Ready-to-review version
0.4	Lasse Nitz (FIT), Mehdi Akbari Gurabi (FIT)	2021-01-28	Incorporated feedback
1.0	Lasse Nitz (FIT), Mehdi Akbari Gurabi (FIT)	2021-01-29	Final version

Internal review history:

Reviewed by	Date	Summary of comments
Mischa Obrecht (DL)	2021-01-28	Grammar/spelling, content
Tomas Plesnik (MU)	2021-01-28	Grammar, content
Benjamin Heitmann (FIT)	2021-01-29	Review on standardisation and risk mitigation context

Executive Summary

This deliverable is part of task T4.1, and builds on the results of deliverables D4.1 (Formal Methodology for Modelling of Response and Recovery Actions and their Triggers) and D4.3 (Approach for Capturing Incident Response and Recovery Steps). The core part of this deliverable is the definition of a vocabulary to document response and recovery (R&R) steps in form of machine-readable playbooks, based on the playbook methodology defined in D4.1. Due to this connection, D4.1 is revisited in this deliverable to summarize the most important aspects of the formal methodology on an intuitive level, and to document the decision of how open trade-offs have been settled. Additionally, the mitigation measures taken to address the potential risk surrounding the sharing of playbook examples within the consortium are discussed. Further, updates to the knowledge capturing tool from D4.3 are documented. Additionally to the definition of a response and recovery vocabulary, initial ideas for adding automation support to the playbooks are presented. These ideas include the approach to link SAPPAN playbooks to MISP incident reports by defining conditions as logical statements over incident data. Lastly, plans for standardization are discussed.

Table of Contents

1 Introduction

Within deliverable D4.1, a formal methodology to model response and recovery actions as cybersecurity playbooks has been defined. The result is the definition of a generic playbook structure that utilizes semantic technologies. As part of this deliverable, the work done for D4.1 is extended by defining an incident response and recovery vocabulary, which allows to attach response and recovery actions to the semantic playbooks. The knowledge capturing tool presented in D4.3 has also been updated. Since the SAPPAN playbook structure defined in T4.1 serves as basis for tasks T4.3, T4.4 and T5.4, changes to the proposed vocabulary will potentially be made within respective deliverables, in order to match new insights gained within these tasks. This specifically is considered for task T4.4, which deals with the automation of response and recovery steps. As the vocabulary presented in this deliverable is supposed to be primarily understood by human operators, it might be necessary to extend the vocabulary for automation purposes. This, however, has already been considered on a conceptional level.

Regarding the structure of this deliverable, the most important points of D4.1 will be briefly recalled. This includes a summary of the semantic technologies used and an intuitive description of the playbook structure. Then, the open trade-offs documented in D4.1 are revisited and decided. This is followed by a description of the mitigation measures taken to tackle the identified potential risk of not being able to gather sufficiently many examples of cybersecurity playbooks within the SAPPAN consortium. After this, updates to the proof-of-concept implementation of the knowledge capturing tool presented in D4.3 are documented. Then, the importance of a common understanding is discussed in regards to the sharing of playbooks, followed by a brief overview of existing cybersecurity vocabularies and glossaries. After this, the SAPPAN vocabulary for incident response and recovery steps is presented. Next, initial ideas for adding automation support to SAPPAN playbooks are documented. Further, plans for standardization efforts for playbooks are presented, and lastly, the results of interviews with domain experts are briefly discussed.

2 Revisiting D4.1

Since the content of D4.1 is relevant in context of this deliverable, it is very briefly presented in the following on an intuitive level. For a more formal description of, e.g., the generic playbook structure, please be referred to deliverable D4.1. After the short recap of the most important points of D4.1, the open trade-offs documented in D4.1 will be discussed and decided, followed by a documentation of how the SAPPAN consortium has dealt with the identified risk regarding confidentiality concerns surrounding the sharing playbooks within the consortium.

2.1 Intuitive description of the generic playbook structure

In deliverable D4.1 (Formal Methodology for Modelling Response and Recovery Actions and their Triggers), a generic playbook structure has been defined using semantic

technologies. As the underlying technology, the Resource Description Framework (RDF) [1] has been chosen, as it is a standardized recommendation by the W3C. It follows an intuitive *(subject, predicate, object)* description of relationships between resources (the *subject* stands in relationship *predicate* with the *object*) and serves as basis for many other semantic technologies and standards, such as the RDF-query language SPARQL [2] and the semantic RDF-extensions RDFS [3] and OWL 2 [4]. Further, knowledge expressed in RDF can be serialized using popular formats such as JSON and XML.

The benefit of using RDF as the underlying data model is its graph-like structure of expressing information as *(subject, predicate, object)* triples. The *subject* and *object* can be seen as nodes, which are connected via an edge labelled with the *predicate*. Since this comes very close to popular visualizations of playbooks as graphs, the data model naturally fits this purpose. Another benefit is that RDF is independent of any visualization built on top of playbooks described with it: Sharing an RDF serialization allows to visualize it differently in each receiving organization, depending on which visualizations are used. There just needs to be a mapping from the SAPPAN playbook components to the visual components of the intended visualization. Then it is possible to express SAPPAN playbooks using different process visualizations, such as the Business Process Model and Notification (BPMN) [5].

Intuitively, a playbook (as defined in D4.1) can be seen as a sequence of steps that fulfils the following:

- Each playbook has
  - a clearly defined starting point, and
  - a clearly defined end point
- Between the starting point and the end point, the response and recovery process is described by intermediate steps in a clearly defined order
- For each intermediate step, it has to hold that
  - it is reachable from the starting point, and that
  - the end point is reachable from it
- The sequence of steps (flow) can be split via
  - exclusive branching (only a single branch is supposed to be followed)
  - parallel branching (the flow is split into several sub-flows that need to be carried out, but can be parallelized)
- Several sub-flows can be merged into a single flow again
- Optional steps, which are not part of the main workflow and do not necessarily need to be carried out, can be attached to intermediate steps
- Indicators can be included within the flow described by the playbook, e.g., to indicate when the application of a response and recovery action has been successful
- Indicators can also be directly assigned to the playbook (not part of the sequence of steps), e.g., to indicate when a playbook is suitable for a certain situation

The respective RDF resources have been defined by using semantics of OWL 2. For the specific definition of resources and an explanation of the notation used for RDF-related aspects in this deliverable, please be referred to D4.1.

2.2 Addressing open trade-offs documented in D4.1

As part of D4.1, several trade-offs have been presented, but not ultimately decided. This was mainly due to the early stage in which related WP4 tasks have been in at the point of writing. Since, however, progress has been made towards the other tasks, it is time to revisit these open trade-offs and to document how they have been decided.

2.2.1 Reasoning vs. application logic

In this context, reasoning refers to the process of gaining explicit knowledge about things that are implied by the semantics of the used resources. An intuitive way to describe this process is as follows:

Assume that we have a relationship "isAncestorOf" and a description consisting only of these three triples:

- (Alice, isAncestorOf, Bob)
- (Bob, isAncestorOf, Carol)
- (isAncestorOf, rdf:type, owl:TransitiveProperty)

The first triple defines that Alice is an ancestor of Bob, the second triple that Bob is an ancestor of Carol. The third triple utilizes OWL 2 semantics to define "isAncestorOf" as a transitive property. According to the semantics of transitivity, it also has to hold that (Alice, isAncestorOf, Carol). Reasoning is the process of discovering such implied knowledge, which (after discovery) can be explicitly included in the knowledge base. The main problem, however, is that this discovery process can be highly complex, depending on the set of semantics that is used. The complete set of resources with fixed semantics defined by OWL 2 is even undecidable.

For the purpose of modelling playbooks, we have decided to step away from using reasoning extensively. As discovery of implicit knowledge is primarily interesting in scenarios dealing with large, convoluted collections of data, it does not exactly serve the purpose of modelling response and recovery processes, which (in contrast) deals with very well structured workflows.

As a natural consequence of this, we do not have any playbook-related use case in SAPPAN, which requires a broad use of reasoning. Hence, this trade-off has been decided in favour of the use of application logic, as this allows to enforce constraints regarding the knowledge that is modelled. It also allows to model playbooks without in-depth knowledge about the used semantics to avoid unintended consequences resulting from reasoning. The decision towards application logic thus gives the users more intuitive control about what is modelled and reduces the overhead the user has to tackle when adapting to the playbook structure proposed in SAPPAN.

This, however, does not render the use of semantic technologies useless, as application logic can be implemented using SPARQL queries, and it also allows for the potential integration of playbooks into large cybersecurity knowledge bases.

2.2.2 Reasoning after updates vs. reasoning during queries

As already mentioned above, reasoning does not play a crucial role for response and recovery playbooks. The formal methodology defined in D4.1, however, does support the option to use reasoning, e.g., to automatically discover triples that define reachability relations. As reachability is considered to be transitive in this context, the automated generation of such a relation via reasoning is possible based on the semantic description of a playbook.

There are different ways to trigger reasoning. On the one hand, reasoning could be triggered when respective queries (e.g., via SPARQL) are run, to mine implicit knowledge that is relevant for the query. On the other hand, reasoning could be applied whenever the knowledge base (in our case, the semantic description of a playbook) is updated, either by adding or by removing triples. Reasoning during queries is advised in scenarios, in which the knowledge base is often updated, but only relatively rarely queried with queries that are not time-critical. Reasoning after updates, on the other hand, allows for more performant queries (as no reasoning needs to be applied), and comes with the benefit that the knowledge base is always up-to-date.

Since playbooks are very limited in size (compared to other knowledge bases) and are not expected to be changed at high frequency, the option to apply reasoning after updates is advised. This also ensures that relevant implied knowledge is available for any application built on top of the semantic playbook description, even if a non-semantic application wants to access the serialized description. Due to the relatively small size of playbooks, the respective increase in size caused by reasoning does not have a severely negative impact. Thus, the use of reasoning after updates is recommended, if reasoning is used by an application built on top of the semantic playbook description.

2.2.3 Expressiveness vs. complexity

The trade-off between expressiveness and complexity refers to the use of resources with fixed semantics. As already mentioned above, the whole set of semantic resources defined by OWL 2 is undecidable, and different subsets of resources have different complexities. In this context, complexity refers to the complexity of reasoning and queries. As described above, we have decided to not significantly rely on the use of reasoning, which hence will not be explicitly considered further. For queries (e.g., via SPARQL), on the other hand, we have not run into a situation yet, in which the runtime of queries on RDF playbooks becomes notable. This also results from the relatively small size of playbooks, as compared to large knowledge bases. Hence, this trade-off has been decided in favour of expressiveness.

The semantic connections of SAPPAN resources as defined in D4.1 are kept, as this conceptually allows to integrate these playbooks into a larger cybersecurity knowledge base. Even if these knowledge bases utilize a different vocabulary, semantic playbooks can be integrated by defining connections between the playbook and the knowledge base vocabularies via OWL 2 resources, such as *owl:differentFrom* and *owl:sameAs*. Since we cannot make any reasonable assumption on the subset of semantic resources that is already used by such a knowledge base, we have not reduced the set of OWL 2 resources. Consequently, the set of semantic resources used in D4.1 has not been reduced.

2.3 Mitigation of the risk surrounding the collection of playbook examples

As part of D4.1, the risk of not being able to get enough samples for playbooks has been documented. This problem was the result of confidentiality concerns, since playbooks describe actions taken to mitigate real-world threats and potentially also allow to draw conclusions about the internal technical infrastructure of an organization. Sharing playbooks that are in active use hence does not only concern privacy, but in fact also security of the respective organizations.

This problem has been mitigated as follows:

- Selected playbooks have been shared by some partners.
- Some partners have created playbook examples just for this purpose. These playbooks are not in use by their SOC and do not necessarily reflect how the organizations react to respective attacks. Sharing it in the consortium is consequently possible. For the purpose of tasks T4.1 and T4.2, it is not relevant that the specific actions might differ from the ones in the playbooks that are actually used by the respective SOCs.
- Suitable open-source playbooks (e.g., [6] [7]**)** have been identified and proposed as good examples by various partners.
- Interviews and discussions with SOC members have been organized to get a better understanding of how the target group uses playbooks and where potential pain points of the current state are.

Since the problem of security and privacy concerns also extends to the research plans of the project to make playbooks shareable via the SAPPAN platform, the vocabulary proposed in this deliverable has been designed in a hierarchical fashion, based on the awareness created by resolving this issue. This allows to describe the response and recovery process in various levels of details, such that the playbook is still useful, even if a complete level of detail is removed from the playbook before sharing. These different levels can, for example, make the difference between infrastructure-specific and infrastructure-independent workflows for response and recovery. Additionally, our playbook methodology allows to assign a confidentiality level via the Traffic Light Protocol [8] to each individual piece of information (more specifically, each resource) in the playbook. Due to the machine-readability of the underlying data model, this allows to efficiently, in an automated fashion, remove all information from a playbook that the recipient of the sharing process is not supposed to see.

3 Updates on knowledge capturing tool

After proposing a formal methodology for modelling of response and recovery actions, a proof of concept has been developed to check the feasibility and suitability of the developed model. Based on the feedback from domain experts in the interview sessions, the vocabulary is extended, and the capturing tool has been updated to cover the main requirements. The content of interview sessions is documented and available inside the consortium Confluence space. The full document cannot be published and discussed in a public deliverable due to the confidentiality level of shared information.

However, the main points and discussions are documented in Deliverable D4.3. We cover the domain expert suggestions and feedback regarding response and recovery steps and actions. Besides, ideas on playbook triggers are considered. However, the suggestion on implementation of the connectors to specific incident reporting tools, e.g., ticketing systems, are not in the scope of the current deliverable. It may be considered with regards to the integration of the prototype into the SAPPAN dashboard. Main feedback points and the actions to address them are as follows:

Main feedback	**Response to feedback**
Consideration on loops and parallel steps in the capturing tool	Applied
Initial, expiration, and duration time for steps	Applied to the actions connected to the steps
Steps could belong to multiple playbooks	Had been already considered
Playbook branching	Had been already considered (metadata for playbook's main focus and versions are considered in the current vocabulary)
Presenting all steps of a playbook in one spot	Applied with playbook graph and table of steps
Playbook information separate from specific incidents	Considered by current vocabulary
Integrate with case-management and incident management workflow	Considered via current vocabulary
A federated solution to protect organisations' confidential data	Each organisation has its own SMW instance
Risks or possible collateral effects, vocabulary for risk assessment	Considered in the vocabulary
Share playbooks as workflows	Considered via export component
Triggering alerts	Considered in the vocabulary
Flexibility with custom queries	Had been already considered
Categories for intermediate steps	Considered via action categories in the vocabulary
Implementation of the connectors to specific incident reporting tools	Not in the scope of T4.1 and T4.2

We decided to choose Semantic MediaWiki (SMW) for rapid prototyping of a capturing tool considering the benefits of the semantic technologies for a knowledge base as it is proposed in deliverable D4.3. For this purpose, we have developed a preliminary domain vocabulary based on the formal methodology proposed in D4.1. The domain model has been translated and modelled into SMW forms, templates, categories and properties. Also, the possibility to re-use existing domain taxonomies (such as the ones

defined by MISP) in the proof-of-concept prototype has been considered. Therefore, data can be inserted directly into SMW via the web interface or imported via an import component. Figure 1 shows the conceptual view of semantic knowledge modelling for the capturing tool.

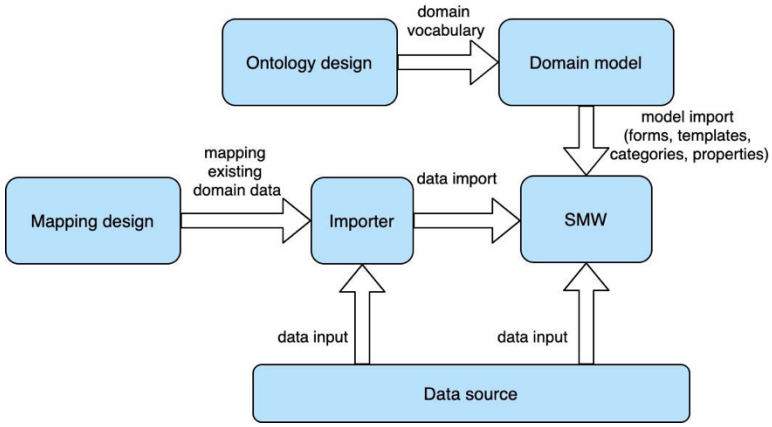

Figure 1: Conceptual view on semantic knowledge modelling

The SMW has a web interface based on MediaWiki which is connected to the SMW core component to utilize semantic technologies on a wiki knowledge base. It has an API which allows import/export in JSON, XML and other formats. Besides, an RDF/SPARQL backend can be used for advanced queries on the data. We use a dockerised version of SMW for easy deployment. Figure 2 displays the architecture of the capturing tool based on SMW. Given the sensitive nature of this information, a federated solution is considered where every organisation has its own instance and decides what to share through the SAPPAN sharing component using exported outputs. In this case, there is no need to trust a central administrative entity.

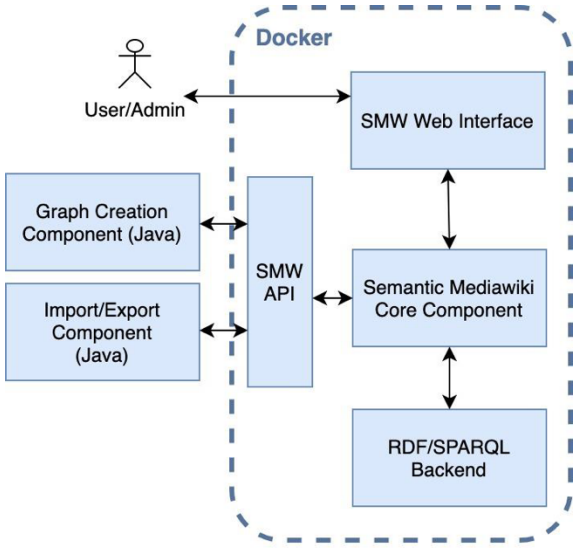

Figure 2: Architectural view on the capturing tool

3.1 User actions

In the SAPPAN capturing tool, a user can apply three different types of action: creation, search and fetching information. With the creation action, a user can add a playbook, add or edit pages, properties and resource values connected to a playbook or its steps. Also, import of XML, JSON and RDF/XML (an RDF serialization format) files is possible. Search actions include wiki searches for pages and semantic queries via SPARQL. Information fetching actions are divided into four main categories: First, getting a specific playbook, indicator or any other item of captured data. Second, viewing the graph representation of a playbook. Third, getting contact info of a reporter, corresponding role or responsible person. And forth, export playbooks in XML, JSON, or RDF/XML serialisation format to reuse data in other tools. The categories of actions are shown in Figure 3.

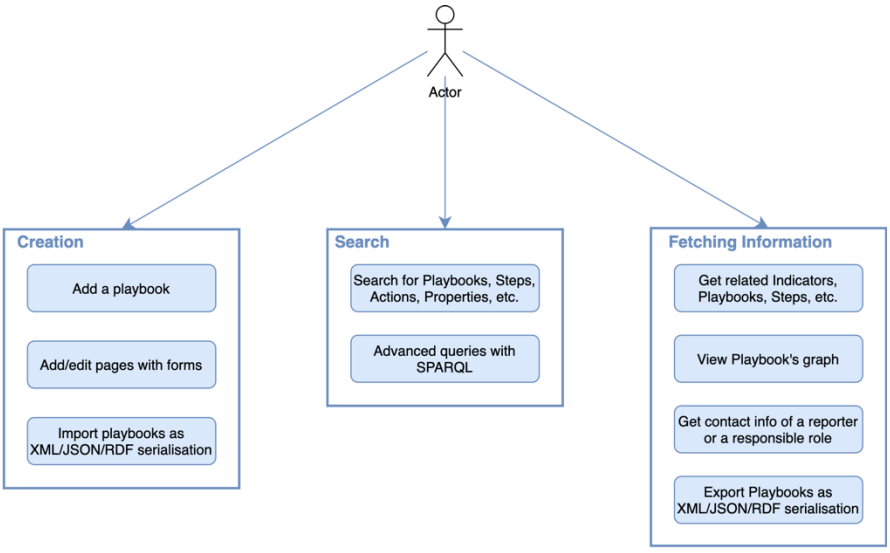

Figure 3: Actions diagram of capturing tool

3.2 Capturing a playbook

Creation of a playbook is possible via a respective form in the SAPPAN capturing tool after a successful login to the system. Playbook creation includes adding general information to the playbook and setting up the steps. The playbook author should connect steps to generate a workflow sequence. The corresponding resources to each step (e.g., an action) can be set via another form. If the authors need to introduce new properties to capture playbooks, they can create them via the corresponding form. Also, the confidentiality level of a playbook and each resource are defined. For a non-public playbook, a sharable version of it can be created based on the confidentiality level of the resources. Aspects surrounding the privacy and approaches for masking or removing confidential data will be considered in more detail in T5.4. Finally, the graph representation of the playbook based on the JSON output will be created and embedded in the playbook's wiki page. Figure 4 represents the workflow of the creation of a playbook via the SAPPAN capturing tool.

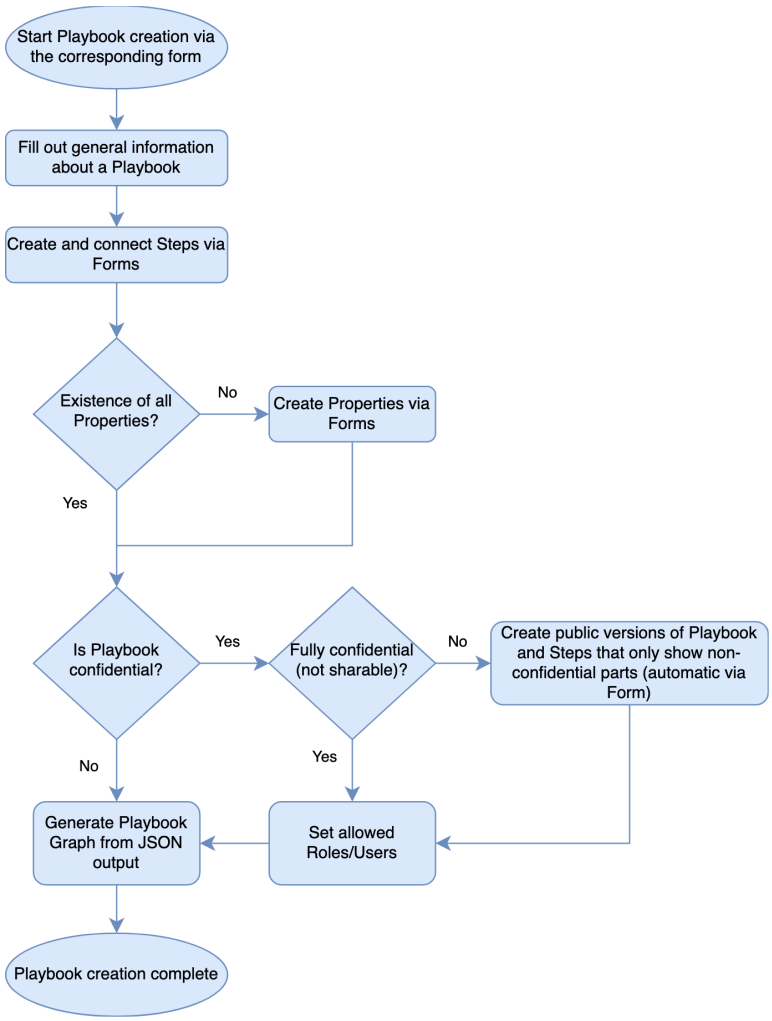

Figure 4: Workflow diagram for the creation of a new Playbook

3.3 Graph representation of a playbook

Here, we have a preliminary playbook for email phishing for a human handler as an example:

1. The handler sets the category of a phishing monitoring tool to Email phishing.
2. The handler runs a phishing monitoring tool according to its manual. (Can log into mail server)
3. The handler interprets the results of the phishing monitoring tool in case of phishing.
4. If the blocker runs successfully (Optional: add the address to a block list), sending e-mails to the reply-to address will be blocked. Then sends warning to each victim.
5. The handler verifies if the email address is blocked. If not, he blocks it manually and reports a bug to the administrator. Sending e-mails to the reply-to address will be blocked. Then, sends warning to each victim.
6. Receive victims' confirmation.
7. The handler closes the ticket as solved.

These steps are stored in a structured way in the capturing tool. Figure 5 is the current graph presentation of this playbook in the capturing tool which can be mapped to a process visualisation such as BPMN in later versions.

Each playbook has exactly one Initial step, which is represented by an oval. Similarly, each playbook has exactly one final step, which is represented by an oval and thick outline. Intermediate steps (Including indicator steps) are displayed with boxes. Also, optional steps are in grey dotted boxes. Exclusive choice steps are shown with a diamond shape.

The confidential information will be masked or removed during the process of deriving a sharable version of a playbook. A step may contain data, which cannot be revealed in any detail through a sharable version of the playbook (fully confidential). In that case, the step will be shared as an empty box with a "Confidential" tag.

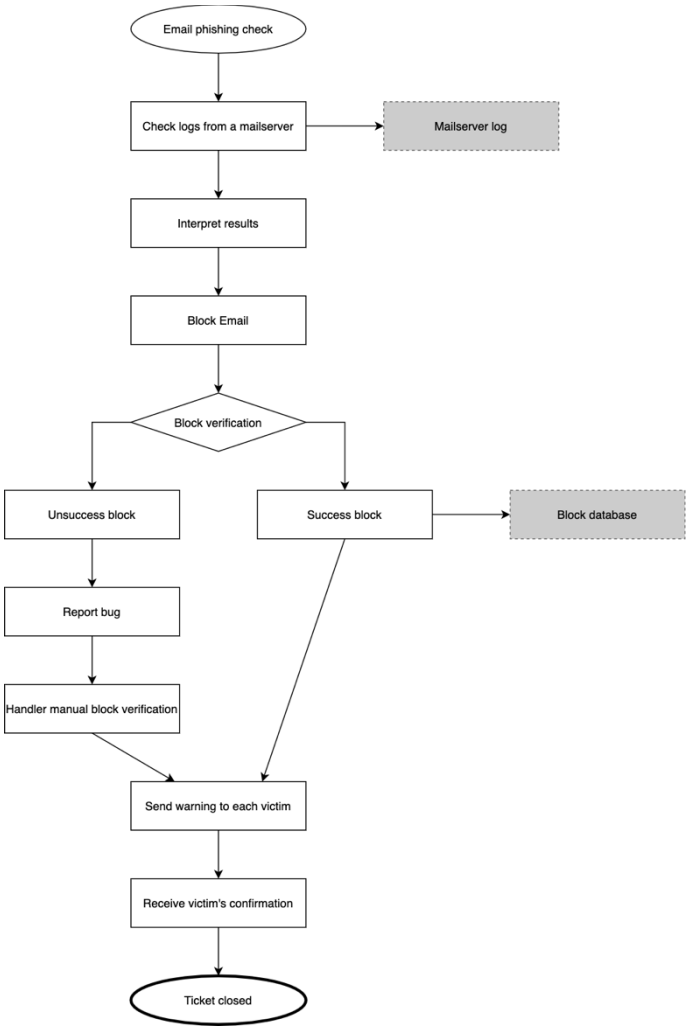

Figure 5: Graph representation of a sample playbook in the capturing tool (Email phishing)

3.4 Future plans for capturing tool

The latest version of the capturing tool is available in the SAPPAN internal repository: https://gitlab.fit.fraunhofer.de/sappan/dockersmw

The development of the tool for response and recovery knowledge capture will be continued for the recommendation and automation deliverables. Also, it has a close connection to sharing and visualisation tasks. Moreover, automation- and privacy-related aspects of the vocabulary will be revisited in the corresponding deliverables and will be integrated into the prototype. Also, another round of interviews with domain experts will be organized for the evaluation of the results.

4 Vocabulary

4.1 Importance of a common understanding

One of the primary aspects of the SAPPAN project, surrounding incident handling data is the sharing of cybersecurity playbooks across organization borders. Moreover, response and recovery knowledge will be used for the recommendation of suitable response and recovery actions to human agents, as well as for the automation of workflows. For these purposes, a common understanding of terminology and sequences is crucial. Considering the automation tasks, this common understanding will also apply to the machine level. Common understanding can be achieved by a clear definition of a mutual vocabulary for the domain including resource metadata, a sequence of steps in a workflow, common understanding of actions and action categories, representation of conditions, and connection to incidents.

4.2 Existing vocabularies and glossaries

The focus of this deliverable is on the development of a vocabulary to model response and recovery actions, since there are mature developments of widely used vocabularies and taxonomies for incident documentation, such as MISP [9] and STIX [10].

For the response and recovery vocabulary, using a vendor-agnostic format to describe workflows will benefit the goal of sharing workflows between Security Orchestration, Automation and Response (SOAR) of different platforms and products. The Open Command and Control (OpenC2) [11] standard is a machine-readable, and platform-and-product-agnostic language specification that enables the automation and interoperability for cybersecurity tools. We consider covering the generic incident handling actions and their specifications identified in the standard.

Atlassian solutions play an important role in the SOC domain, therefore the Atlassian glossary [12] is a common and well-known glossary for the domain experts which cover incident management. We have developed our vocabulary considering to be consistent with the Atlassian glossary for generic incident handling actions, followed by a reiteration based on domain expert feedback and categorizing the actions based on their reflections on the state of the system.

Moreover, Integrated Adaptive Cyber Defense (IACD) has introduced a common vocabulary for playbooks with the perspective of Security Orchestration, Automation and Response. The abstract content of playbook categories, their purpose and characteristics are discussed in [13]. Their vocabulary has influenced the SAPPAN vocabulary development, especially regarding playbook categorisation and metadata.

4.3 The SAPPAN vocabulary

4.3.1 General information about a playbook

In the following, information that should be attached to a playbook is:

- **Purpose of the playbook**: E.g., resolving alerts due to a detection of DGA activity
- **Precondition** or **Trigger**: A condition that has to hold for the playbook to be applicable.
- **Postcondition** or **Goal**: A description of the goal state after successful application of the playbook.
- **Confidentiality level**: Information about who is allowed to see the playbook. It is recommended to follow the traffic-light-protocol [8]**:**
  - *TLP:WHITE* (Public)
  - *TLP:GREEN* (Shareable with trusted external entities)
  - *TLP:AMBER* (Shareable with trusted internal entities)
  - *TLP:RED* (Confidential)
- **License**: Information about the license of the playbook. This information might be necessary, since playbooks can be considered to be intellectual property.
- **Author**: Information about the creator of the playbook. This information might prove to be useful in case there are inconsistencies or questions about the R&R actions.
- **Playbook focus**: Information about the primary focus of a playbook base on the incident handling life cycle. The following categorization is recommended by [14]**:**
  - *Preparation*
  - *Detection and Analysis*
  - *Containment, Eradication, and Recovery*
  - *Post-Event Activity*
- **Playbook category**: Information about the category of the playbook. The following categorization utilizes the categories proposed by [13] and is extended by a category for exercise playbooks:
  - *Playbook*: High-level overview of a response and recovery process (process oriented)
  - *Workflow*: Detailed (but not organization-specific) overview of a response and recovery process (technical steps)
  - *Local workflow*: Detailed and infrastructure-specific description of a response and recovery process (technical steps at system-level)
  - *Exercise*: An information rich description of a response and recovery process that is intended to be used for trainings, e.g., to teach new employees.
- **Playbook state:** Information about the state in which the playbook is:
  - *Work-in-progress*: This state is supposed to be attached to playbooks that are still a work-in-progress.

  - *Up-to-date*: These playbooks are considered to be suitable to be in-use.
  - *Under-revision*: These playbooks are currently under revision. Thus changes to the playbook can be expected in the near future.
  - *Out-dated*: These playbooks are considered to be out-dated and are not recommended to be in-use anymore.
  - *Invalid*: Playbooks with this label are not considered to be structurally sound. They do not follow the proposed playbook structure, and are also not a work-in-progress.
- **Relationship to governance:** Information about aspects related to governance and regulatory requirements. This field is supposed to be used similarly to the field "relationship to governance and regulatory requirements" in [15]**.**
- **Version**: The version number of the playbook. This avoids confusion about the version of a playbook, e.g., in case of updates.
- **ID**: The unique identifier of the playbook.

Especially in context of the automation task T4.4, the attachment of information about the automation level of the playbook becomes relevant. Potential values could be "Automated", "Partially automated", and "Not automated". Since this vocabulary, however, focuses on playbooks that are meant to be understood by human experts, a respective resource has not been defined yet.

The following RDF resources have been defined to allow attachment of respective information to playbooks that follow the playbook structure documented in D4.1:

Resource **(Predicate attached to playbook)**	**Values** **(Object attached via the predicate)**	**Description**
:hasPlaybookPurpose	string	A brief human readable description of the playbook purpose.
:hasPrecondition	string	A human readable description of the preconditions that have to hold, such that the playbook is applicable.
:hasPostcondition	string	A human readable description of the postconditions that are supposed to hold after the response and recovery process documented by the playbook has been applied. It can be considered as the description of the goal state.

:hasConfidentiality	{
	
	"TLP:WHITE",
	
	"TLP:GREEN",
	
	"TLP:AMBER",
	
	"TLP:RED"
	
	}
		The confidentiality level of the playbooks via the Traffic Light Protocol [8]**.**
		
		As confidentiality levels can also be attached to resources (see below), the confidentiality level of a playbooks should be consistent with the confidentialities assigned to individual resources. For example, a playbooks should not have the confidentiality level *TLP:GREEN*, if it contains resources with the confidentiality level *TLP:AMBER*.

:isLicencedUnder	string	Information about the license of the playbook. This information is especially relevant, if playbooks are shared.
:hasAuthor	string	Information about the author of the playbook.

:hasPlaybookFocus	{
	"Preparation",
	"Detection&Analysis",
	"Containment&Eradication&Recovery",
	"Post-EventActivity"
	}
		Information about the primary focus of a playbook in the incident handling life cycle:
		• *Preparation*
		• *Detection and Analysis*
		• *Containment, Eradication, and Recovery*
		• *Post-Event Activity*

:hasPlaybookCategory	{
	"Playbook",
	"Workflow",
	"Local workflow",
	"Exercise"
	}
		The playbook category that is assigned to the playbook. It reflects the level of detail, in which response and recovery actions are documented by the playbook:
		• *Playbook*: High-level overview of a response and recovery process (process oriented)
		• *Workflow*: Detailed (but not organization-specific) overview of a response and recovery process (technical steps)
		• *Local workflow*: Detailed and infrastructure-specific description of a response and recovery process (technical steps at system-level)
		• *Exercise*: An information rich description of a response and recovery process that is intended to be used for trainings, e.g., to teach new employees.

:hasPlaybookState	{
	"Work-in-progress",
	"Up-to-date",
	"Under-revision",
	"Out-dated",
	"Invalid"
	}
		Information about the state in which the playbook is:
		• *Work-in-progress*: This state is supposed to be attached to playbooks that are still a work-in-progress.
		• *Up-to-date*: These playbooks are considered to be suitable to be in-use.
		• *Under-revision*: These playbooks are currently under revision. Thus changes to the playbook can be expected in the near future.
		• *Out-dated*: These playbooks are considered to be outdated and are not recommended to be in-use anymore.
		• *Invalid*: Playbooks with this label are not considered to be structurally sound. They do not follow the proposed playbook structure, and are also not a work-in-progress.

:hasRelation-shipToGov-ernace	string	Information about aspects related to governance and regula-tory requirements.
:hasVersion	string	The version of the playbook. This value should be updated with every change to the playbook.
:hasId	string	The unique identifier of the playbook.

4.3.2 Overview of attaching information to steps

In deliverable D4.1, a generic playbook structure has been defined. To model specific response and recovery processes, additional information needs to be attached to the steps defined in D4.1. **[Figure 6]** illustrates the basic idea of how to attach information to these steps. Currently, at most one response and recovery actions is intended to be attached to a step in the playbook. This allows to maintain control over the order in which the actions are carried out in the playbook description.

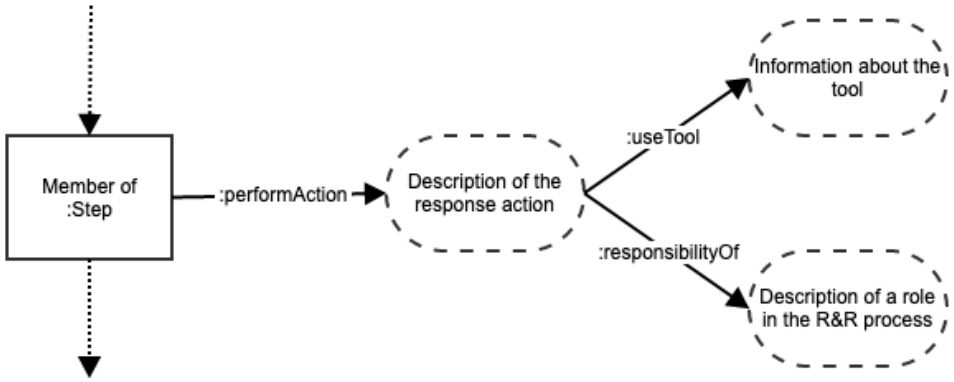

Figure 6: Intuition of how information relevant to the response and recovery process is attached to the generic playbook structure. The dashed boxes explain the idea behind more complex descriptions, which are explained in a more detailed fashion below.

4.3.3 Documentation of meta-resources

The resources that are listed as part of this chapter are intended to be attachable to every resource in the vocabulary. They provide additional information, document the confidentiality level of a resource, or define a display name that can be accessed by applications for improved user experience.

4.3.3.1 Attachment of display names to vocabulary resources

Attaching a display name to each defined resource in the vocabulary improves readability and allows tools that are built ontop of the proposed methodology to use the human-readable display name in the interface. This can improve the use experience significantly.

Resource (Predicate attached to playbook)	Values (Object attached via the predicate)	Description
:hasDisplayName	string	This is a meta-resource that can be attached to every vocabulary resource. It attaches a human-readable name as a string.

4.3.3.2 Attachment of confidentiality levels to vocabulary resources

The attachment of confidentiality levels to individual resources allows partition the resources according to the Traffic Light Protocol. In the context of sharing playbooks, it allows to automatically remove all pieces of information from the playbook that are considered to be too confidential for the recipient. As a consequence, the creation of redacted versions of playbooks from information-rich ones becomes a viable option.

Resource (Predicate attached to playbook)	Values (Object attached via the predicate)	Description

:hasConfidentiality	{
	
	"TLP:WHITE",
	
	"TLP:GREEN",
	
	"TLP:AMBER",
	
	"TLP:RED"
	
	}
		This is a meta-resource that can be attached to every vocabulary resource.
		
		It attaches a confidentiality level to a vocabulary resource via the Traffic Light Protocol to indicate the confidentiality level of this specific resource.

4.3.3.3 Attachment of comments to vocabulary resources

Under some circumstances, the attachment of human-readable comments to resources in a playbook might prove to be useful, for example, to provide additional information to human operators.

Resource (Predicate attached to playbook)	Values (Object attached via the predicate)	Description
:hasComment	string	Via this meta-resource, additional information can be provided in form of a string. It is intended to be human-readable.

4.3.4 Actions

Within the response and recovery process, actions play a crucial role, as they describe the steps taken to mitigate the effects of an incident. As such, they represent the core part of the SAPPAN response and recovery vocabulary. To not overwhelm any interested potential user of the SAPPAN vocabulary with a vast amount of actions, the actions have been defined in a hierarchical way. On the most general level, *generic actions* are defined. These actions are intended to be generic enough to model all common response actions on a conceptional level. Examples include generic actions like "locate", "block", and "restore". These generic actions, however, need to be further specified by additional resources (*specifications*) to model specific actions. The specifications are defined per generic actions. For example, a generic "locate" actions can be specified via a location type (e.g., "geographical" or "logical") and a location target (e.g., an IP address). The focus of this vocabulary as on the identification of suitable generic options. It is hence possible that not every relevant specification for a generic option is included.

An example of how actions are attached to steps in playbooks is shown in [Figure 7]. Note that this example visualizes how the action is supposed to be documented on the lowest level (RDF description). A tool that takes the raw data as input can visualize the playbook in more human-friendly ways, e.g., adding the type ":locate" of the blank node "_:action1" directly inside it, and by using the attached display name for each resource instead of the actual resource name. This way, the perhaps unintuitive way of describing actions via blank nodes in RDF can be hidden from the end-user.

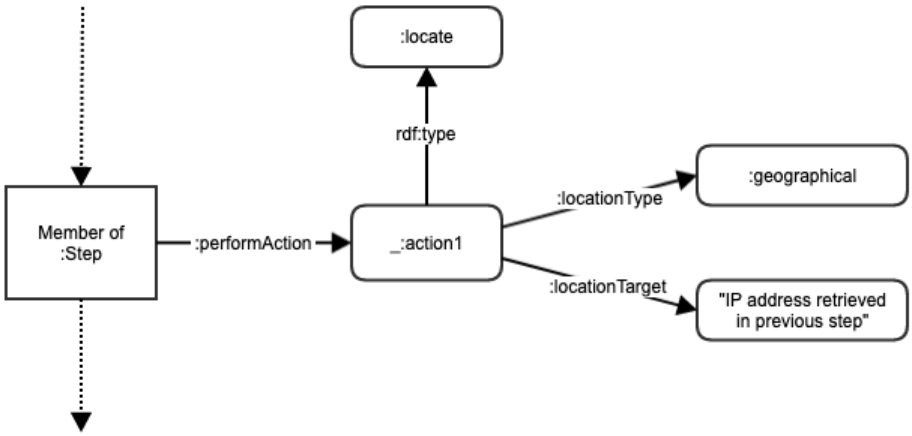

Figure 7: **Example of the visualized RDF documentation for attaching an action to a step. In this example, the action is a blank node ("_:action1") and is defined as the generic action ":locate" via "rdf:type". It is specified via the ":locationType" ":geographical" and a human-readable description of the ":locationTarget". Intuitively, this action documents the action to geolocate the IP address that has been retrieved in the previous step.**

Actions are intended to be attached to members of the "step" class in the formal playbook methodology such that at most one generic action is attached to a step via the resource ":performAction". This way the order in which steps are supposed to be carried out is clearly defined. The definition of ":performAction" is as follows.

Predi-cate	Object	Description
:perfor-mAction	blank node	The ":performAction" resource attaches a blank node (object) to a step of the workflow (member of :Step as subject). The blank node models an action. Its type is defined via the RDFS resource "rdf:type", which is supposed to be a generic action. The specifications to the generic action are also attached to the blank node. The use of blank nodes in this context has formal reasons, based on how resources are interpreted in RDF.

On a conceptual level, all generic actions are considered to be of rdf:type ":Action".

4.3.4.1 Classification of actions

The actions have been classified into four categories, which reflect their impact on the state of the target system. The categories are visualized in **[Figure 8]** and are defined as follows:

- **Management (and communication) actions:** Actions that deal with the management of the incident and inter-human communication. Examples for specific actions include the assignment of different roles for the handling of the incident and contacting a specific role.
- **State-preserving actions:** These are actions which do not change the state of the compromised system. Examples for specific actions include documentation tasks and detonation of malware in controlled environments.
- **State-restoring actions:** These actions are intended to return the compromised system into a known safe state. These actions include, for example, resetting configurations or passwords back to default and restoring a system state by loading a backup.
- **State-changing actions:** These actions actively change the state of the compromised system without intentionally returning it to a known safe state. Examples for respective actions are the application of updates and blocking specific kinds of traffic.

Classifying the defined actions comes with various benefits:

- It is easier to learn the new vocabulary in a top-down fashion. This avoids that potential users are overwhelmed by a vast amount of resources.
- The classification reflects the intended purpose of the playbook (automation), as the impact on the state of the target system also implies the potential risk of an action, if applied in case of a false-positive incident alert.
- If actions need to be looked up later, such a classification makes the lookup process significantly more efficient and thus reduces frustration of human operators.

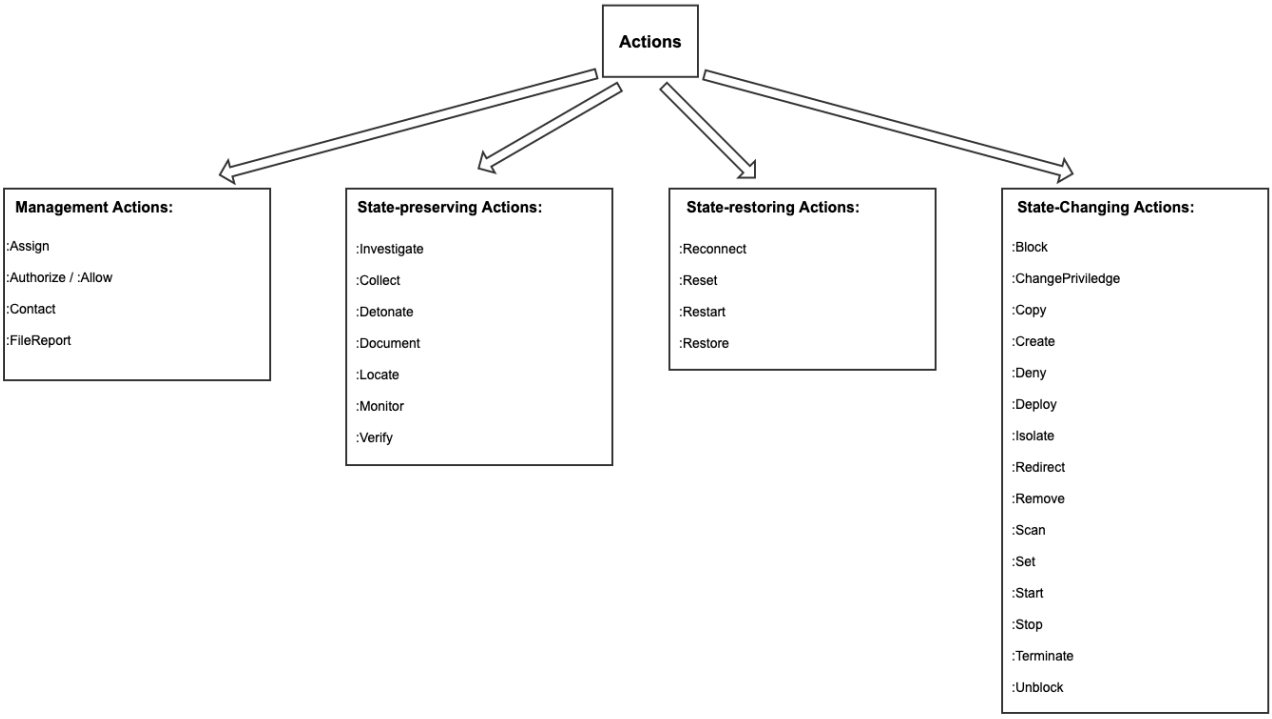

Figure 8: High-level classification of actions

4.3.4.2 Documentation of identified actions by category

Following the concept of a top-down definition of resources, a set of generic actions has been defined. These generic actions are supposed to model the most common response and recovery actions, and are further specified by additional resources. The benefit of this hierarchical definition of vocabulary resources is that the level of detail in which an action is described naturally forms a tree-like structure, which makes redaction of classified information easier. The underlying assumption is that if a more general form of an action is confidential, then the specifying information most certainly is confidential as well, while the vice-versa case does not hold in general. Another benefit is that learning and looking up the vocabulary resources becomes more efficient.

Management actions:

Actions that deal with the management of the incident, such as the assignment of different incident handling roles, and inter-human communication.

Generic Action	Description	Specification
:Assign	This generic actions deals with the assignment of various incident-related aspects. It allows for the following specifications: • **:assignRole**: Assigment of a specific role for the R&R process. • **:assignSeverityLevel**: Assignment of a severity level to the incident. As severity level scales and interpretations often vary between organizations, there is no restriction or proposal on how to document the severity level. Because severity levels are often defined by an integer, just providing the number without additional context on how to interpret it proves to be difficult if the playbooks are shared. • **:assignInformationImpact**: The impact of information leakage will be assigned. The possible values are "none" in case of no information leak, "Privacy Breach", when personally identifiable information leaks, "Proprietary Breach", when proprietary information of organisation leaks, and "Integration Loss" when sensitive data is manipulated. • **:recoveryEffort**: Assignment of a recoverability effort to the incident. The possible values are "Regular" when the recovery is possible with existing resources, "supplemented" when additional resources for recovery are needed, "Extended" when additional support and resources for the recovery are needed, and "Not Recoverable" when no recovery from the state is possible.	• :assignRole ○ "Communications lead" ○ "Customer support" ○ "Incident commander / Incident manager" ○ "Incident responder" ○ "Incident stakeholders / Incident observers" ○ "Operations lead" ○ "Second line support" ○ "Site reliability engineer" ○ "Subject matter expert" ○ "Tech lead" ○ "Other" • :assignSeverityLevel: ○ string with the specific severity level (if a specific severity level is dictated via the playbook) ○ "" (if the assigned value if left to the human operator) • :assignInformationImpact ○ "Privacy Breach" ○ "Proprietary Breach" ○ "Integrity Loss" ○ "None" • :recoveryEffort ○ "Regular" ○ "Supplemented" ○ "Extended" ○ "Not Recoverable"

:Authorize / :Allow	Get the authorization to apply a specified action. The role that is supposed to authorize the action can also be specified.	• :authorizeBy
		  ○ "Communications lead"
		  ○ "Customer support"
		  ○ "Incident commander / Incident manager"
		  ○ "Incident responder"
		  ○ "Incident stakeholders / Incident observers"
		  ○ "Operations lead"
		  ○ "Second line support"
		  ○ "Site reliability engineer"
		  ○ "Subject matter expert"
		  ○ "Tech lead"
		  ○ "Law enforcement"
		  ○ "Customer"
		  ○ "Other"
		• :authorizationPurpose
		  ○ Human-readable description as a string

:Contact	Establish contact
	• to a specified role
	• via a specified communication channel
	• for a specified purpose
		• :hasReceiver
		  ○ "Communications lead"
		  ○ "Customer support"
		  ○ "Incident commander / Incident manager"
		  ○ "Incident responder"
		  ○ "Incident stakeholders / Incident observers"
		  ○ "Operations lead"
		  ○ "Second line support"
		  ○ "Site reliability engineer"
		  ○ "Subject matter expert"
		  ○ "Tech lead"
		  ○ "Law enforcement"
		  ○ "Customer"
		  ○ "Other"
		• :viaChannel
		  ○ "Telephone"
		  ○ "Email"
		  ○ Any specification of a communication service (Slack, Zoom, ...)
		• :forPurpose
		  ○ Human-readable description of the communication purpose

:FileReport	This action is for sending a report, e.g., GDPR report.	• :reportType
		  ○ A string defining the kind of report, e.g., "GDPR report"
		• :reportReceiver
		  ○ "Communications lead"
		  ○ "Customer support"
		  ○ "Incident commander / Incident manager"
		  ○ "Incident responder"
		  ○ "Incident stakeholders / Incident observers"
		  ○ "Operations lead"
		  ○ "Second line support"
		  ○ "Site reliability engineer"
		  ○ "Subject matter expert"
		  ○ "Tech lead"
		  ○ "Law enforcement"
		  ○ "Customer"
		  ○ "Other"

State-preserving actions:

Actions that are not management actions and that do not result in changes to the infrastructure. They do not change the state of a network, device, configuration, etc.

Generic Action	Description	Specification

:Investigate	This is an action to analyze and understand the behaviors of a resource, service, tool or incident. This action will have specifications such as the location of the investigation target, e.g., firewall logs, and the purpose of investigation, e.g., anomaly detection.	• :investigationOn
		  ○ :application
		  ○ :domain
		  ○ :emailAddress
		  ○ :incidentReport
		  ○ :ip
		  ○ :packet
		  ○ :port
		  ○ :service
		  ○ :url
		  ○ :user
		• :investigationLocation
		  ○ The location of the investigation target, given as a string. This could, for example, refer to a specific file.
		• :investigationPurpose
		  ○ The purpose for the investigation, given as a string.
		• :investigationMethod
		  ○ The method in use for analyzing the system, e.g., query.

:Collect	Collection of a specified type of data from a specified location, without enriching it with additional information. This action is considered to be used when information is collected from the specified location to a safe location, like an archive or a database, without overwriting existing information. It is hence considered to be state-preserving regarding the compromised system, e.g, taking a page screen shot or collecting a log file.	• :collectFrom • :collectTo • :collectionType ○ E.g., taking a "pageScreenShot" or collecting a "logFile"

:Detonate	Execute malware in a controled, isolated environment (such as a virtual machine).	• :detonationLocation
		  ○ The location of the controled, isolated environment.
		• :detonationTarget
		  ○ The malware that should be executed.

:Document	Documentation of incident-related aspects. Unlike the ":Collect" action, the ":Document" actions deals with information that is aggregated or enriched, e.g., by human operators.	• :documentAs
		  ○ E.g., "written report", "bullet points", ...
		• :documentPurpose
		  ○ Human-readable description of what should be documented

:Locate	Find the location of a physical or logical item of interest. The type of the location and the item of interest need to be specified.	• :locationType
		  ○ :geographical
		  ○ :logical
		• :locationTarget

:Monitor	Monitor a target item of interest for a specified duration. This action can further be specified by attaching information about the intended tool to the step (see below, section Tools). Additionally, the type of monitoring (active, passive, reactive) can be specified.	• :monitoringTarget
		• :duration
		• :monitoringType
		  ○ "Active"
		  ○ "Passive"
		  ○ "Reactive"

:Verify	Check if a specified hypothesis is true.	• :hypothesis

State-restoring actions:

Actions that return the system or parts of it to a previous state.

Generic Action	Description	Specification

:Reconnect	End and establish an existing connection.	• :connectionProtocol
		  ○ "TCP"
		  ○ "UDP"
		  ○ ...
		• :connectionFrom
		• :connectionTo

:Reset	Return a configuration or a password back to default.
	
	In this context, the default setting is considered to be a known state.
		• :resetTarget
		  ○ Password for a specified service
		  ○ A specified onfiguration

:Restart	Stop and then start a running application, process, or system.
	
	Note: Depending on context, this action is not guaranteed to return the restarted system back to a known state and might under some circumstances be state-changing.
		• :restartTarget
		  ○ :application
		  ○ :process
		  ○ :system

:Restore	Return the system to a previously known state, e.g., by loading a backup. The difference to the ":reset" option lies in the scope (":reset" for files, values, database entries; ":restore" for systems).	• :targetSystem • :backupLocation

State-changing actions:

Actions that apply changes to the system without intentionally returning it to a known safe state.

Generic Action	Description	Specification

:Block	Block any kind of traffic of specified type with a specified block level.	• :hasType
		  ○ :application
		  ○ :domain
		  ○ :emailAddress
		  ○ :emailDomain
		  ○ :ip
		  ○ :packet
		  ○ :port
		  ○ :service
		  ○ :url
		  ○ :user
		• :hasBlockLevel
		  ○ machine
		  ○ organisationIntranet

:Change-Priviledge	Change the priviledge of a specified account to a specified level. This can, for example, be used to counter the effects of an elevation of priviledge, or to reduce the priviledges of an account, if it shows suspicious behavior.	• :targetAccount
		  ○ The target account for which the priviledge level is supposed to be changed, given as a string.
		• :newPriviledgeLevel
		  ○ The new priviledge level that the specified account should have, given as a string.

:Copy	Copy a file from a specified source to a specified location. As the destination location could also be on a compromised machine, this actions is considered to be state-changing.	• :source
		  ○ string
		• :destination
		  ○ string

:Create	Add a new entity with a certain type.	• :createTarget
		• :hasEntityType
		  ○ :data
		  ○ :directory
		  ○ :file
		  ○ :flow
		  ○ :process
		  ○ :user

:Deny	Actively prevent a programm execution, traffic transfer, or similar from reaching its goal.	• :denialTarget

:Deploy	Deploy a specified system, executable, or change at a target location.	• :targetLocation
		  ○ The target location of the deployment.
		• :deploymentType
		  ○ :coldStandby
		  ○ :hotfix
		  ○ :update

:Isolate	Isolate a specific entity that prevent modification or access to processes or resources.	• :isolationTarget
:Redirect	Change the traffic flow to a different destination from the original one.	• :trafficType • :originalTrafficDestination • :newTrafficDestination

:Remove	Remove an entity with a certain type.	• :hasEntityType
		  ○ :data
		  ○ :directory
		  ○ :file
		  ○ :flow
		  ○ :process
		  ○ :user
		• :removeTarget

:Scan	Examination of a specific entity or environment.	• :scanTarget

:Set	Assign a value to a specific entity or state.	• :modificationTarget
		  ○ The target entity or state.
		• :newValue
		  ○ The new value for the specified entity or state.

:Start	Start a process, application or system.	• :hasEntityType
		  ○ :application
		  ○ :process
		  ○ :system

:Stop	Stop a process, application or system.	• :hasEntityType
		  ○ :application
		  ○ :process
		  ○ :system

:Terminate	Terminate a specified thread or process.	• :terminationTarget

:Unblock	Unblock any kind of traffic of specified type with a specified unblock level.	• :hasType
		  ○ :application
		  ○ :domain
		  ○ :emailAddress
		  ○ :emailDomain
		  ○ :ip
		  ○ :packet
		  ○ :port
		  ○ :service
		  ○ :url
		  ○ :user
		• :hasUnblockLevel
		  ○ machine
		  ○ organisationIntranet

Custom action:

As we cannot expect that the proposed actions allow to model every thinkable response and recovery actions, we additionally define a resources that allows for the custom definition of an action. Since this, however, conflicts with the goal of a mutual understanding of resources, which is especially crucial in the context of playbook sharing, the use of a custom actions should only be considered as a last resort. It should not be used, if the action can also be modeled via a single or a combination of defined vocabulary actions.

Generic Action	Description	Specification
:OtherAction	This action allows to defined a custom action and should only be used, if the action cannot be modeled as a combination of the defined vocabulary actions.	This action requires the following to be defined: • The display name of the resource needs to be defined via the meta-resource ":hasDisplayName". • The resource needs to be described via the resource ":hasCustomDescription", as no official documentation of the action exists. The ":hasCustomDescription" resource is only defined for the custom action. No further constraints are made about which information needs to be provided. It is seen as the responsibility of the playbook designer to carefully specify the custom action.

4.3.4.3 Meta-resources for actions

Assignment of action types:

Assignment of the action type to each action allows for a machine-readable description of the respective classification.

Resource	Values	Description
:hasState-Type	• "Management action" • "State-preserving action" • "State-restoring action" • "State-changing action" • "Unspecified type of action" (this is the default value of the custom action)	Via this resource, information about how an action is classified in regards to the state-based classification can be added to the respective action. It also allows to specify the type of custom actions, which by default are considered to be of an unspecified type of action.

Assigment of responsibilities:

Via the attachment of information about which role is supposed to carry out a certain action, the workflow can be further refined.

Resource	Values	Description
:responsi-bilityOf	• "Communications lead" • "Customer support" • "Incident commander / Incident manager" • "Incident responder" • "Incident stakeholders / Incident observers" • "Operations lead" • "Second line support" • "Site reliability engineer" • "Subject matter expert" • "Tech lead" • "Law enforcement" • "Customer" • "Other"	This resource can be used to attach a role to an action, to specify whose responsibility the respective action is.

Other generic meta-resources:

Resource	Values	Description
:hasCommand	string	This resource is used to attach the exact script execution command for the action.
:hasStartingTime	integer	Information about the starting time of an action. It should be 0 or higher. The default value is 0, which means immediate execution of the action.
:hasExpiration-Time	integer	Information about the expiration time of an action. It should be -1 or higher. The default value is -1, which means the action will not be expired.
:hasAssociat-edRisk	string	Information about the risk level of the action. It is supposed to indicate, how damaging the application of the action would be in case of a false-positive incident alert.

4.3.5 Conditions

Since the formal methodology described in deliverable D4.1 allows for exclusive branching, the branching conditions need to be documented within the playbooks.

Resource (Predicate)	Values (Object)	Description
:hasCondition	string	This resource can be attached to exclusive branching steps. It means to attach a human-readable condition to the step, depending on which a branch in the workflow is selected.
:hasOption	blank node	This resource is used to connect a blank node to the exclusive choice step to model one of the option. The blank node is further specified as the subject for the ":hasValue" resource, and as the object for the ":followIf" resource.
:hasValue	string	This resource attaches a value in form of a string to a blank node that has been defined as an option for an exclusive branching step via the ":hasOption" resource.
:followIf	blank node	This resource is attached to the steps that are directly reachable from an exclusive choice step to indicate, which option it connects to. It hence connects the step to with the blank node that models the option representing the respective branch.

The application is illustrated in the following example. **[Figure 9]** Please keep in mind that example shows how exclusive branching is modeled on the lowest level, and does not necessarily reflect how a human operator would see it in a tool built on top of this representation.

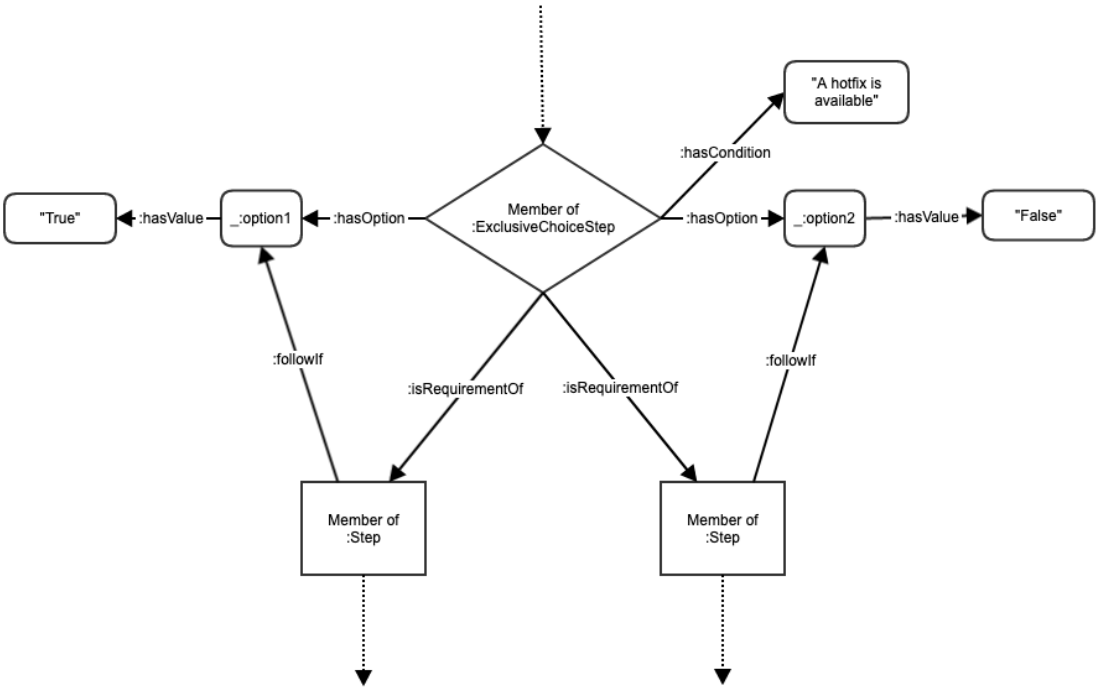

Figure 9: Example of the visualized RDF documentation for attaching conditions

Note that exclusive branching can be used to define conditional loops: One of the branches can refer back to the exclusive choice step. In such a case, the loop is carried out until the other branch is taken.

4.3.6 Tools

Within a response and recovery process it might be useful to know which tool to use for a certain action. Having respective information defined in a playbook can speed up the response and recovery process.

4.3.6.1 Attaching tool information to actions

Tool information is attached to actions. More specifically, it is connected to a blank node that models a response and recovery action. Tool information itself is modeled via blank nodes as well, as this allows to differentiate different versions and instances of the same tool. An example is shown in **[Figure 10]**. In this simple example, a blank node is used to model the tool, but no differentiation between tool instances is documented. This differentiation, however, might become important for web-based tools, where a specific instance has to be used to perform an action, e.g., due to data protection regulations.

Resource (Predicate)	Values (Object)	Description
:useTool	blank node	Via the ":useTool" resource, a blank node representing the tool (object) is connected to a blank node representing an action (subject).

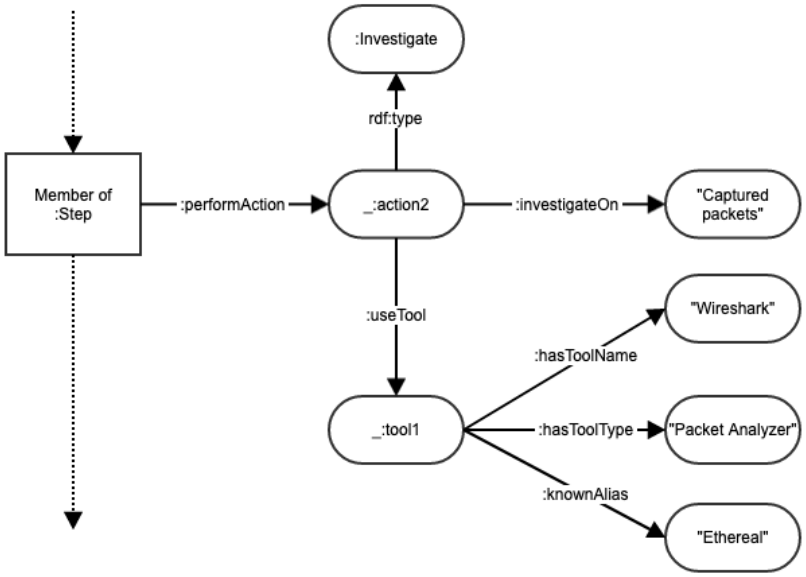

Figure 10: An example of how tool information can be attached to an action. This, again, represents the lowest level of the playbook description and hence contains blank nodes that are used to model a specific action ("_:action2") and a specific tool ("_:tool1").

4.3.6.2 Relevant information about tools in playbooks

Via the following resources, information about the tool can be attached to the blank node that represents the specific instance of the tool.

Predicates that can be attached to blank nodes representing tools	**Description / Values**
:hasToolName	The name of the tool as a string.
:knownAlias	Known alias of the tool, e.g., name before rebranding. This predicate can be assigned more than once to the tool, since a tool can have several known aliases.
:hasToolType	The type of the tool as a string. By default, the following tool types (values) are supported: • "Case Management" • "Communication Channel" • "Documentation" • "Flow Analyzer" • "Issue Management" • "Packet Analyzer" • "Service Desk" • "SIEMs and IDPSs" • "Threat Intelligent" • "Monitoring Service" • "OSINT"
:documentationLocation	The location where the documentation (e.g., user guide) can be found, given as a string. (Allows to describe the location in text, to provide a link, etc.)
:toolHomepage	The official homepage of the tool.
:localInstance	Location of the local instance of the tool, e.g., a URL.

4.3.6.3 Pre-selection of tools

Tool Type	**Tool**
Case Management	• The Hive

Communication channels	• Mattermost • MS Teams • Mumble (Voice over IP, open-source) • Rocket.Chat • Skype for Business • Slack • Zoom
Documentation	• Confluence
Flow Analyzer	• Nfdump • Yaf • Real-Time NetFlow Analyzer (Solarwinds)
Issue Management	• Jira • OTRS • RT • RTIR
Packet Analyzer	• Arkime (formerly known as Moloch) • Capsa Network Analyzer • Charles Web Debugging Proxy • Carnivore (software) • CommView • dSniff • EndaceProbe Analytics Platform by Endace • ettercap • Fiddler • Kismet • Lanmeter • Microsoft Network Monitor • NarusInsight • NetScout Systems nGenius Infinistream • ngrep, Network Grep • OmniPeek, Omnipliance by Savvius • SkyGrabber • snoop • tcpdump • Observer Analyzer • Wireshark (formerly known as Ethereal) • Xplico Open source Network Forensic Analysis Tool
Service Desk	• Jira Service Desk
SIEMs and IDPSs	• AlienVault® OSSIM • Snort • Suricata

Threat Intelligent	• IntelIMQ • MISP • OpenCTI • OTX • STIX

4.4 Initial ideas for automation support

As task T4.4 considers automation of response and recovery steps based on the playbook methodology proposed in task T4.1, respective plans have already been considered in the design of the results of this deliverable. Since the playbooks are already machine-readable due to the use of RDF, which supports serialization into popular formats such as JSON, automation routines can be built on top of the playbook description. In the following, ideas about different aspects of automation are discussed, including the automation of response and recovery actions, machine-readable conditions for automated decision-making, and the re-use of MISP taxonomies to connect SAPPAN playbooks to incident reports. Since, however, automation is in focus of task T4.4 (and not T4.1, which is the task that this deliverable is written for), please consider the following descriptions as initial ideas that will be refined and possibly changed as part of the future deliverable D4.7.

4.4.1 Automation of response and recovery actions

Specific actions connected to steps in the playbook can be linked to automated response and recovery actions (e.g., by attaching an execution command for a respective script as additional information to the action in the playbook). But as respective automated actions are likely to take inputs in form of parameter assignments, any program that goes automatically through a playbook requires information about which information is important and where to find it. This dictates us two preliminary requirements:

1. Each automated action defined in the vocabulary requires a definition of what information is necessary for automation. This information can be both conceptually important (information that in general is important to carry out the respective action), or specifically important (information that is important to carry out the respective action in a specific infrastructure). Since we cannot make any reasonable assumption about the infrastructure, it is probably best advised to focus on conceptually important information.
2. The playbooks need to provide information about where to look up the parameter values necessary to execute the automated actions. As this information is most likely specific to the incident, gathering the inputs from the incident report seems to be a good starting point. Expressing respective information in a machine-readable playbook, however, requires the use of the same vocabulary as the incident report when referring to it.

As a consequence of these two preliminary requirements, support for incident report vocabularies might prove to be useful for automation.

4.4.2 Machine-readable conditions for automated decision-making

As a starting point for automated decision-making, consider rule-based systems. Conceptionally, such systems need to know two things: They need to understand the rules, which deterministically dictate decision-making, and they need to know the respective parameter values in order to apply these rules. If we step away from rule-based systems towards approaches based on heuristics, these requirements do not substantially change. Again, such a system needs to know when to apply an action (e.g., if some threshold is exceeded), and where to get the respective values from (e.g., by applying a specified heuristic with certain inputs).

On a conceptual level, we can derive two preliminary requirements from this:

1. We need a machine-readable description of rules. This may include logical statements based on the *if-then-else* scheme, as well as decisions based on the outcome of specified heuristics (e.g., similar to *switch-case*, but extended to intervals of the range of the heuristic). Additionally, conditional loops could be considered, which could conceptually be modelled using *if-the-else* statements over variables that can change their assigned values within the loop.
2. The system needs to know the inputs, i.e., the parameter values. As decisions made within the response and recovery process depend on the specific incident, it might be beneficial to consider that these parameter values could already be part of the incident report.

In combination, this means that decision-making could be modelled as rules over variables, which are assigned via the results of method calls. The methods themselves could take incident data as inputs. These, however, are only preliminary considerations, as automation is in scope of task T4.4, and not T4.1.

4.4.3 Re-use of MISP taxonomies to connect playbooks and incident reports

In deliverable D4.1, the theory of translating MISP machine tags into RDF triples in the formal methodology of SAPPAN has been described. However, since the focus of MISP is incident documentation, and not the modelling of response and recovery actions, it becomes more relevant for our purposes as a way to connect playbooks to incident reports, i.e., by using MISP taxonomies to define conditions for automated decision-making in playbooks. Using the same vocabulary for such conditions as the incident report allows to trivially lookup respective values in the incident report. As a small example, consider the following condition that utilizes the MISP phishing taxonomy [16]:

IF report.misp.phishing.techniques == "email-spoofing" **AND** report.misp.phishing.distribution == "whaling"

THEN follow_left_workflow

ELSE follow_right_worflow

In this example, a simple *if*-condition looks up, whether the fields *techniques* and *distribution* in the incident report using the MISP taxonomy *phishing* equal certain predefined values ("email-spoofing", "whaling"). Depending on the result, one of two possible

workflows is executed. Note that this simple, rule-based example could be extended, e.g., by using values from the incident report as inputs for a heuristic.

As MISP provides many different taxonomies, is widely used and open-source, it seems to be the ideal starting point to connect playbooks and incident reports for the purpose of automation. Hence, the use of MISP taxonomies for this purpose will be further considered as part of the automation task T4.4.

4.5 Standardization

Standardization of playbooks plays an important role, when the playbooks are considered to be shareable. But also if playbooks are not shared, the existence of a standard might improve the quality of documented playbooks. Smaller SOC would benefit, since they do not need to think of their own playbook methodology, and SOC agents do not need to learn local documentation styles when they change employers. Overall, standardizing the documentation of response and recovery actions has the potential to improve the state of the art.

On 12.01.2021, shortly before the submission deadline of this deliverable, OASIS has publicly announced the approval of a security playbook specification: The Collaborative Automated Course of Action Operations (CACAO) [17] specification defines a schema and taxonomy for cybersecurity playbooks, while also considering standardized creation, documentation, and sharing of respective playbooks. The scope of the CACAO specification consequently matches the playbook-related aspects of the SAPPAN project. Hence, the SAPPAN consortium plans to provide feedback during the open-feedback phase of the CACAO specification. This way, knowledge gained during the SAPPAN project could find its way into the CACAO standardization, and could thus help to improve the resulting standard.

5 Results of interviews with target group

Interview sessions have been organized in two round to collect feedback and suggestions for the developed vocabulary and capturing approach from domain experts. Information about current approaches, advantages, disadvantages, and their expectations has been gained from different organizations participating in the project in the first round. The answers to the first round of interviews include much sensitive and confidential information, but the lesson learned has been addressed in the approach development. Information about the first rounds of interviews has been included in deliverable D4.3 and the addressed issues are listed in the section on updates on knowledge capturing tool in this deliverable.

The second round of interviews was held with eight Security Operation Centre (SOC) members of CESNET, Dreamlab, HPE, and Masaryk University. The Interview sessions have been organized to collect feedback on the developed vocabulary from domain experts. The interviewees have different backgrounds and roles in their team which helped us to cover a broader view regarding the essential vocabulary. Each interview started with a brief introduction and description of the main goals, followed by a brief presentation of the current development of the capturing tool and vocabulary. The interview questions have covered the following aspects:

- General limitations and pain points of existing response and recovery capturing tools
- Domain experts general impression of the developed vocabulary
- Detailed comments on the word choices and categories of the developed vocabulary
- Feedback on missing resources
- Ideas for automation of response and recovery processes

We have applied the feedback to the vocabulary, especially the detailed wording choices and categories, and missing resources, into our vocabulary. Ideas for automation of the response and recovery processes will be considered for the final automation deliverable D4.7.

6 Conclusion

In this deliverable, open trade-offs documented in D4.1 have been decided, mitigation measures for the risk of confidentiality concerns regarding playbook sharing have been presented, and updates to the knowledge capturing tool developed in T4.2 have been documented. As the core part of this deliverable, an incident response and recovery vocabulary has been defined, which can be used to attach response and recovery information to the playbooks following the formal methodology that has been presented as part of D4.1. In the design of the vocabulary, aspects like machine-readability of the format, confidentiality of individual pieces of information and role responsibilities have been considered. Additionally, initial ideas on adding support for automated execution of response and recovery actions within the playbooks have been presented. Also, plans to utilize insights gained during the SAPPAN project to participate in the open-feedback phase for the newly emerging OASIS playbook specification CACAO have been proposed.

7 References

[1] RDF Primer (W3C documentation): https://www.w3.org/TR/rdf11-primer/

[2] SPARQL Query Language (W3C documentation): https://www.w3.org/TR/sparql11-query/

[3] RDF Schema (W3C documentation): https://www.w3.org/TR/rdf-schema/

[4] OWL 2 Web Ontology Language Primer (W3C documentation): https://www.w3.org/TR/owl2-primer/

[5] Business Process Model and Notation Specification (documentation by the Object Management Group): https://www.omg.org/spec/BPMN/2.0/

[6] Pre-defined playbooks by the Incident Response Consortium: https://www.incidentresponse.com/playbooks/

[7] Example playbooks by IACD: https://www.iacdautomate.org/playbook-and-workflow-examples

[8] TRAFFIC LIGHT PROTOCOL (TLP) - FIRST Standards Definitions and Usage Guidance - Version 1.0: https://www.first.org/tlp/docs/tlp-v1.pdf

[9] MISP Taxonomies (overview and taxonomy repository): https://github.com/MISP/misp-taxonomies

[10] STIX (official introduction): https://oasis-open.github.io/cti-documentation/stix/intro

[11] Open Command and Control (OpenC2) Language Specification Version 1.0 (OASIS Committee Specification 02): https://docs.oasis-open.org/openc2/oc2ls/v1.0/cs02/oc2ls-v1.0-cs02.html

[12] The language of incident management (Atlassian glossary): https://www.atlassian.com/incident-management/glossary

[13] Introduction to Integrated Adaptive Cyber Defense (IACD) Playbooks: https://static1.squarespace.com/static/5a94b67ff93fd440f0516297/t/5b3fb74170a6ad89b74de3b0/1530902339037/Introduction+to+IACD+Playbook+.pdf

[14] Computer Security Incident Handling Guide: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-61r2.pdf

[15] Introduction to Integrated Adaptive Cyber Defense (IACD) Playbook Thin Specification: https://static1.squarespace.com/static/5a94b67ff93fd440f0516297/t/5b3fb799352f538aa0beaa04/1530902425941/IACD+Playbook+Thin+Specification.pdf

[16] MISP Phishing Taxonomy: https://github.com/MISP/misp-taxonomies/blob/main/phishing/machinetag.json

[17] CACAO Security Playbooks v1.0 (OASIS committee specification): https://www.oasis-open.org/2021/01/15/cacao-security-playbooks-v1-0-approved-as-a-committee-specification/

Note: All links in this deliverable were last accessed on 28.01.2021.