Translating Incident Response Playbooks from Enterprise-Specific Format to the CACAO standard

ARTURS VLADIMIRS VISNAUSKS, University of Twente, The Netherlands

With the sophistication and scale of cybersecurity attacks at an all-time high, many organisations use incident response playbooks. Playbooks are structured sets of instructions that guide security personnel in preventing, detecting and remediating cyberattacks. They also provide a way to automate repetitive processes, further reinforcing the security defence mechanism of a company. The Collaborative Automated Course of Action Operation (CACAO) is an ongoing project that aims to standardize incident response playbook format to enhance interoperability and collaboration between security teams. Since the CACAO standard was introduced only recently, many vendors still use their own playbook formats, and there is a lack of publicly available tools to translate them to the new standard. This research focuses on exploring possible approaches to translate proprietary playbook formats to the CACAO standard. We conduct an extensive analysis to identify the key components of the most common playbook formats. Then, we develop a proof of concept utilizing mapping files to transform vendor-specific playbooks to valid playbooks of the CACAO format. Finally, we evaluate the translation accuracy of the developed prototype tool.

Keywords: Incident Response, Security Automation, Cybersecurity Playbooks, CACAO Standard

1 INTRODUCTION

Efficient cybersecurity incident management is crucial for the well-being of any organisation. With cyberattacks evolving and becoming more complex [4], in the past years, there has been a rise in demand for Security Orchestration Automation and Response Incident (SOAR) products.

SOAR products are based around the concept of incident response playbooks, which are step-by-step guidelines for performing countermeasures against cyber threats [15]. They provide a possibility to define courses of action for threat detection, mitigation, response and recovery in a structured and unambiguous way. This approach allows an organisation to avoid potentially flawed subjective judgement of security analysts ensuring consistent and predictable quality of security operations [8]. Furthermore, it creates an opportunity to accumulate and share cyber attack response and prevention knowledge among security personnel of an organisation, which makes the process of onboarding new members of security teams more streamlined [17]. More generalized versions of playbooks, stripped of the organisation-specific and sensitive data, can also be made publicly available, contributing to the increase of cyber-awareness [2] and allowing the new companies to be more competent by leveraging the incident response knowledge of larger organisations. Finally, some SOAR frameworks provide automation of some actions, allowing for faster response times and more effective protection against increasingly automated cyber attacks [15].

Growing interest in playbook-oriented cyber incident response practices enabled the emergence of numerous SOAR product vendors, which resulted in a situation where playbook file formats and data structures vary per organisation [14], introducing challenges for playbook sharing within and between organisations, as each format requires knowledge of the underlying data structure and proficiency in the according software used to process the playbooks.

Significance of cooperation by exchange of cyber attack and defence information is stressed by the recently implemented Network and Information Security Directive (NIS2) [18]. Aiming to improve cybersecurity resilience and incident response across critical sectors in the EU, this directive introduced mandatory reporting of severe cybersecurity incidents and strongly endorsed incident response information sharing through national agencies, implying the importance of a unified and interoperable playbook format.

The Collaborative Automated Course of Action Operation (CACAO) [11] is an ongoing effort by OASIS to provide the industry with a standardized, shareable and machine-readable specification of incident response playbooks, facilitating interoperability and automation of incident response and prevention strategies.

To be resilient in the constantly evolving landscape of cyber threats, it is beneficial for companies to transition to a standardized playbook format, which implies translation of a company's collection of enterprise-specific playbooks to equivalent CACAO playbooks. Due to potentially large amounts of data, manual translation of every playbook is not viable, which calls for an automated solution. Furthermore, it is wasteful for each organisation to develop their own software that programmatically translates playbooks, since many companies use the same SOAR product vendors.

These issues create the need for a unified solution that allows translation from arbitrary vendor-specific playbooks to the CACAO standard. Nevertheless, there is a scarcity of publicly available tools designed to solve this problem.

Our research aims to investigate the above-described problem by defining the following research question:

RQ: What approach can be developed to systematically convert proprietary incident response playbooks into the CACAO standard format?

This research question can further be broken down into several sub-questions:

SRQ1: To what extent can playbooks retain their original structure and fidelity after conversion to CACAO format?

SRQ2: To what extent can the translation process to the CACAO standard be generalized to support arbitrary enterprise-specific playbook formats?

To address the defined research questions, we conducted a comparative analysis of four major SOAR playbook formats, designed a modular translation tool based on vendor-specific mapping files, and evaluated its ability to convert playbooks into the CACAO standard. The resulting tool demonstrates that automated, format-agnostic translation is feasible and provides a foundation for improving interoperability in incident response automation.

Author's address: Arturs Vladimirs Visnausks, a.v.visnausks@student.utwente.nl, University of Twente, The Netherlands.

2 METHODOLOGY

In this section, we present the methodology we used to answer the proposed research questions.

2.1 Literature review

We first determined the state of the art by reviewing the publicly available literature and internet resources.

To find information about various SOAR product playbook formats and the CACAO standard, as well as to discover the latest developments in the incident response automation sphere, we conducted a search of relevant grey literature such as industry white papers, documentations and cybersecurity community publications. For academic literature, the search utilized Google Scholar digital library.

We applied post-filtering to exclude non-peer-reviewed journal articles or conference papers, as well as documents that were outdated, lacked technical depth, or did not focus on playbooks or the CACAO standard. After filtering, we selected 18 documents in total — 9 academic publications and 9 grey literature sources.

The search employed combinations of the following keywords:

- "incident response playbooks",
- "cybersecurity automation",
- "playbook transformation" or "playbook translation",
- "CACAO standard" OR "CACAO playbooks"
- specific SOAR product names.

2.2 Preliminary analysis

To identify challenges of translating enterprise-specific playbook formats to CACAO standard, we conducted a manual comparative analysis between CACAO format and playbooks of various SOAR vendors. We used a dataset of incident response playbooks of multiple vendors [9] as a source for our analysis samples. This dataset was scraped during a 2024 study by Schlette et al. [15].

For our analysis, we selected four SOAR product vendors based on their market share and availability of publicly accessible playbooks. Resulting SOAR vendors chosen for the analysis are Cortex XSOAR [12], Splunk SOAR [16], FortiSOAR [5] and Palo Alto Demisto [13].

We conducted a detailed examination of publicly available documentation and representative playbook files for each format. The analysis involved identifying and mapping individual attributes from the CACAO format to their closest equivalents in the vendor-specific formats, as well as exploring strategies to infer information that cannot be directly mapped.

Through this approach, we aimed to create a set of preliminary comparison tables highlighting direct or partial mappings as well as any information gaps between the formats. The end goal of the analysis was to build a knowledge foundation for the design of the prototype tool through identifying common mapping patterns, observable for all four playbook formats.

3 STATE OF THE ART

The notion of an incident response playbook forms the basis of our research. Works by Bollinger et al. [1], Schlette et al. [14] and Stevens et al. [17] contributed to the definition of the core concepts and requirements of a playbook to enhance the usability and effectiveness of real-world applications. In particular, our research is centered around the OASIS CACAO standard [11].

The work of Kaufhold et al. [8] explored challenges in cyber situational awareness and highlighted the importance of structured response protocols. While not focused on CACAO directly, the study provides context for operational expectations from playbook frameworks. Bouwman et al. [2] investigated large-scale threat intelligence sharing networks and found that interoperable, standardized knowledge records such as playbooks improve organisational resilience. Moreover, the ENISA Threat Landscape 2023 report [4] highlights the growing complexity and volume of cyber threats and explicitly emphasizes the need for standardized and automated incident response practices, reinforcing the relevance of the CACAO standard.

The 2022 project of Akbari Gurabi et al. [7] resulted in the development of SASP, a CACAO playbook management and sharing tool that features translation to Business Process Model and Notation (BPMN) for visualization of playbooks. Their implementation highlights the importance of visual representation for comprehensibility and reuse of playbooks.

In 2023, Empl et al. [3] developed a working method to translate security advisories in Common Security Advisory Framework (CSAF) format into CACAO playbooks. This research, though focused solely on CSAF format, provides valuable insights for potential generalization of the algorithm.

Furthermore, the 2024 IEEE publication of Schlette et al. [15] examines the similarities and differences of various playbook formats used in the industry. As part of their research, Schlette et al. created and analysed a dataset of all publicly available incident response playbooks, classified by their formats, currently available on GitHub [9].

Schlette, Caselli and Pernul [14] also noted that CACAO playbooks rely on JSON serialization and currently lack integrated validation schemas, which can complicate automation. However, more recent developments have addressed this limitation. The OASIS CACAO Technical Committee has since published machine-readable JSON Schema files that define the structure of valid CACAO playbooks [10]. These schemas enable validation of generated playbooks using standard tooling, enhancing consistency and reducing errors.

Additionally, Goessner [6] introduced JSONPath, a query language for JSON data structures. It allows for flexible extraction and transformation of data. Given that CACAO and multiple enterprise-specific playbooks use JSON format, such tools are necessary to programmatically read and convert playbooks across different formats.

Together, these works provide a solid foundation for understanding current practices in incident response automation, the technical foundation of the CACAO standard, as well as relevancy and the practical challenges of standardizing incident response playbooks.

4 PRELIMINARY ANALYSIS

In this section, we discuss the results obtained from the comparative analysis of different playbook vendors.

On a general level, as mentioned in the research of Schlette et al. [15][14], playbooks of most vendors share core properties. Since all playbooks are designed with a goal of storing courses of actions for various scenarios, most playbook formats have objects that represent a *workflow* which consists of *steps* of different types. Furthermore, Schlette et al. outlined several general step types that are shared by most playbooks: *start*, *end*, *action*, *condition*, *loop*, *parallel* and *sub-playbook*.

Although all playbooks contain general metadata fields such as name and description on a single step and playbook levels, we discovered significant discrepancies when attempting to map the logic and functionality of steps to CACAO format. One of the main issues observed was that all four formats are designed to be integrated with the specific software that executes them. These playbooks assume some form of context which they use to retrieve and write data. We discovered a reoccurring pattern of concrete definitions of actions, such as commands or API calls, being defined outside the playbook, while playbook steps only reference these definitions and provide parameters. Therefore, step data gathered only from the playbook file is often not enough to define a tangible step in CACAO format.

Similarly, for some agents, targets and variables involved in the playbook execution can be inferred only from their references within the playbook, as actual definitions and assigned values are often not included in the playbook.

Another common issue encountered in all four formats is the conversion of conditions to STIX patterning grammar, required by the CACAO standard. Firstly, the condition specification strategy is different for each playbook format. Secondly, playbooks often involve variables from the assumed context which, given only the playbook file, is inaccessible. Furthermore, we have observed cases where a condition step has multiple condition statements and more than two outgoing connections. Such step is to be interpreted as a multi-branch conditional statement. Since CACAO only supports binary conditional statements, a single multi-branch statement would need to be converted into multiple nested binary statements.

For all formats analysed, mapping sub-playbook steps to CACAO steps is challenging because the CACAO standard requires each sub-playbook to also be in CACAO format. As a result, sub-playbooks in proprietary formats must also be converted into valid CACAO playbooks.

The issues outlined above make the task of converting vendor-specific playbook formats to fully functional CACAO playbooks highly complex. Nevertheless, the overall playbook structure, step types and some step functionality can still be mapped. The following subsections present the mapping strategies developed through comparison of the CACAO format with each of the four selected vendors.

4.1 Cortex XSOAR and Palo Alto Demisto

Cortex XSOAR playbooks are essentially extended iterations of the original Demisto playbooks, since Cortex XSOAR utilizes Demisto as the foundational framework for its playbooks. Since the structure and step types of the two formats are highly similar, in this subsection we refer to both formats as Cortex XSOAR. This playbook format represents a playbook step as *task* and workflow as *tasks*.

Step type mapping. All Cortex XSOAR steps contain a `"type"` field. This playbook format has a distinct type for start steps. All action steps have a type `"regular"`, however there is no distinct type for parallel steps, although it can be deduced if an action step has more than one outgoing connection. Furthermore, this format includes steps of type `title` which are used to partition the playbook into segments, as well as to indicate the end of playbook execution. A step of such type can be considered as an action or end step, depending on the amount of outgoing connections. Condition steps in Cortex XSOAR are of type `"condition"`. This format has no loop step; instead it utilizes pre-configured sub-playbooks. Sub-playbook steps have type `"playbook"`.

Step connection mapping. In Cortex XSOAR playbooks, each step contains a list of identifiers of the following steps, which can be found in `"nexttasks"` object. For condition steps, condition outcomes are labelled and `"nexttasks"` contains a list of following steps for every label.

4.2 Splunk SOAR

Playbooks of Splunk SOAR format store step information in a playbook *cell*.

Step type mapping. Splunk SOAR steps are typed, although this format utilizes significantly more distinct step types than CACAO. Both start and end steps have the same type `"coa.StartEnd"`, however they can be distinguished by the amount of outgoing connections. Condition steps are represented by types `"coa.Filter"` and `"coa.Decision"`. Splunk SOAR playbooks support executing other playbooks within a playbook, such steps have type `"coa.CallPlaybook"`. The rest of the types encountered in this format represent some form of action or, in case the number of outgoing connections exceeds 1, a parallel action.

Step connection mapping. One of the main distinctions of Splunk SOAR playbooks is the fact that connections between steps are also playbook cell objects with type `"link"`. They are located inside the same `"cells"` object as the actual playbook steps. Such object of type `"link"` contains a source and target ID of a single connection, which is a major difference from CACAO's `"on_completion"` property defined within a step object.

4.3 FortiSOAR

In FortiSOAR playbooks, playbook step list can be found under `"steps"` object.

Step type mapping. Even though a `"stepType"` property is present for each ForiSOAR playbook step, it does not contain a type identifier, but an API endpoint, which retrieves a type based on the provided step type ID. Because of this feature, direct mapping of FortiSOAR step `"stepType"` values to CACAO types is challenging. However, some step types can be inferred from other properties of a step. For instance, we observed that condition steps can be identified if they include `"arguments"` object, which either contains `"type"="DecisionBased` or has `"conditions"` object present. Start steps can be recognized if a step has no incoming connections or, alternatively, steps with no outgoing connections

can be mapped to end steps in CACAO playbooks. Steps that contain "for_each field inside the "arguments" object can be interpreted as loop steps.

Step connection mapping. FortiSOAR playbooks define step connections outside in "routes", a list defined outside "steps" object. Each element of "routes" contains "sourceStep" and "targetStep" fields. Notably, these fields contain API endpoints to steps of the form "/api/3/workflow_steps/<STEP_ID>". Therefore, in order to map the connections, the step IDs first need to be extracted from these endpoints.

5 PROTOTYPE DESIGN

This section describes the architecture of the prototype playbook translation tool developed using Python programming language. The tool is intended to be a unified solution accommodating the intrinsic differences between four analysed playbook formats.

Our chosen approach was to isolate the vendor-specific information extraction logic to a *map* - a JSON schema that defines how to extract key fields from a specific vendor's playbook. Thus, adapting the system to other playbook formats requires only the creation of a corresponding map file, without any modifications to the code base. It also allows for potential further generalization of the system through automatic generation of map files for an arbitrary playbook format.

Given that a valid playbook-map file pair is provided, the tool parses the playbook and extracts or derives information that is necessary to produce a valid CACAO playbook file.

5.1 Playbook map

The map JSON file contains paths to various fields in the vendor-specific playbook format. Paths are specified in JSONPath [6] format. Listing 1 shows an example of such map file.

At the root level, the map must include the following flat key-value pairs:

- "name" - a path to the playbook name,
- "description" - a path to the playbook description,
- "steps" - a path to the dictionary or list containing the steps of the playbook,
- "edges" - a path to the dictionary or list containing the definitions of edges between steps.

Moreover, the map file must also contain two key-object pairs - "edge" and "step" - each instructing the system how to act on each item found at paths specified, respectively, for "edges" and "steps" keys of the map. For both "edge" and "step" objects, all paths specified are expected to be relative to a single respective item.

- "edge" object must contain fields "source" and "targets", both of which expect string paths that specify locations for source and target IDs of a connection.
- "step" object must include paths for step-level fields such as "id", "name", and "description".

Filters. Filters are objects containing logical conditions, structured as comparisons between a specified path and an expected value, or combinations of such comparisons using logical operators. In the "edge" object of the map, "edgeFilter" can be defined - a rule dictating which items from the edge list to include in the processing. "steps" object can contain filters for generic steps as well as steps of specific types.

By design, filters are optional, since they may not be necessary for translating some vendor-specific playbook formats, although specifying them can greatly increase the precision of playbook translation.

Listing 1. FortiSOAR map file

```
{
    "name": "$.name",
    "description": "$.description",
    "steps": "$.steps",
    "edges": "$.routes",
    "edge": {
        "source": "$.sourceStep",
        "targets": "$.targetStep"
    },
    "step": {
        "id": "$.uuid",
        "name": "$.name",
        "description": "$.description",
        "conditionStepFilter": {
            "or": [
                {
                    "equals": {
                        "path": "$.arguments.conditions",
                        "function": "exists",
                        "value": true
                    }
                },
                {
                    "equals": {
                        "path": "$.arguments.type",
                        "value": "DecisionBased"
            ]
        }
    }
}
```

5.2 Playbook parsing

The parser first extracts global properties such as the playbook name and description. Then it processes the step definitions based on the paths and filters specified in the map. For each valid step, the parser collects the necessary attributes such as identifiers, names, and descriptions, and uses map filters to classify the step types. Steps that fail to be categorized by any specific step type filters are marked as undefined steps to be handled later in the translation process. Subsequently, the parser identifies control flows between steps by extracting source and target identifiers from the mapped fields.

All extracted and classified elements are stored in an intermediate representation object, ready for validation and eventual transformation into a playbook compliant with the CACAO standard.

5.3 Playbook correction

The playbook correction component transforms an initially parsed workflow into a valid CACAO playbook object. This process begins by inferring the appropriate CACAO type for each step marked as undefined based on its position within the workflow. The algorithm converts undefined steps to

- start steps, if there are no incoming connections,
- end steps, if there are not no outgoing connections,

- parallel steps, if there are incoming connections and more than one outgoing connection,
- action steps, if no above-mentioned criteria were met.

Moreover, this component alters the structure of the workflow with the goal of complying with the CACAO standard. For example, if a step is classified as a start step, but contains multiple outgoing edges, then the algorithm inserts a dedicated parallel step between a start step and all of its neighbours. Similarly, condition steps that branch into more than two paths are recursively nested, as the CACAO format expects conditional steps with binary structure. Where action steps are identified without any successors, the system appends a required end step.

Throughout this refinement, the system also ensures that each step contains the mandatory fields required by the standard, such as agent definitions, execution commands, as well as necessary control flow fields such as `on_true`, `on_false` for condition steps, or `on_completion` for action steps.

Upon completion, the system adds the required metadata to the CACAO object, including CACAO-compliant unique identifiers and timestamps. The CACAO playbook can then be exported as a structured JSON document or visualised as a directed graph for further analysis or validation.

6 EVALUATION

To assess the effectiveness of the proposed translation tool, we conducted a series of experiments focusing on the validity, completeness, and fidelity of the generated CACAO playbooks. This section presents the dataset used, the evaluation metrics applied, and the results obtained from translating playbooks across four different SOAR platforms.

6.1 Dataset

We used a GitHub repository [9] of publicly available incident response playbooks as our evaluation sample. The final dataset used for the experiments consisted of 463 playbooks, including 81 Splunk SOAR playbooks, 197 FortiSOAR playbooks, 127 Cortex XSOAR playbooks, and 58 Palo Alto Demisto playbooks.

6.2 Metrics

In this subsection we introduce metrics developed to evaluate the completeness and fidelity of translated playbooks.

6.2.1 Completeness. We measured the completeness of the translated CACAO playbooks by first splitting the root-level CACAO playbook fields into several categories:

- *Playbook Core* - essential playbook fields, such as playbook name and workflow object,
- *Tracking and Validity* - various timestamps and signatures,
- *Descriptive Fields* - properties that provide information about the playbook's type, purpose and involved operations,
- *Definitions* - fields that are used to define data such as agents, targets, variables involved in the execution of a playbook,
- *Values* - properties that facilitate assignment of values to the previously defined fields,
- *Metrics* - priority, severity and impact scores of the playbook.

The specific fields that were assigned to each category can be observed in Appendix A. For each field category c we calculated Completeness Score (CS) in the following way:

$$CS_c = \frac{\#\text{ of } c \text{ fields present in translated playbook}}{\text{expected } \#\text{ of } c \text{ fields}}$$

It is a fraction of category fields that were present in the output playbook, ranging within $[0,\ 1]$.

6.2.2 Fidelity. To measure the fidelity of playbook translation, we introduced three metrics: Node Count Ratio (NCR), Node-Adjacency Jaccard Similarity ($NAJS$) and Type-Adjacency Cosine Similarity ($TACS$).

Node Count Ratio. NCR represents the size difference between the input and output playbook. It is calculated the following way:

$$NCR = \frac{\#\text{ of steps in CACAO playbook}}{\#\text{ of steps in original playbook}} - 1$$

NCR value above zero (indicating that steps were added to CACAO playbook) or below zero (indicating that steps were skipped during translation) points to structural differences between playbooks.

To compute $NAJS$ metric, original and translated CACAO playbooks are converted to directed graphs, with nodes identified by original playbook step IDs. Nodes of the translated CACAO playbook graph maintain the mapping to original playbook step IDs, while the edges represent the structure after translation. Any steps that were added in the process of translation are given a mock ID that does not overlap with any of the original playbook step IDs.

Node-Adjacency Jaccard Similarity. NAJS compares adjacency sets of original and translated playbooks. We defined adjacencies as ordered tuples of original step IDs:

$$(u, v) \text{ where step } u \to v$$

Then, adjacency sets are:

$$A_o = \{(u, v) \mid u \to v \text{ in original playbook}\}$$
$$A_c = \{(u', v') \mid u' \to v' \text{ in CACAO playbook}\}$$

Finally, the $NAJS$ metric is the Jaccard index of both adjacency sets:

$$NAJS = \frac{|A_o \cap A_c|}{|A_o \cup A_c|}$$

Value of this metric ranges within $[0,\ 1]$ and measures how well the control flow in the original playbook is preserved in the CACAO translation. $NAJS = 1$ means exact structural match of the two playbooks, whereas $NAJS = 0$ indicates no structural overlap.

Type-Adjacency Cosine Similarity. TACS also processes graph representations of original and translated playbooks. Before calculation, every node in both input and output playbooks is mapped to one of the semantic types, defined in Section 4: *start*, *end*, *action*, *parallel*, *condition*, *loop* or *playbook*. Then, $TACS$ metric collects type adjacency tuples, defined in the following way:

$$(type(u), type(v)) \text{ where step } u \to v$$

For each graph, all type adjacencies are collected in a multiset. Next, a combined vocabulary of all observed type pairs is constructed, after which each playbook's list of type adjacencies is

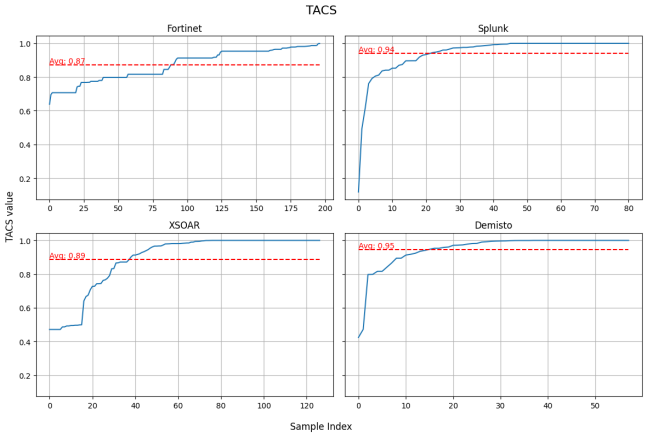

Fig. 1. Distributions of *TACS* scores for 4 playbook vendors

converted into a frequency vector over this vocabulary. *TACS* is then calculated the following way:

$$TACS = \frac{\vec{u} \cdot \vec{v}}{||\vec{u}|| \times ||\vec{v}||} \quad \text{,where}$$

$\vec{u}$ is a frequency vector of type adjacencies in original playbook

$\vec{v}$ is a frequency vector of type adjacencies in CACAO playbook

TACS is the cosine of the angle between the two type adjacency frequency vectors, ranging within $[0,\ 1]$. $TACS = 1$ indicates that type-to-type transitions of original and translated playbooks are identical and $TACS = 0$ signifies that the transitional patterns of playbooks do not overlap.

Unlike Jaccard index, *TACS* reflects how often transitions occur, not just whether they exist. Due to various constraints posed by CACAO standard, playbook structure may be changed, even though the translated playbook may remain semantically equivalent to the original playbook. *TACS* allows assessing semantic similarity of the overall workflow behaviour and logic, abstracting away from the exact node-to-node structure.

6.3 Experiments

We conducted multiple experiments on validity, completeness and translation quality of the CACAO playbooks produced by the prototype tool.

6.3.1 CACAO validity. To automate validation of the translated CACAO playbooks, we utilized `jsonschema` Python library which offers JSON object validation based on a JSON schema. We used the official CACAO playbook validation schema [10] uploaded by OASIS. After verifying the validity of all 463 playbooks post-translation, the tool achieved a 100% validity rate.

6.3.2 CACAO completeness. To get an overview of CACAO playbook completeness, for each translated playbook we computed completeness scores of every playbook field category specified in Appendix A, after which we determined the range of the completeness values for each format. Resulting values are summarized in Table **??**

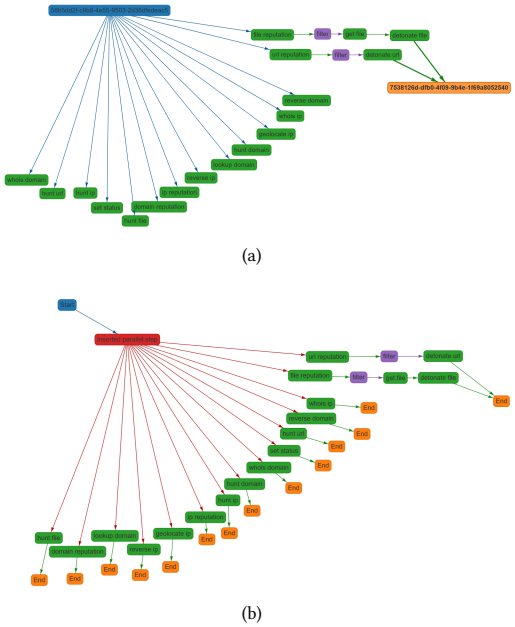

Fig. 2. Splunk playbook with $TACS = 0.12$ and $NAJS = 0.14$ before (a) and after (b) translation. To comply with CACAO standard, the tool appends an end step (orange) to all action steps (green) with no outgoing connections and inserts a parallel step (red) after start step (blue).

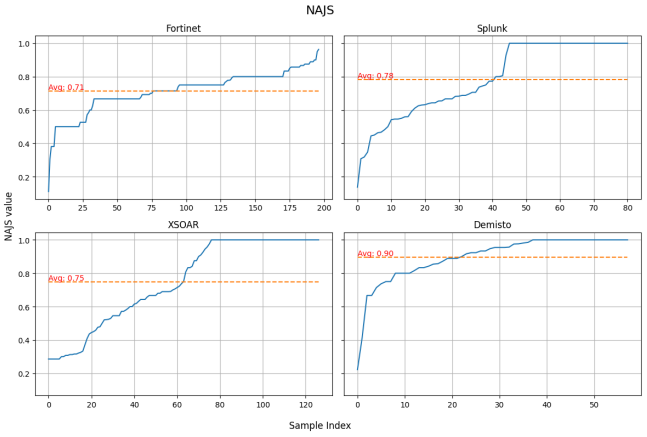

Fig. 3. Distributions of *NAJS* scores for 4 playbook vendors

For "Playbook Core" category, our tool consistently achieved 85.6% completeness, missing only `"workflow_exception"`, for which we were not able to find an equivalent in other playbook formats.

For "Tracking and Validity" fields, the tool reaches a constant 42.85% completeness score, since it only inserts `"created_by"`, `"created"` and `"modified"` properties due to them being required by the CACAO standard. The rest of the fields in this category are designed to be filled after the playbook becomes fully operational.

Completeness score of "Descriptive Properties" category ranges from 0% to 11%, depending on whether a description value is detected in the original playbook. Remaining category properties are specific to CACAO format and present significant challenges when attempting to map them to values in proprietary playbooks.

Similarly, mapping "Definitions" category fields is a complex problem, as non-CACAO playbooks define agents, targets and variables in highly dissimilar ways. Thus, completeness score for this category is 20%, as only `"agent_definitions"` field contains a definition of a placeholder agent.

Because the tool is not capable of mapping definitions, "Values" category fields remain unfilled with a completeness score of 0%.

"Metrics" category contains CACAO-specific properties that have no mapping in any of the four analysed playbooks. Consequently, the completeness score for this category is also 0%.

Category	Completeness
Playbook Core	85.6%
Tracking and Validity	42.85%
Descriptive Properties	0% to 11%
Definitions	20%
Values	0%
Metrics	0%

Table 1. Translated playbook completeness

6.3.3 Translated playbook similarity. To draw conclusions on the fidelity retention after playbook translation, we programmatically compared original and translated playbooks, collecting similarity metrics *NCR*, *NAJS* and *TACS* in the process.

The average *TACS* scores for all four playbook formats range from 0.87 to 0.95, suggesting that most translated playbooks preserve the semantic transitions between node types. Palo Alto Demisto and Splunk SOAR attain the highest *TACS* values (0.95 and 0.94, respectively), indicating highly consistent behaviour at the type level between the original and CACAO formats. Figure 1 shows the *TACS* scores of all 4 playbook vendors in ascending order. Notably, the figures show that relatively small fractions of playbooks have *TACS* scores significantly below the average value. The most distinct drop is observable among Splunk SOAR playbooks. Figure 2 demonstrates the large amount of changes made by our tool to make the playbook valid for the CACAO standard, which caused *TACS* score to plummet.

The results show that on average our tool achieves high structural similarity. Figure 3 demonstrates collected *NAJS* scores of 4 playbook types with means between 0.71 (Fortinet) and 0.90 (Demisto). Notably, Demisto exhibits the highest structural retention, indicating that most relationships between original steps were preserved in the CACAO version. Since *NAJS* score tracks node-to-node relations based on node IDs, it is strongly affected by the structural changes made to comply with the CACAO standard. Figure 4 demonstrates how the *NAJS* and *TACS* scores are affected by the additions made by our tool.

Across the four playbooks analysed, average *NCR* values range from 0.09 to 0.30, as seen in Figure 5. This indicates that the CACAO

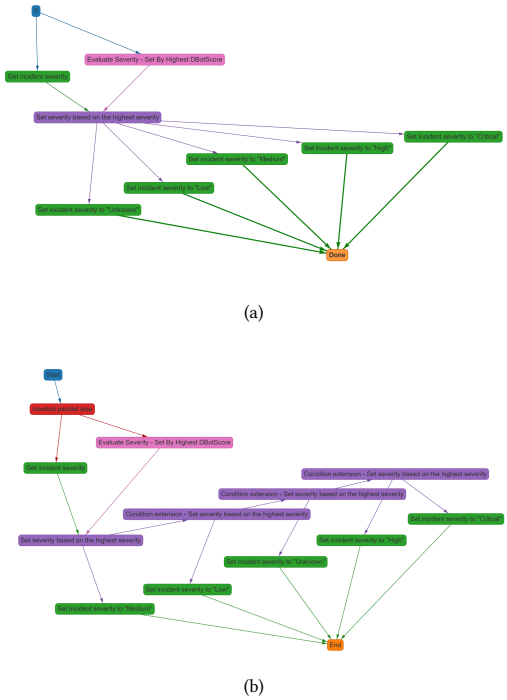

Fig. 4. Cortex XSOAR playbook with relatively high $TACS = 0.88$ and low $NAJS = 0.33$ before (a) and after (b) translation. The tool converted a multi-branch condition step (purple) into a series of nested single-branch condition steps, thus changing the id-to-id relationships, but preserving most type-to-type connections

translations are consistently larger than the original playbooks. The Fortinet format exhibits the highest relative increase in node count ($NCR = 0.30$), suggesting that during translation the tool introduces more intermediate or final steps than with other formats. In contrast, the Splunk, XSOAR, and Demisto playbooks yield relatively modest size increases, with average *NCR* being around 0.10, reflecting more compact translations.

7 LIMITATIONS

The approach and prototype developed in this research have several limitations, which should be addressed in future work.

Firstly, scope of our research included an analysis of only four playbook vendors (Cortex XSOAR, Splunk SOAR, FortiSOAR, and Palo Alto Demisto). While these vendors represent a significant portion of the market, other SOAR platforms, especially smaller or emerging vendors, were not considered. This limited scope may not capture the full diversity and complexity of vendor-specific playbook implementations.

Secondly, although our developed system is capable of converting general control flow and capturing some metadata, resulting CACAO playbooks lack the detailed step functionality. The conversion does not fully reflect the nuances of each vendor-specific playbook step, potentially reducing the fidelity of translated playbooks.

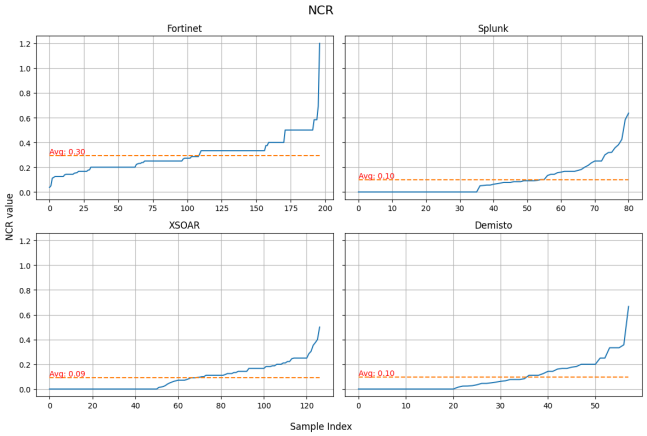

Fig. 5. Distributions of *NCR* scores for 4 playbook vendors

Furthermore, another limitation of the current approach is its inability to extract agents and variables from playbooks. Although the preliminary analysis revealed the significant complications of mapping agents and variables, mainly caused by limited contextual information, the resulting incapacity to extract agents and variables may result in a loss of important contextual information.

In addition, the tool does not parse or assign meaningful courses of action for conditional branches. Specifically, it is unable to interpret the intended outcomes when conditions evaluate as true or false, leading to a random control flow that must be corrected manually before the resulting CACAO playbook can be considered for exploitation.

Finally, the translation process is still largely dependent on manually created maps between vendor-specific formats and the CACAO standard. While the current system is capable of inferring certain playbook step types, it relies heavily on maps to parse the overall playbook control flow. This approach limits the flexibility and automation of the tool, especially when trying to scale or adapt to new or unknown playbook formats.

7.1 Future work

The identified limitations point to several potential directions for future research.

Expanding the vendor support. Future work should include additional SOAR vendors and proprietary playbook formats to validate the scalability and robustness of the translation tool across a wider range of platforms. This would also involve evaluating the tool with smaller vendors and open-source playbooks to ensure a more generalized solution.

Enhancing playbook translation fidelity. Developing a more detailed mapping mechanism that captures not only the general control flow but also the specific functionalities of each step is essential to minimize the amount of manual corrections that must be applied before the resulting CACAO playbook is ready for execution. This includes capturing complex workflows, external system interactions, custom actions, as well as extracting agents, targets, and variables. This would significantly improve the fidelity of translated playbooks, as these elements are critical for the execution of actions within the playbooks.

Moreover, future research could focus on maintaining logical consistency in the translated playbooks through more accurate mapping of multi-branch conditions and converting of condition statements to equivalents in STIX Patterning Grammar, potentially by employing Natural Language Processing (NLP) models.

Generalizing the translation process. A fully generalized approach to the translation process is still yet to be developed. Machine learning or AI techniques could be leveraged to automatically generate mappings between playbook formats. This would eliminate the need for manual map creation and improve the tool's adaptability to new formats or changes in existing playbook structures.

8 CONCLUSION

The goal of our research was to explore how enterprise-specific incident response playbooks can be systematically translated into the CACAO standard, focusing on four widely used SOAR platforms: Cortex XSOAR, Splunk SOAR, FortiSOAR, and Palo Alto Demisto. Through a detailed comparative analysis of their playbook structures, we uncovered commonalities such as shared generic step types and basic workflow logic, as well as significant differences in metadata and steps connection representation, conditional logic, and external dependencies.

To address the main **RQ**, we designed a prototype translation tool based on the results of the preliminary analysis. The tool uses configurable mapping files to isolate vendor-specific parsing logic, enabling it to extract relevant information and restructure it into a CACAO-compliant format.

To answer **SRQ1**, the evaluation showed that the translated playbooks closely reflect the overall control flow of the originals. This was demonstrated by high similarity scores in both structural and semantic metrics. However, the tool was unable to capture detailed step functionality, external agent definitions and variable usage, thus necessitating manual refinement of converted playbooks before operational deployment.

SRQ2 focused on the generalizability of the translation process. By separating format-specific knowledge into external JSON map files, the system enables support for additional SOAR platforms without any changes to the code base. However, full generalization remains an open challenge. The diversity in how vendors implement playbooks means that some formats may require significantly more complex mapping rules or additional preprocessing to expose hidden semantics.

To conclude, in this research we present a functional, extensible solution to transferring the overall structure and control-flow logic of incident response playbooks into comprehensible and valid playbooks of CACAO format. While not without limitations, our work lays a foundation for future improvements in fidelity, coverage, and automation.

REFERENCES

[1] Jeff Bollinger, Brandon Enright, and Matthew Valites. 2015. *Crafting the InfoSec Playbook: Security Monitoring and Incident Response Master Plan.* Accessed: 2025-04-30. O'Reilly Media. ISBN: 978-1491949405. https://www.amazon.com/Crafting-InfoSec-Playbook-Security-Monitoring/dp/1491949406.

[2] Xander Bouwman, Victor Le Pochat, Pawel Foremski, Tom Van Goethem, Carlos H. Gañán, Giovane C. M. Moura, Samaneh Tajalizadehkhoob, Wouter Joosen, and Michel van Eeten. 2022. Helping hands: measuring the impact of a large threat intelligence sharing community. In *31st USENIX Security Symposium (USENIX Security 22)*. Accessed: 2025-04-30. USENIX Association, Boston, MA, 1149–1165. https://www.usenix.org/conference/usenixsecurity22/presentation/bouwman.

[3] Philip Empl, Daniel Schlette, Lukas Stöger, and Günther Pernul. 2023. Generating ics vulnerability playbooks with open standards. *International Journal of Information Security*, 23, 1, 1–20. Accessed: 2025-04-30. DOI: 10.1007/s10207-023-00760-5.

[4] European Union Agency for Cybersecurity (ENISA). 2023. ENISA Threat Landscape 2023. Tech. rep. Accessed: 2025-04-30. European Union Agency for Cybersecurity, (Oct. 2023). https://www.enisa.europa.eu/publications/enisa-threat-landscape-2023.

[5] Fortinet. 2025. Fortisoar: security orchestration, automation, and response platform. https://www.fortinet.com/products/fortisoar. Accessed: 2025-05-08. (2025).

[6] Stefan Goessner and Carsten Bormann. 2020. JSONPath—XPath for JSON. Internet-Draft draft-goessner-dispatch-jsonpath-00. Internet-Draft; expires Jan 13, 2021; work in progress; Accessed: 2025-06-22. IETF DISPATCH Working Group, (July 2020).

[7] Mehdi Akbari Gurabi, Avikarsha Mandal, Jan Popanda, Robert Rapp, and Stefan Decker. 2022. Sasp: a semantic web-based approach for management of sharable cybersecurity playbooks. In *Proceedings of the 17th International Conference on Availability, Reliability and Security (ARES 2022)*. Accessed: 2025-04-30. Association for Computing Machinery, New York, NY, USA. DOI: 10.1145/3538969.3544478.

[8] Marc-André Kaufhold, Thea Riebe, Markus Bayer, and Christian Reuter. 2024. We do not have the capacity to monitor all media: a design case study on cyber situational awareness in computer emergency response teams. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*. Best Paper Award; Accessed: 2025-04-30. Association for Computing Machinery, New York, NY, USA, 580:1–580:16. DOI: 10.1145/3613904.3642368.

[9] Ludus Librum. 2023. Awesome playbooks: a curated repository of incident response playbooks. https://github.com/luduslibrum/awesome-playbooks. Accessed: 2025-04-30. (2023).

[10] OASIS CACAO TC Open Repository. 2024. Cacao-json-schemas: json validation schemas for cacao security playbooks. https://github.com/oasis-open/cacao-json-schemas. Apache-2.0 license; Accessed: 2025-06-22. (2024).

[11] OASIS Open. 2023. Cacao security playbooks version 2.0. Accessed: 2025-04-30. (2023). https://docs.oasis-open.org/cacao/playbooks/v2.0/cacao-playbooks-v2.0.html.

[12] Palo Alto Networks. 2025. Cortex xsoar: extended security orchestration, automation, and response. https://www.paloaltonetworks.com/cortex/cortex-xsoar. Accessed: 2025-05-08. (2025).

[13] Palo Alto Networks. 2025. Demisto: security orchestration, automation, and response platform. https://apps.paloaltonetworks.com/marketplace/demisto. Accessed: 2025-05-08. (2025).

[14] Daniel Schlette, Marco Caselli, and Günther Pernul. 2021. A comparative study on cyber threat intelligence: the security incident response perspective. *IEEE Communications Surveys & Tutorials*, 23, 4, 2525–2556. Accessed: 2025-04-30. DOI: 10.1109/COMST.2021.3090902.

[15] Daniel Schlette, Philip Empl, Marco Caselli, Thomas Schreck, and Günther Pernul. 2024. Do you play it by the books? a study on incident response playbooks and influencing factors. In *Proceedings of the 2024 IEEE Symposium on Security and Privacy (S&P)*. Accessed: 2025-04-30, 3625–3643. DOI: 10.1109/SP54263.2024.00060.

[16] Splunk Inc. 2025. Splunk soar: security orchestration, automation, and response. https://www.splunk.com/en_us/products/splunk-security-orchestration-and-automation.html. Accessed: 2025-05-08. (2025).

[17] Rock Stevens, Daniel Votipka, Josiah Dykstra, Fernando Tomlinson, Erin Quartararo, Colin Ahern, and Michelle L. Mazurek. 2022. How ready is your ready? assessing the usability of incident response playbook frameworks. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*. Accessed: 2025-04-30. Association for Computing Machinery, New York, NY, USA. DOI: 10.1145/3491102.3517559.

[18] European Union. 2022. Directive (eu) 2022/2555 of the european parliament and of the council of 14 december 2022 on measures for a high common level of cybersecurity across the union (nis2 directive). Accessed: 2025-04-30. (2022). https://eur-lex.europa.eu/eli/dir/2022/2555/oj.

Appendices

A CACAO PLAYBOOK FIELD GROUPINGS

Group	Fields
Playbook Core	type spec_version id name workflow workflow_start workflow_exception
Tracking and Validity	created_by created modified revoked valid_from valid_until signatures
Descriptive Properties	description playbook_types playbook_activities playbook_processing_summary industry_sectors related_to derived_from external_references labels
Definitions	authentication_info_definitions agent_definitions target_definitions extension_definitions data_marking_definitions
Values	playbook_variables markings playbook_extensions
Metrics	priority severity impact