



Sharing and Automation for  
Privacy Preserving Attack Neutralization

(H2020 833418)

**D4.7 Algorithm to automate recommended response and  
recovery actions without human operators, final version  
(M30)**

**Published by the SAPPAN Consortium**

**Dissemination Level: Public**



**H2020-SU-ICT-2018-2020 – Cybersecurity**

## Document control page

<b>Document file:</b>	D4.7 Algorithm to automate recommended response and recovery actions without human operators, final version
<b>Document version:</b>	1.0
<b>Document owner:</b>	Mehdi Akbari Gurabi (FIT)
<b>Work package:</b>	WP4
<b>Task:</b>	T4.4 Develop approach to automate response and recovery actions without human operators
<b>Deliverable type:</b>	Other
<b>Delivery month:</b>	M30
<b>Document status:</b>	<input checked="" type="checkbox"/> approved by the document owner for internal review <input checked="" type="checkbox"/> approved for submission to the EC

### Document History:

Version	Author(s)	Date	Summary of changes made
0.1	Mehdi Akbari Gurabi (FIT), Avikarsha Mandal (FIT), Lasse Nitz (FIT)	2021-08-27	Document outline
0.2	Mehdi Akbari Gurabi (FIT), Pavel Eis (CESNET), Martin Lastovicka (MU), Mischa Obrecht (DL), Jan Popanda (FIT)	2021-10-22	First draft
0.3	Mehdi Akbari Gurabi (FIT), Avikarsha Mandal (FIT)	2021-10-26	Ready-to-review version
0.4	Mehdi Akbari Gurabi (FIT), Milan Cermak (MU), Richard Kalinec (MU), Avikarsha Mandal (FIT), Mischa Obrecht (DL), Jan Popanda (FIT), Sebastian Schäfer (RWTH)	2021-10-29	Updated version with Incorporated feedback
1.0	Mehdi Akbari Gurabi (FIT)	2021-10-29	Final version

### Internal review history:

Reviewed by	Date	Summary of comments
Milan Cermak (MU)	2021-10-27	Grammar and spelling corrections
Sebastian Schäfer (RWTH)	2021-10-28	Comments regarding technical details and wording
Richard Kalinec (MU)	2021-10-29	Comments regarding technical details, grammar and spelling corrections

## Executive Summary

This deliverable D4.7 is the final version of task T4.4, where we develop and evaluate SAPPAN approaches for automatic response and recovery steps without the involvement of human operators. As there is a shortage of human experts to analyze a large volume of false alarms generated in the detection phase, operators may miss true alerts due to fatigue or inexperience or simply for the late response. The objective of the automation task is to propose and evaluate approaches that take away some of the responsibilities of human operators and rapidly react to potential attacks in real-time. Further, identifying and reducing the damage, which the false response at the wrong time can cause, is very important. In this SAPPAN task T4.4, we contribute towards automating some response actions depending on the risk and confidence metrics associated with the detection, the asset involved, the severity and impact of an incident, and the corresponding response.

In the preliminary version of T4.4 (D4.6), we outlined the necessary background in response automation and presented a framework that can capture the approaches for automating some types of responses actions depending on several factors. We considered two showcases from WP3 (phishing and DGA) that have moderate risks for response actions and discussed possible steps of automation.

In this final version D4.7, we continue our previous work by extending it to three showcases (phishing, DGA, and malware analysis), implementing and evaluating respective response automation prototypes. We first present cybersecurity playbook specifications and standardization activities in line with response automation. We mention SAPPAN contributions in this scope. Further, we overview workflow automation and review currently available automation tools. We detail our contributions in regards to workflow automation. Next, we present our prototypes for response automation, including the prototypes for phishing and DGA showcases and a malware response automation platform. We discuss our lessons learned, further considerations and preliminarily evaluate the results of the automation platform. In conclusion, further evaluation plans and future activities are stated in line with this task.

## Table of Contents

<b>Executive Summary .....</b>	<b>3</b>
<b>1 Introduction .....</b>	<b>6</b>
<b>2 Cybersecurity Playbook Specification for Automated Response Actions..</b>	<b>8</b>
2.1 Overview of the CACAO Playbook Specification .....	8
2.1.1 Playbook Structure.....	8
2.2 Overview of the SAPPAN Playbook Specification and Vocabulary .....	10
2.3 SAPPAN Vocabulary vs. CACAO Specification.....	10
2.4 SAPPAN Contributions in CACAO Standardization Activities.....	11
2.4.1 Feedback on CACAO .....	11
2.4.2 Automatic CACAO Validation Utility.....	11
2.4.3 A MISP Data Model to Share CACAO Playbooks .....	12
2.4.4 CACAO Playbooks Translator .....	12
<b>3 Workflow Automation .....</b>	<b>13</b>
3.1 Security Orchestration, Automation and Response .....	13
3.2 Available Automation Tools.....	13
3.2.1 Current Situation .....	13
3.2.2 Analysis.....	14
3.2.3 Frameworks .....	14
3.3 Using Apache Airflow for Workflow Automation.....	19
3.3.1 Workflow Model .....	19
3.3.2 Implementation of Workflow Steps .....	19
3.3.3 Automated Workflow Execution .....	20
3.3.4 Built-in Visualization and Evaluation .....	20
3.3.5 Concerns and Feedback.....	22
3.4 CACAO Playbooks Translator into Airflow DAGs .....	22
<b>4 SAPPAN Response Automation Prototypes .....</b>	<b>23</b>
4.1 Overview .....	23
4.1.1 Requirements.....	24
4.2 Phishing Showcase Prototype .....	25
4.2.1 Workflow .....	25
4.2.2 Decision Formula and Case Evaluation.....	29
4.2.3 Interfacing .....	29
4.2.4 Example Run .....	30

4.2.5	Outlook.....	32
4.3	DGA Showcase Prototype .....	33
4.3.1	System Architecture .....	33
4.3.2	Workflow Description .....	35
4.3.3	Illustrations .....	37
4.3.4	Lessons learned.....	41
4.4	Malware Response Automation Platform Prototype .....	45
4.4.1	Malware Evaluator Workflow and Architecture .....	45
4.4.2	Deployment and Configuration .....	47
4.4.3	System Usage Demonstration .....	48
4.4.4	Evaluation of Automation Impact on Response .....	51
<b>5</b>	<b>KPI Evaluation .....</b>	<b>55</b>
<b>6</b>	<b>Conclusion.....</b>	<b>56</b>
<b>7</b>	<b>References.....</b>	<b>57</b>

## 1 Introduction

In the cybersecurity incident response and recovery process, it is an extremely challenging task to efficiently respond against a large number of cyberattacks due to a lack of experienced security analysts. The cybersecurity industry is moving towards developing and deploying new tools that can help less experienced operators with incident response and recovery. Furthermore, the repetitive nature of many response actions for a particular genre of threats introduces our research question: To what extent can we perform suitable response and recovery activities autonomously without any human involvement? However, automating response actions is tricky as one must be aware of the risk of automatically performing an action in case of a false positive (especially for high-impact response action with consequences). On the other hand, response automation could potentially benefit operators to save time and mitigate attacks that need to be contained within a narrow timeframe.

Whereas T4.3 deals with the development of incident response recommendation algorithms and providing contextual information to human operators, this task T4.4 investigates approaches for automating response and recovery steps with a number of showcases. In the initial version of the T4.4 (D4.6) deliverable, we gave the necessary background and proposed a framework for automating response and recovery steps for cybersecurity incidents. However, response automation includes risks of incorrect action resulting from missing contextual information and corner cases available to the experts but not considered during automation process design. Hence, it is safer to start with automating steps that have a less severe impact on incorrect decisions. In the first version of the task deliverable, we identified two showcases (Phishing and DGA) that have a moderate risk of response activities in order to discuss the steps of automation. In this final version of T4.4 (D4.7), we extend the work in the line of cybersecurity playbooks, particularly incident response workflow automation for three showcases: i) phishing ii) DGA, and iii) malware analysis.

To begin with the approaches for automation, it is essential to acquire some common knowledge about incident response and recovery steps. In section 2, we give an overview of specifications for cybersecurity playbooks in the context of automated response actions. Along with the SAPPAN playbook specification introduced in T4.1 and T4.2, we further investigate a recently published playbook standard by OASIS CACAO technical committee [CACAO-21]. We then identify commonalities between both specifications, discuss the benefits of using CACAO specification and its limitations, explain how SAPPAN contributes to CACAO developments.

Section 3 presents the background on workflow automation and discusses workflow automation in the context of Security Orchestration, Automation, and Response (SOAR). In recent times, many Security operation centers (SOCs) are integrating SOAR capabilities to allow operators to respond against cyberattacks with pre-defined playbooks. We provide a concise review of existing frameworks and popular efforts to implement SOAR capabilities. Then we give the necessary background for Apache Airflow, an open-source workflow management platform that we used for implementing three workflow automation showcase prototypes. We also include some general concerns that we noticed about Airflow in case anyone wants to implement any cybersecurity response automation. Finally, we detail the translator tool from CACAO format to Airflow DAG and vice versa, which we developed in the course of SAPPAN.

Section 4 includes our efforts in the development of three different response automation prototypes for phishing, DGA, and malware analysis. We outlined design goals and key requirements for implementing our response automation prototypes. In the phishing response showcase, we implement a workflow such that low severity response actions (e.g., blacklist URLs, run malware scan) can be automatically triggered based on the thresholds and the inventory values using a decision formula. In the DGA response showcase, the workflow automation is based on a binary DGA classifier (developed in SAPPAN, integrated into Dreamlab's CySOC solution.), asset importance, and using a case management solution called TheHive/Cortex. Finally, the malware evaluator showcase includes workflow, architecture, and deployment details. The evaluation results show that the workflow automation of the developed malware evaluator significantly reduces the time required for manual analysis of suspicious samples by human analysts. Finally, we list relevant WP4 KPIs and plan to continue the rest of the KPI evaluation as part of WP6 deliverables.

## 2 Cybersecurity Playbook Specification for Automated Response Actions

Shortly before the submission of the SAPPAN vocabulary deliverable (D4.2), OASIS has announced a public specification for security playbooks: The Collaborative Automated Course of Action Operations (CACAO) [1]. The first public version (v.1.0) of the specification was released on 12.01.2021. The specification defines a schema and taxonomy for cybersecurity playbooks. CACAO also considers standardized creation, documentation, and sharing of respective playbooks. We look at the CACAO playbook specification in more detail.

### 2.1 Overview of the CACAO Playbook Specification

A CACAO playbook [1] is a workflow for security orchestration, which consists of a set of steps to describe certain actions. These playbooks may be triggered by an automated or manual event or observation. A playbook should be viewed as a recommended set of steps that guide users or organizations on how to handle certain security events, incidents, problems, or attacks. A playbook may also reference or include other playbooks, which allows the composition of multiple playbooks, from low- to high-level playbooks similar to modular software development.

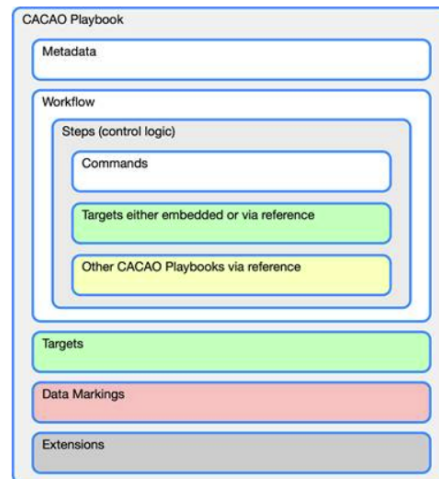
The CACAO specification distinguishes two definitions of playbooks, one being executable playbook, which can be immediately executed (actionable) without any modifications or updates of workflow inside the playbook. The second type of playbooks is the playbook template, which provides examples of actions related to a particular security incident, malware or security operation. Playbook template is not immediately executable but may pass some additional and valuable information to an executable playbook in a specific environment.

Every CACAO playbook may have a different purpose, which is distinguished by playbook type. CACAO specification defines seven different playbook types, and all of them are described in the CACAO specification. Among the most important types, we can include detection playbook (contains workflow required for detection of a known security event, malware, ...) and mitigation playbook with remediation playbook, which also helps with the handling of security incidents, mainly their direct consequences.

#### 2.1.1 Playbook Structure

Every playbook consists of multiple building blocks that altogether define actions and logic, which are performed when the playbook is executed. The playbook metadata contains basic data about the playbook (including mainly the information whether the playbook is executable or just a template), playbook types (every playbook may address multiple operational functions), and the name of the playbook with an overall description of the playbook. This information is needed to give the user of the playbook a quick and useful overview of the playbook's purpose. This basic and mandatory overview can be supplemented with other useful information, such as the playbook's impact, severity, and priority. If a playbook contains some workflow with multiple steps, it is also required to specify the first step, which should be executed via the workflow-start attribute.





**Figure 1: Overview of CACAO playbook Structure. [1]**

Figure 1 shows the playbook structure proposed by the CACAO specification. The main part of the playbook is defined as a **workflow** with multiple steps describing actions that should be taken on one or more targets when some trigger is activated. The trigger itself is generally out of the scope of the playbook, and the playbook should be activated by some automated tool (e.g., IDS system). Every step has some common properties to enable sequential linking of multiple steps as workflow (attributes `on_completion`, `on_success`, ... directly points to next step in workflow), and some other useful attributes to help with automatic processing of the playbook such as timeout or delay of the step.

The CACAO specification defines multiple workflow steps, which can be linked together. Every workflow starts with the *start step* and ends with the *end step*. The main part of a workflow is then constructed with *single steps*, which contain actual commands to be executed. In addition to these basic steps, the CACAO specification further defines the *playbook step*, which allows executing another playbook within the current playbook and other conditional steps (*if*, *while*, *switch*) with also *parallel step*. The variety of combinations is therefore significant.

In addition to these steps, the core of a workflow is the commands, which make the playbook executable. The specification allows usage of a few basic types, such as *bash* command, *ssh* command, etc. It also supports *openc2-json* command [2], which is a standardized language for the command and control of technologies that provide or support cyber defences. These commands are what make security playbooks automated. In addition to these commands, it is also possible to use a *manual command*, which describes a command that is intended to be processed by a human operator. One of the last pieces utilized in CACAO playbooks for their true automation is *Targets*. When some automatic command is defined, it needs the location where it should be executed. *Targets* are objects that contain detailed information about entities that somehow accept, receive, and execute commands defined in workflow steps.

The CACAO specification is a well-defined way of structure used for cyber security playbooks. When sharing of playbooks comes into the game, the CACAO data model offers so-called *data markings*, which define the handling or sharing requirements of playbooks. Both widely used sharing classification mechanisms TLP and IEP can be

utilized by CACAO playbooks. If the base CACAO model is not enough, it also allows creating self extensions.

## 2.2 Overview of the SAPPAN Playbook Specification and Vocabulary

The SAPPAN vocabulary and capturing tool was developed in the scope of WP4 based on semantic web ontology development. We identified the requirements, studied the benefits and shortcomings of utilising semantic technologies, and analyzed the resources to structure response and recovery steps in deliverable D4.1. Later, in deliverable D4.3, a proof-of-concept tool for response and recovery knowledge capturing was developed based on Semantic MediaWiki, as well as the preliminary vocabulary. And finally, the SAPPAN vocabulary for response and recovery actions has been developed in deliverable D4.2. Initial ideas for automation support were also discussed in the same D4.2. However, further discussions regarding automation vocabulary are in the scope of the current deliverable.

## 2.3 SAPPAN Vocabulary vs. CACAO Specification

The scope of the CACAO specification meets the vocabulary-related objectives of the SAPPAN project in the course of response automation. The methodology and structure of the playbooks of CACAO are relatively similar to our proposed methodology in D4.1; However, CACAO follows a more pragmatic approach with less flexibility in changes in the methodology and the structure.

Besides, OASIS pushes for standardisation in the domain of cybersecurity automation playbooks. CACAO is a mature specification with an active and impactful technical committee, including major organisations in the cyber security domain. CACAO has regular refinement on the specification, which shows a great effort for further supports on the specification.

On the other hand, developing SAPPAN vocabulary utilising semantic technologies potentially provides the opportunity for easy integration of a *knowledge-base*, which follow a similar specification. However, as a lesson learned, we find out that the cyber security domain is fragmented from the semantic web solutions. It means the current documentation of cyber security knowledge follows a more pragmatic approach closer to CACAO specification. Therefore, CACAO potentially has a large userbase in comparison to SAPPAN vocabulary.

Further, CACAO covers the missing parts of SAPPAN vocabulary regarding automation. Although, the missing automation vocabulary of SAPPAN (thresholds, automation privileges, confidence scores, and risk metrics) is identified in the SAPPAN Response Automation Prototypes chapter of the current deliverable and can be integrated into SAPPAN vocabulary. This makes CACAO vocabulary a suitable match for the automation playbooks in comparison to the current SAPPAN vocabulary. Also, the standardisation effort of CACAO and reusing other standards in the specification allow SAPPAN to reuse the CACAO standard without having to redefine the schemas or constructs again within SAPPAN, which was one of the concerns during the development of SAPPAN vocabulary.

We decided to contribute to the CACAO standardisation activities by our feedback and developing utilities to ease the work with CACAO. These efforts are described in the

following section. Additionally, we plan to integrate the CACAO vocabulary into our capturing tool.

## 2.4 SAPPAN Contributions in CACAO Standardization Activities

### 2.4.1 Feedback on CACAO

While the SAPPAN consortium has a closer look at the CACAO standardization, we prepare feedback on the specification based on our experiences. We introduce CESNET as the member organisation of the OASIS and plan to contribute to the CACAO specification through a Technical Committee member from the SAPPAN consortium. Thus, knowledge gained during the SAPPAN project will improve the results of the CACAO standardization.

Our main feedback currently consists of the following points, but the investigation and discussions are still ongoing:

- *start/end* steps;
- *UUID* generation;
- *if/switch* condition steps;
- several *parallel* steps merging;
- utility of *parallel/while* condition steps;
- variables dictionary definition;
- workflow step common properties (*timeout*, *on\_success*, and *on\_failure*);
- *playbook* type versus *playbook step* type;
- "*playbook-features*" datatype definition for exclusion;
- dictionary/example conflict in "*cases*" property in *switch-condition* objects;
- dictionary/identifier conflict in "*target\_extensions*" property of *target* objects;
- dictionary/identifier conflict in "*workflow*" property of *playbook* objects;
- versioning of drafts;
- impact, risk, and threshold metrics.

### 2.4.2 Automatic CACAO Validation Utility

The CACAO specification is a new format. Therefore, no utilities to ease its use are available yet. The first step in this direction is developing a tool to process the machine-readable data of a CACAO Playbook and assure that it conforms to specifications. This tool is helpful for two reasons. First, because the CACAO format is only human-readable in a limited fashion, therefore difficult to find errors purely by using a human operator. Second, because there is still a lack of utility for creating such Playbooks, thus all examples so far are created manually, where minor errors are inevitable.

For this purpose, the CACAO Validation Utility has been developed in the course of this deliverable, which takes a CACAO playbook in *JSON* format as input and checks its integrity. The current version of the validator is available in the consortium GitLab repository and can be accessible upon a request to [info@sappan-project.eu](mailto:info@sappan-project.eu).

The current version of the validator tool performs the following steps, particularly:

- assures that the playbook follows a valid *JSON* format;

- assures that all atomic data types (*string, boolean, etc.*) and data storage types (*list, dictionary, etc.*) in the playbook conform to the conditions outlined in the specification;
- assures that complex objects contain all required properties;
- assures that when a property is included, it also fulfils any additional rules for this property, if any are specified;
- provides warnings when practices that are suggested in the specification are not followed;
- provides a warning when a referenced object is not present in the playbook;
- provides a stack trace for errors to ease resolving them.

### 2.4.3 A MISP Data Model to Share CACAO Playbooks

The contribution of sharing CACAO playbooks via the MISP threat sharing platform [3] is in the scope of SAPPAN deliverable D5.8. The default objects of MISP are not fit for the CACAO playbook structure. Therefore, after identifying all included metadata in CACAO playbooks and considering what will be needed for sharing, some of them are mapped as attributes and the others as tags from MISP taxonomies [4] or galaxies [5]. Based on this mapping, the final CACAO-compatible playbook object has been created for MISP. This object hosts not only CACAO playbooks but also other specifications such as the SAPPAN playbook specification. For more details, please read the corresponding deliverable D5.8.

### 2.4.4 CACAO Playbooks Translator

In the scope of this task, we develop a utility to translate from CACAO format to directed acyclic graphs used in a workflow management platform Apache Airflow. The effort is described in the following chapter, in a separate section: "CACAO Playbooks Translator into Airflow DAGs". This two-way translator is a utility to connect CACAO automation playbooks to a workflow management platform towards automation of cyber security response actions.

## 3 Workflow Automation

### 3.1 Security Orchestration, Automation and Response

In the context of this deliverable, workflow automation is discussed in relation to Security Orchestration, Automation and Response (SOAR). Since this is a relatively new topic, it is important to use a well-defined understanding of the term.

We understand the term SOAR to encompass the following capabilities:

- Case management and workflow capabilities: Just like IT support and helpdesk teams use IT service management tools to track and control their work, Security Operations Centers (SOCs) need tools to manage and control the work of triaging alerts, as well as raising, investigating, and solving incidents.
- Automation of tasks from those activities via orchestration of multiple tools, such as Endpoint Detection and Response (EDR), Security Information and Event Management (SIEM) and Network Detection and Response (NDR) tools. Often times with the goal to automate response and recovery actions (R&RA). A representation of response and recovery actions can be understood to be a prerequisite for a working SOAR solution because the workflows and processes that are used within a SOAR framework need to be described and stored in a machine-readable format. The same can be said for other components of SOAR, such as workflow engines.
- Accessing and querying threat intelligence in a centralized fashion for enrichment purposes.

The main promise of SOAR is to achieve all of this in a vendor-agnostic fashion, such that security tools can be picked and chosen from in a plug and play fashion. Unfortunately, there are many incentives that counteract this very goal: Every project and especially every vendor that is putting time and money into developing a SOAR framework or other security products, naturally has the interest to tie the respective users to the given product and will likely use some kind of product lock-in enabling procedures or designs, like proprietary data formats.

### 3.2 Available Automation Tools

#### 3.2.1 Current Situation

Currently, a number of different frameworks to implement and/or describe SOAR, as well as represent information on response and recovery actions, are available.

Popular efforts include:

- Integrated Adaptive Cyber Defense (IACD),
- OASIS Collaborative Automated Course of Action Operations (CACAO) for Cyber Security,
- The Hive - Cortex,
- Splunk Phantom Cyber,
- WALKOFF,
- Shuffle,
- IBM Node-Red,

- Apache Airflow.

as described in SAPPAN deliverable 5.7.

Within the SAPPAN WP4, a format to represent response and recovery actions (R&RA) was developed. To plan the next steps, a consensus on how to proceed with regard to the different existing frameworks as opposed to the SAPPANs format is needed. The following analysis aims to provide some insights into this decision.

### 3.2.2 Analysis

The SOAR and R&RA space are currently in a strong consolidation phase, which means that no standardization or consensus is available. Experience and history show that top-down standardization rarely is successful in fragmented environments. It also shows that defining yet another abstract format without providing any tools is unlikely to speed along further standardization. Opposed to the top-down approach, bottom-up, tool, and plugin-based efforts are much more likely to facilitate change and consolidation, and are also more likely to generate value.

From this perspective, sensible courses of action for SAPPAN are to:

- focus on providing the necessary tooling to translate between two or more different formats for R&RA, or
- develop tools based on existing SOAR/automation solutions that are able to understand existing R&RA formats or interface with third-party security solutions.

Such steps might be able to counteract the forces that are currently preventing the consolidation of the different frameworks and projects: Once tools become widely available that understand different formats and frameworks, a wider user base can get involved and will, in turn, be able to choose the frameworks and formats that fit their respective needs in the best way possible.

### 3.2.3 Frameworks

In the context of SAPPAN, different SOAR frameworks to automate response and recovery actions have been analyzed. Table 1 shows the results of the investigation.

**Table 1: Overview of the SOAR frameworks to automate R&R actions.**

Frame-work	Description	Programing Language / Format	License	Experience/Dis-cussion	Resources
Integrated Adaptive Cyber Defense (IACD)	IACD is a strategic framework for SOAR. It aims to simplify the shar- ing of work-	XML/BPMN	n/a	IACD does not seem to be very well consolidated and/or adapted. The documenta- tion shows a lot of interesting ideas	<a href="https://www.iacdauto- mate.org/">https://www.iacdauto- mate.org/</a>

	flows between organisations by defining a common framework and vocabulary.			but does not provide any end-to-end implementation examples. Whenever it comes to the question of actual implementation/tooling, other frameworks and tools seem to be used.	
OASIS Collaborative Automated Course of Action Operations (CACAO) for Cyber Security	CACAO is a standard for defining a sequence of cyber defence actions that can be executed for different types of playbooks.	-	n/a	CACAO is in its early stages but the standardization effort has been carried out by a wide variety of committee members.  It seems like a good candidate for SAPPAN to align its efforts with.	<a href="https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=cacao">https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=cacao</a>
The Hive - Cortex	The Hive provides a stable framework for case management in a SIEM context.  Cortex provides ways for observables, such as IP and email addresses, URLs, domain names, files or hashes, to be analyzed one by one or in bulk mode using a web interface. One can also <b>automate</b> these operations thanks to the Cortex REST API.	Scala, AngularJS, Python	Open source and free software released under the <u>AGPL</u> (Affero General Public License).	The Hive as a case management solution is a ticketing solution tailored to incident response and security operations. It provides analysts with tools to treat alerts by either dismissing them, turning them into cases and then further gathering information and documenting the whole process.  Cortex integrates well with TheHive and provides a lot of flexibility, as well as standardization through its "responder" and "analyzer" inter-	<a href="https://github.com/TheHive-Project/">https://github.com/TheHive-Project/</a>

				<p>faces. Cortex-analyzers operate on given observables (which are assigned to a case) and enrich the case with additional data (e.g. the result of a lookup to <a href="http://virustotal.com">virustotal.com</a> or <a href="http://robtex.com">robtex.com</a>), whereas Cortex-responders provide ways to interact with other components based on recorded observables (e.g. blocking a given IP on an endpoint through interaction with an endpoint-agent).</p> <p>Both solutions work well, are rather mature and well established.</p>	
Splunk Phantom Cyber	Splunk Phantom is re-branded to Splunk SOAR. Splunk SOAR combines security infrastructure orchestration, playbook automation, case management and integrated threat intelligence.	Python	Limited for free usage	<p>License: Limited for free usage.</p> <p>It would be feasible for SAPPAN demo purposes but cannot be leveraged for CySOC-production without purchasing a license.</p> <p>Conclusion: Phantom is commercial, and we do not want to integrate commercial software with CySOC.</p>	<a href="https://github.com/phantomcyber/">https://github.com/phantomcyber/</a>
WALKOFF	WALKOFF is an automation framework allowing users to integrate their capabilities	Python	This work was prepared by an U.S. Government employee and, therefore, is excluded from copyright by	Does not work out of the box. Solution is not too actively developed.	<a href="https://nsacyber.github.io/WALKOFF/">https://nsacyber.github.io/WALKOFF/</a>  <u>(abandoned)</u> <u>Installation of</u>



	and devices to ease repetitive and time-consuming tasks. It is a modular solution deployable on Windows or Linux, includes a visual analytics component.		<p>Section 105 of the Copyright Act of 1976. Copyright and Related Rights in the Work worldwide are waived through the CC0 1.0 Universal license.</p> <p><a href="https://github.com/nsacyber/WALKOFF/blob/master/LICENSE.md">https://github.com/nsacyber/WALKOFF/blob/master/LICENSE.md</a></p>	<p>Trouble getting apps to run.</p> <p>DL contributed a fix for some issues (<a href="https://github.com/nsacyber/WALKOFF/pull/273">https://github.com/nsacyber/WALKOFF/pull/273</a>).</p> <p>Conclusion:</p> <ul style="list-style-type: none"> <li>WALK-OFF and Shuffle are more or less the same;</li> <li>we can use them, but we'd have to extend them (not enough/good enough automation available in the solutions).</li> </ul>	<u>WALKOFF</u> <u>SOAR</u>
Shuffle	Shuffle is an automation platform focused on accessibility to the solutions and integration to customers' tools.	Python	<p>All modular information related to Shuffle will be under MIT (anyone can use it for whatever purpose), with Shuffle itself using AGPLv3.</p> <ul style="list-style-type: none"> <li>Workflows: MIT</li> <li>Documentation: MIT</li> <li>Shuffle backend : AG-PLv3</li> <li>Apps, specification</li> </ul>	<p>Works out of the box. UI is a bit immature but seems to do the job.</p> <p>Shuffle seems not to be well suited for our use-cases (scenario involves taking actions in case management tool (the hive)).</p> <p>Conclusion:</p> <ul style="list-style-type: none"> <li>WALK-OFF and Shuffle are more or less the same;</li> <li>we can use them,</li> </ul>	<a href="https://github.com/frikky/Shuffle">https://github.com/frikky/Shuffle</a>

			and App SDK: MIT	but we'd have to extend them (not enough/good enough automation available in the solutions).	
Node-Red	Node-RED is a programming tool, which provides a browser-based flow editor.	NodeJS	Unclear	Mature solution, based on nodejs.  Conclusion: Could be used, we'd rather not introduce a new language (javascript) into the CySOC universe.	<a href="https://nodered.org/">https://nodered.org/</a>
Airflow	Apache Airflow is a workflow management platform. It allows the creation of workflows in Python for the purpose of automation. It also has a GUI to visualize workflows and provides visualizations for various purposes (E.g., workflows, workflow execution, etc.).	Python	Apache License 2.0  <a href="https://www.apache.org/licenses/LICENSE-2.0">https://www.apache.org/licenses/LICENSE-2.0</a>	Airflow is more of an automation/workflow engine than a SOAR framework. However, it is mature and stable and provides an API that can be used from TheHive.	<a href="https://airflow.apache.org/">https://airflow.apache.org/</a>

As we listed in the Table 1, The Hive is a mature case management system widely in use in the domain of SIEM. Moreover, Airflow is a stable automation workflow engine with an API to connect to The Hive. For Phishing and DGA detection and response showcases, we decided to focus on a combination of both solutions, to develop SAPPAN response automation prototypes.

### 3.3 Using Apache Airflow for Workflow Automation

#### 3.3.1 Workflow Model

Apache Airflow models its workflows as Directed Acyclic Graphs (DAGs), which means that every node is only triggered once and for each node no path exists that leads to one of its parents, such as shown in Figure 2.

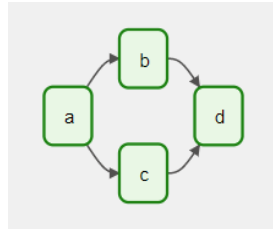


Figure 2: Example of a DAG.

In Airflow, every node of the Graph represents a task to be executed. These tasks are self-contained. These are triggered during a run once their upstream tasks fulfil the task's trigger condition. By default, this means that a task starts executing once all upstream tasks have terminated, without being marked as skipped or failed.

Execution of a workflow is called a DAG Run. Each time a workflow executes, it creates a DAG Run instance for that workflow. Multiple instances of the same workflow can be executed in parallel.

Just as each DAG Run is an isolated instance, executing a task creates a self-contained instance, as visualized in Figure 3. A task gets scheduled once its trigger rule is satisfied. Then it gets executed by the Airflow Engine as soon as possible, either sequentially or in parallel, depending on the configuration. This model of encapsulation and queue-based execution allows for efficient parallelism.

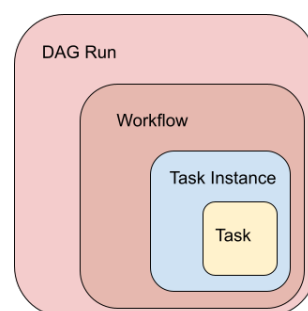


Figure 3: Visualization of the encapsulation principles of Apache Airflow.

#### 3.3.2 Implementation of Workflow Steps

Each Workflow is defined in a Python file where each step is built using an Operator provided by Airflow. The function of each Operator is defined by its type. Examples include but are not limited to:

- **DummyOperator:** A step that does not perform any action and can be used for visual distinction or flow management;

- **BranchOperator:** This step type allows for branching paths within workflows, allowing a subset of downstream steps to be triggered on termination, instead of all of them;
- **PythonOperator:** Allows the calling of executable code written in Python as complex or simple as desired;
- **Bash/Email/Sql/... Operators:** A variety of operators designed to execute simple and commonly used commands.

While steps by default trigger when all upstream steps are marked as successful, this behaviour is customizable to suit the desired objective. Therefore, for example, a task triggers when at least one upstream task is successful.

The only communication between Steps occurs through XCOMs, which is a communication method provided by Airflow. It means that for the sharing of small data, they can contain any serializable information and are accessible by all tasks within the same DAG Run. Each task holds a dictionary of XCOMs with key and value pairs that can be accessed from other tasks in the same run through the task id and the relevant key.

### 3.3.3 Automated Workflow Execution

Airflow offers two methods of workflow execution:

- **Scheduled Execution:**
  - Every DAG can be configured to run at certain dates or intervals. This is configured within the DAG itself and gets automatically executed as long as the DAG is marked as active.
- **Manual trigger:**
  - A DAG Run can be started using the web interface by a human operator.
  - A DAG can be called by another DAG that is already being executed. A special operator is provided for this function.
  - A DAG can be called through the Airflow API from a networked computer by script or human operator.
  - A DAG can be called using the command line on the machine, on which Airflow is installed.

### 3.3.4 Built-in Visualization and Evaluation

Airflow provides built-in tools for the evaluation and analyzes of workflows, as well as previous or ongoing DAG Runs. Inherently Airflow provides a visualization of the graph structure of its DAGs, including colourizing each task according to its operator type and status both for ongoing and previous runs, as seen in Figure 4. For a more detailed analysis of previous runs, Airflow logs and visualizes the runtime of individual tasks, as seen in Figure 5, for each run. Finally, as seen in Figure 6, it provides a tree-based overview of previous runs for easy identification of frequent points of failures.

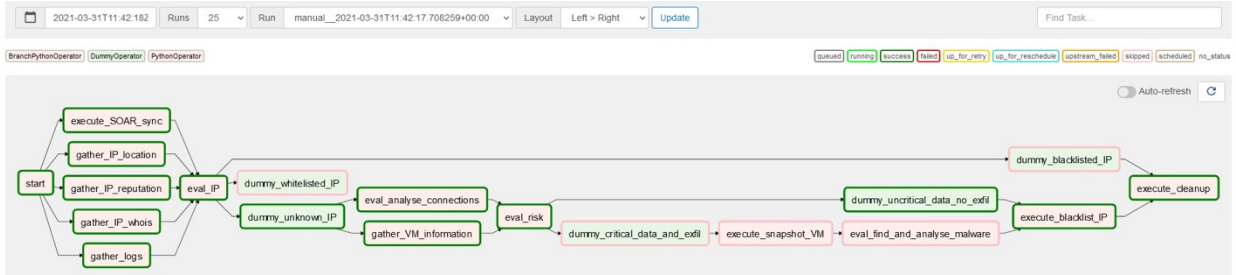


Figure 4: The built-in graph visualization of Apache Airflow for DAGs.



Figure 5: The built-in visualization of Apache Airflow for individual tasks runtime in one DAG run.

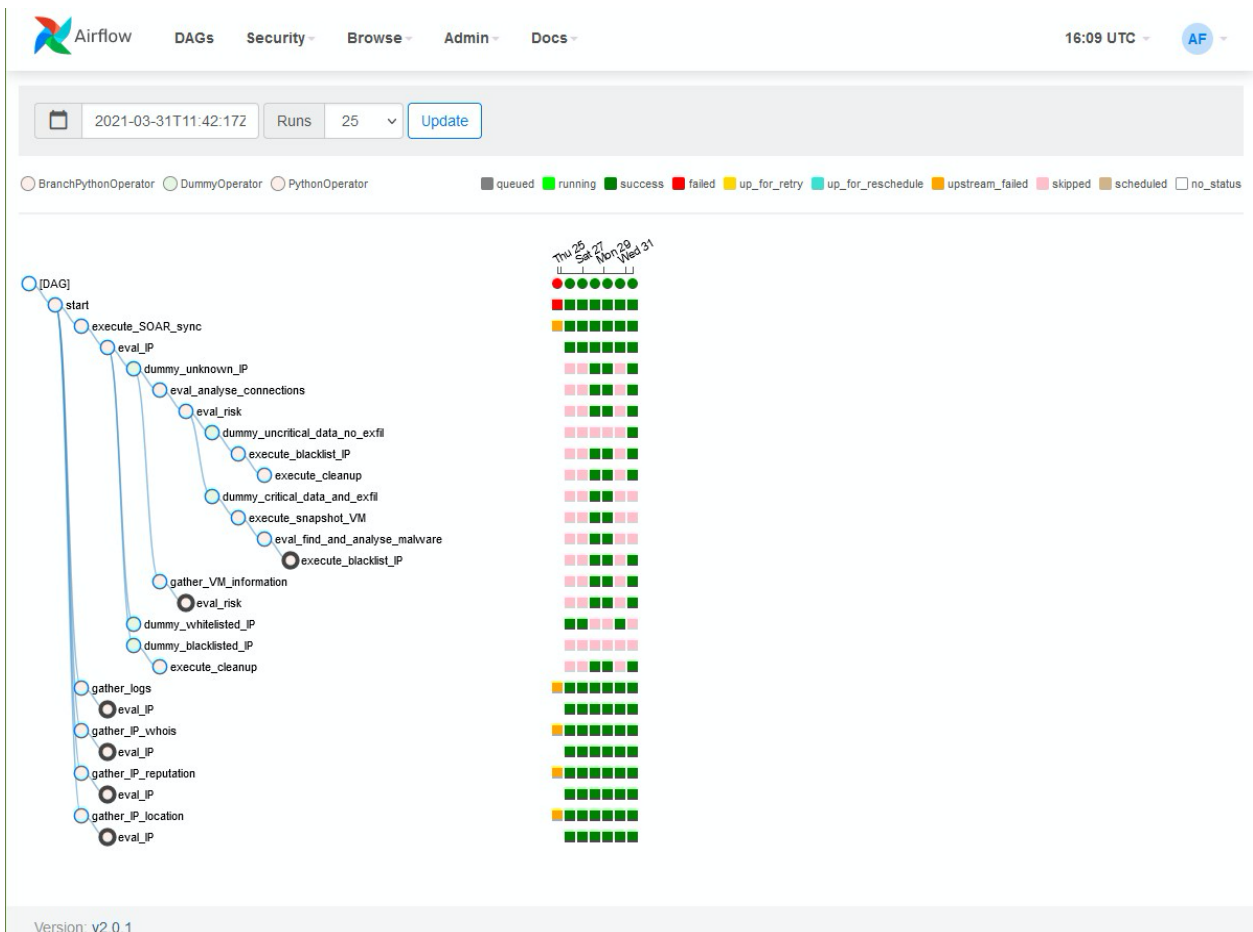


Figure 6: The built-in visualization of Apache Airflow for previous DAG runs showing the status of each task in the run.

### 3.3.5 Concerns and Feedback

Our general concerns about Airflow in the domain of cyber security response automation are listed below:

- As the Apache Airflow workflow model is based on DAGs, no loops are supported directly within a workflow.
- The default combination of task scheduler and database seems to be relatively slow.
- The default way of handing down information between tasks is restricted. However, as tasks are implemented in Python, relevant information could be exported to a file and read by another task. (Pickling, JSON, etc.).

Further, some improvements to the Graphical User Interface can make the tool more user-friendly:

- The creation of new workflows could be supported by a GUI. Workflows could be created using such a GUI via drag-and-drop of components, the tool then automatically creates respective dummy tasks and connections between tasks in code. The dummy tasks then need to be filled in by a human operator.
- The graphical workflow visualization could display more information from the code of each task if clicked on it. This suggests a zooming utility for the tool.

### 3.4 CACAO Playbooks Translator into Airflow DAGs

Airflow provides a good environment for playbooks. It is capable of visualizing the workflow, as well as providing the ability to attach executable code to the workflow steps. A utility that eases the translation of a given playbook in CACAO format into an Airflow DAG is therefore useful.

In the course of SAPPAN, we develop a translator from CACAO format to Airflow DAG and vice versa. The translation utility is able to take a CACAO Playbook and automatically generate a framework DAG file. The utility processes each workflow step in the playbook and creates an equivalent task in the DAG format. In addition, it creates a callable function for this workflow step into which a relevant code can later be inserted. Moreover, it processes the connections between steps, automatically creating the flow in the DAG, as well as choosing operator type appropriate to the relevant task (e.g., DummyOperators when the step does not contain instructions, BranchOperators for if-steps, etc.). However, it does not copy any commands or more detailed information and metadata from the CACAO playbook, merely providing a template on which to migrate the playbook. Since DAGs cannot contain cycles, yet the CACAO specification allows for them, it also detects both the use of while-steps and cycles to abort.

Further development may contain metadata transfer between formats and specific responses to cycle detection beyond declaring failure. The current development of the translator is available at consortium GitLab and can be accessible upon a request to [info@sappan-project.eu](mailto:info@sappan-project.eu).

## 4 SAPPAN Response Automation Prototypes

As a part of deliverable D4.6, we progressed towards designing a general framework that may capture the approaches and algorithms for automating response actions on many risk, confidence and severity metrics. We determined two showcases from WP3 for automation (phishing and DGA), which have moderate response risks. We discussed the steps of automation aligned with our suggested framework. In this section, we give an overview of a response automation system, then list the general requirements for a response automation framework. Further, we progress towards the development of the solutions, concrete analysis of the showcases, and proposing detailed threshold, risk and impact values based on the suggested general metrics for response automation considered in the previous deliverable. We tend to offer an automation framework without a human operator via having a closer look at the state-of-the-art tools and frameworks for response automation. In addition, we propose and evaluate another proof-of-concept prototype of a general platform for malware response automation.

### 4.1 Overview

Incident response in a SOC context usually starts with the creation of an alert by a detection mechanism. Sometimes such an alert can immediately be dismissed as a false positive. If this is not the case, the alert will most likely be entered into some kind of case management solution where it is triaged and investigated further. If the alert turns out to be a false positive, the associated case is closed and the alert ignored. If the alert turns out to be a true positive, the case remains open and further response and recovery actions are required. In most SOC environments, all of this involves a lot of manual labour leading to the initial detection mechanism being the only leverage for improving the whole process. In other words: The detector has to carry the burden of doing a good enough job, such that SOC analysts are not overloaded with unduly amounts of alerts while at the same time retaining a high detection rate. This may be a conflict of interest, and since it is easier to optimize a given system for one property at once, it is better to optimize detectors for detection rates and separately improve process efficiency through means of automation.

SOAR can be viewed as an approach for such efficiency optimization, or rather automation. As SOAR operates at the man and machine interface, it is important to consider design goals and customer needs to increase efficiency.

The following design goals were defined when creating POC for SAPPAN:

- **G1:** Do not act upon critical assets autonomously because this could disrupt operations.
- **G2:** Do not miss alerts regarding critical assets, even if detection certainty is low.
- **G3:** Automate the easy/clear cases, leave the uncertain ones to the analyst for manual treatment
- **G4:** Make analyst life easier than it was before.

### 4.1.1 Requirements

The strength or promise of SOAR lies in the automation part. Automation is regarded as the key to improving efficiency in SOCs by automating the repetitive parts of working through and triaging alerts generated by detection systems (such as the phishing and DGA detection prototypes developed in SAPPAN). Successful automation will lead to an improvement in the triage of cases such that analysts can spend more time on the cases that actually require their attention, meaning that most of alerts should be triaged, analyzed and processed automatically, including any required response and remediation.

The requirements for realizing this kind of prototype that were identified before and during the creation of the prototypes are shown in Table 2.

**Table 2: The requirements identified for a response automation prototype.**

#	Requirement	Description
R1	Distinguish between critical and less critical assets when applying automation  (→ pertinent to G1, G2 and G3)	<p>Remediation actions can have potentially disruptive side effects. This is especially true if one is introducing automation and thus the possibility of a system acting upon false positives without any human interaction. Different assets have different criticalities regarding the operations of an enterprise environment. Typically there are more assets with lower criticality (e.g. normal clients or workstations) and fewer assets with higher criticality (key services, like domain controller, mailserver, fileserver, etc.) Remediation actions thus have different potential for disrupting an environment/organisation, depending on what asset is subjected to a remediation action. For assets of low criticality, the principle that it is "easier to ask for forgiveness than permission" is true, whereas for assets with high criticality (and thus a high potential for disruption) the reverse is true.</p> <p>Any automated workflow must thus take the criticality of assets into account in order to manage the tradeoff between:</p> <ul style="list-style-type: none"> <li>• disruption through an unnecessary remediation action (acting upon a false positive) and</li> <li>• disruption through an undetected and thus unmitigated threat.</li> </ul>
R2	Cap the risk of automated actions to prevent catastrophic outcomes  (→ pertinent to G1)	<p>Automating security-relevant decisions and actions can potentially have catastrophic effects. Think about adding firewall rules or changing account passwords, for example. If the affected device is a domain controller, it will, in most cases, not be advisable to automatically carry out any actions without checking back with an experienced analyst. R1 somewhat mitigate this risk if implemented properly, still the proposed solution must provide a means to make guarantees that certain assets will not be acted upon automatically without completely excluding them from the proposed workflow (completely blacklisting a critical asset is not the solution here).</p>



R3	Use existing platforms / familiar interfaces  (→ pertinent to G4)	SOC analysts are used to utilizing established tools, especially when it comes to case management and ticketing. The proposed solution should thus integrate with existing, known platforms and utilize already available interfaces and APIs as much as possible.
R4	Provide overview for SOC manager  (→ pertinent to G4)	The SOC manager should be given an overview of the automated workflows and their current status in order to trust the automation engine.

## 4.2 Phishing Showcase Prototype

We develop an automated responder using Apache Airflow, The Hive and classifiers provided in deliverable D3.4 to demonstrate a practical application of the diverse automation methods we researched as part of SAPPAN task T4.4.

The idea is to create a response system that can process incoming emails, evaluate them for spam and phishing attempts, and if positive, based on thresholds, either take automatic action or bring the case to the attention of a human operator.

In such an automatic system, the tasks can be performed in parallel. It is beneficial compared to a manual system, in which the tasks run sequentially (e.g., gathering information from multiple sources). Also, an automatic response can be triggered significantly faster than a manual response by a human operator. This can improve the performance and reduce the response time. Further, due to rapid response, it should be possible to contain several attacks before they harm. For instance, resetting a user's credentials immediately after an attack occurred can outperform this task done by a human operator.

### 4.2.1 Workflow

The workflow, shown in Figure 7, begins by getting a new e-mail in plain text format, information about possible attachments, as well as the network actor this e-mail pertains to. In the case of phishing, this would be the user who received the message. Based on the actor, a created inventory for a client is retrieved. It consists of a dictionary containing boolean and numeric values. In parallel, the message and URLs get sent through a classifier each. These produce a certainty value between 0 and 1 to describe how likely this message is a phishing attempt. After this data gathering, a decision is made whether to proceed fully automatically, request a human operator's approval, or not to proceed at all. The Hive is used to communicate with the human operator and automatically create an entry for the case. The final phase consists of action, should the workflow have gotten approval from the operator or proceeded automatically. Here it will perform various actions based on a calculated score and certain properties of the e-mail, always deciding between a low impact and high impact action. The Table 3 lists the main components of the workflow and their descriptions.

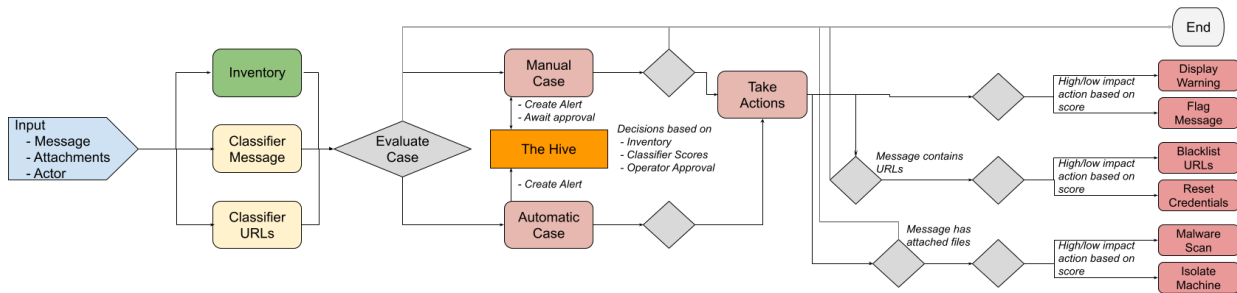


Figure 7: The workflow of the phishing showcase prototype.

Table 3: Description of the workflow steps for the phishing showcase prototype.

Work-flow Activity	Description
Classifier Message	This step uses a Multinomial Naive Bayes classifier, using the scikit-learn framework [6], to analyze the plain text of the message in comparison to confirmed spam or non-spam (HAM) messages. It returns a certainty value between 0 and 1, where 0 means it is unlikely to be spam mail and 1 is the highest likelihood.
Classifier URLs	For the URL classifier, we aim to utilise Machine Learning (ML)-based classifiers developed in SAPPAN WP3. Those classifiers return a confidence value between 0 and 1, where 0 means it is unlikely to be a phishing URL and 1 is the highest likelihood. For the current proof-of-concept, we have not integrated the SAPPAN phishing URL classifiers yet. We use a simple classifier that uses the online database <a href="https://phishtank.org">phishtank.org</a> to check for each URL in the message if it is a known phishing address. The classifier returns 1 when it discovers a phishing URL and 0 if not.
Inventory	This inventory is created by the client and describes the tolerances and permissions in regard to the given actor and consists of the parameters shown in Tables 4 and 5.
Evaluate Case	Once the analysis steps are complete, the decision is made whether to proceed and create an automatic or manual case or terminate based on the permission booleans, the thresholds and the inventory values using the decision formula.
Automatic Case	An alert is created in The Hive for documentation and review purposes, then it intermediately proceeds to the 'Take Actions' step.
Manual Case	This step behaves much like the automatic alert but creates a custom field in the alert named "proceed?" ('open', 'yes', 'no') and initialized with the value 'open'. The step then regularly polls the alert to see if an operator has modified the property. If it is set to 'yes', the workflow proceeds to the 'Take Actions' step; if set to 'no', the workflow terminates and all further action is left to the human operator.

Take Actions	After evaluation and possible approval from a human operator, this step decides which actions to take. In the scope of the prototype, three groups of actions are created, each consisting of a high impact and a low impact action. The general actions consist of displaying a warning to the user as a low impact action and flagging the message as malicious as a higher impact action. If the message contained any URLs, either the low impact action of blacklisting those are taken or in case of higher certainty, the high impact action, e.g., resetting the email account credentials is executed. Lastly, if the message had any attachments, either the low impact action of running a virus scan is taken, or in case of higher certainty, the high impact action of quarantining the affected system is applied.
Action steps	In these steps, the interfaces to the organizations corresponding actions would be implemented.

**Table 4: The switches and thresholds defined in the inventory.**

ID	Type	Description
permit_global	Bool	Designates whether the workflow is permitted to be used for the given actor.  If False, terminates the workflow without taking action.
threshold_global	Float	A certainty value that has to be exceeded for any action to be taken.
permit_auto	Bool	Designates whether automated action is permitted for this user.  If False, a manual case will always be created.
threshold_auto	Float	A certainty value that has to be exceeded for an automated case to be created.  Otherwise, a manual case will be created.
threshold_flag	Float	A certainty value that has to be exceeded for the message to be marked as malicious; otherwise, only a warning will be displayed.
threshold_pass_reset	Float	A certainty value that has to be exceeded for automatic credentials reset to be performed for the actor; otherwise, only the URLs in the message get blacklisted.
threshold_quarantine	Float	A certainty value that has to be exceeded for the automated quarantine of the affected system; otherwise, only a virus scan will be performed on the affected system.
permit_flag	Bool	Designates whether the workflow is permitted to flag a message as malicious.
permit_pass_reset	Bool	Designates whether the workflow is permitted to perform password resets.

permit_quarantine	Bool	Designates whether the workflow is permitted to quarantine the system.
values	Dict	Contains the values used to modify the thresholds in regard to the actor and allows a fudge value the operator can adjust for specific actors.

**Table 5: The values that describe the impact and risk for an actor, as well as the fudge value that can be set by a human operator for adjustment.**

ID	Type	Description
impact	(Float, Float)	<p>A 2-tuple of numeric values.</p> <p>The first value describes the impact actions on this actor would have on the ability of the organization to function, in the range of [0.0, 1.0].</p> <p>A high value represents little to no impact, while a low value represents strong impacts.</p> <p>The second value represents the weight this value has in the final calculation. It should be 1.0 as default.</p>
risk	(Float, Float)	<p>A 2-tuple of numeric values.</p> <p>The first value describes the risk to the organization should the actor be compromised, in the range of [0.0, 1.0].</p> <p>A low value represents little to no damage from data leaks or compromised credentials, while a high value represents a severe risk to the organization.</p> <p>The second value represents the weight this value has in the final calculation. It should be 1.0 as default.</p>
fudge	(Float, Float)	<p>A 2-tuple of numeric values that represent fudging from a human operator, should it be observed that the system is too sensitive or not sensitive enough regarding this actor.</p> <p>The first value should be either 1.0 for increased sensitivity or 0.0 for decreased sensitivity, while the second value represents the weight this value has in the final calculation and should be set as desired.</p> <p>If no adjustment is desired, it should be omitted or weighted 0.</p>

We can incorporate the inventory taxonomy presented in Tables 4 and 5 into the SAPPAN vocabulary developed in D4.2.

## 4.2.2 Decision Formula and Case Evaluation

To decide whether to take any decision, first, the permit boolean that was defined for that decision in the inventory should be enabled. If not, the decision immediately fails.

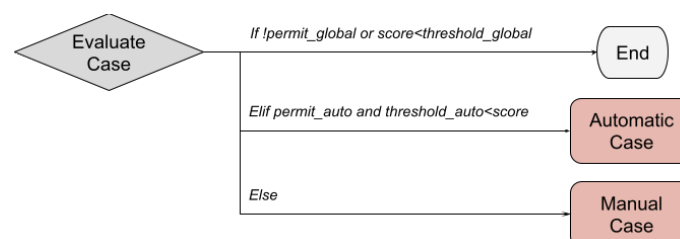
We take the weighted average of the *impact*, *risk* and *fudge* values, as defined in the inventory (cf. Table 5). Here, *risk* represents the danger to the company when an attack succeeds. The *impact* value describes the disruption the company would experience when remedial action is taken. Further, *fudge* value is used by human operators to alter the decision process when they discover it requires adjustment.

The decision formula includes the scores produced by the classifiers in the workflow as well. One calculates a score based on the similarity of the message text to other phishing e-mails and the other by matching URLs in the message against known phishing sites. We take the maximum of these scores to get a value representing the danger of this message, since even if the message text looks (or is) genuine, if it still links to phishing sites, the message is dangerous and vice versa.

As a very simple heuristics, the final score is calculated by taking the average between the weighted average of inventory values for the actor and the maximum classifier score for the message as shown in the following formula:

$$\frac{\text{weighted\_average}(\text{values}) + \max(\text{score\_message}, \text{score\_urls})}{2}$$

If the resulting score exceeds the threshold defined in the inventory, the decision is successful. The detailed decision making is shown in Figure 8. That means if any action is forbidden through the use of the 'permit\_global' flag or the score calculated using the decision formula be less than the 'threshold\_global' value, no action will be taken. If that first hurdle is passed, the workflow decides whether it should proceed completely automatically or wait for approval from a human operator. An automatic case is created if an automatic action is permitted through the 'permit\_auto' flag and the score exceeds the 'threshold\_auto' limitation. Otherwise, the workflow creates a manual case.



**Figure 8: A more detailed look at the decision-making process of the workflow in regards to automatic/manual action.**

## 4.2.3 Interfacing

The workflow has several points of interaction that have to be adjusted for embedding into an organization's existing infrastructure.

Specifically, this includes input:

- the raw message and actor identification, in the input step;
- the thresholds and permissions that are retrieved in the Inventory step.

And output:

- the action steps carrying out the relevant responses;
- credentials for The Hive alert creation in the Automatic/Manual Case steps.

This is luckily very easy in Airflow, since each step can execute Python code. Therefore, either, actions can be directly implemented in the step, other scripts in the network can be called from the step, or API calls can be sent.

Similarly, the input steps can read already existing files on the network, parse parameters passed to the DAG or pull data from other machines on the network.

Meanwhile, The Hive steps require credentials and the address of The Hive instance on the network, which can be either entered directly in the code of the steps or in the accompanying config file.

#### 4.2.3.1 The Hive Integration

Since The Hive is a very widely used security incident response platform, it makes sense to involve it in this workflow. So after the analysis of the problem is complete, the workflow creates an alert in a connected The Hive instance. This serves for record-keeping purposes, allowing security operators to review cases in their own time. If a manual case was created, the alert also contains a custom field "proceed? ('open', 'yes', 'no')", that is initialized with 'open'. The case only proceeds automatically when that field is set to 'yes' and terminates without taking action if set to 'no'.

#### 4.2.4 Example Run

In this section, we present an example. The workflow gets triggered by a message in plain text format and a unique ID relating to the actor. The message contains plain text and URLs, but no attachments.

##### 4.2.4.1 Phase 1: Gathering Info

The message and actor ID arrives in the input step and gets relayed to the classifiers and inventory steps before a case is created, as seen in Figure 9. Our example message contained URLs, so the `url_classifier` checks those against PhishTank and discovers none of them is in the database:

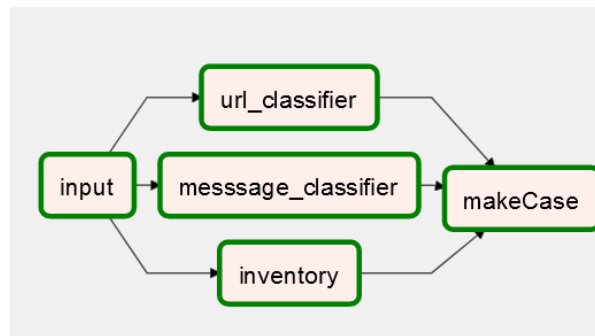
```
url_classifier: returns 0.0
```

The text of the example message is put through our spam classifier and we get a value between 0 and 1 to indicate the certainty it is spam:

```
message_classifier: returns 1.0
```

The inventory step retrieves the actors pertinent information from a database and makes them available for the rest of the workflow:

```
inventory: returns inventory_db['dummyActor']
```



**Figure 9: The information gathering phase of the phishing showcase workflow.**

#### 4.2.4.2 Phase 2: Evaluation

Now the workflow makes the decision on whether to proceed automatically, with manual approval or not at all. The relevant info was gathered in the previous steps:

Inventory:

```

permit_global:True
threshold_global:0.6
permit_auto:True
threshold_auto:0.8
values:
  impact:[0.3,1.0]
  risk:[0.8,1.0]
  fudge:[0.0,0.0]
  
```

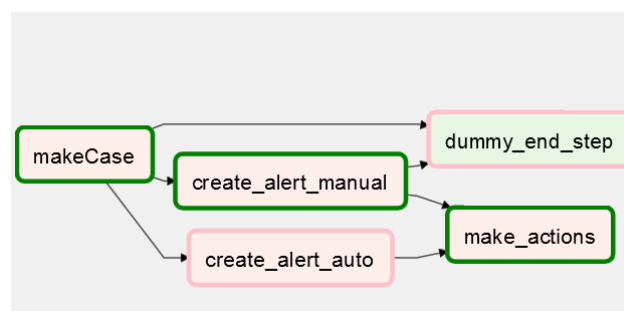
Classifier Message:

```
return:1.0
```

Classifier URLs:

```
return:0.0
```

Applying the decision formula gives us the final score of 0.775. This score is higher than *threshold\_global* but lower than *threshold\_auto*. With both permits set to True, that means we create a manual alert and wait for authorization to proceed. In our example, that authorization is given, and we proceed to the action phase. This path is illustrated in Figure 10.



**Figure 10: Decision on how to proceed a phishing showcase.**

#### 4.2.4.3 Phase 3: Action

After we received authorization from the human operator, we decide what actions to take based on the already calculated score of 0.775, the properties of the message and the inventory.

Inventory:

```

permit_flag:True
threshold_flag:0.4
permit_pass_reset:True
threshold_pass_reset:0.7
permit_quarantine:False
threshold_quarantine:0.8

```

First, we look at the general actions, meaning the low impact action of displaying a warning for the user and the higher impact action of flagging the message as malicious. Our score of 0.775 is higher than the *threshold\_flag* value of 0.6 and the *permit\_flag* was True, so both the flag and warning action were taken. Now the workflow first checks if there were any URLs in the message. In our case, there were so the low impact blacklist action and the high impact pass\_reset action are considered. Again our score exceeds the threshold and the permit was set to true, so both the URL is blacklisted and the credentials of the actor are reset. The last two action steps concern malware. Since our message contained no attachments, this branch is irrelevant and no actions are taken. The resulting actions are shown in Figure 11.

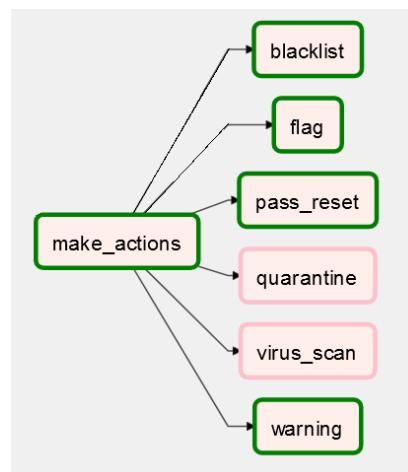


Figure 11: The action phase of a phishing showcase.

#### 4.2.5 Outlook

As this is a proof-of-concept prototype of the automation workflow for the phishing showcase, several roads for improvement are still open to us.

**Integration of the phishing URL classifiers developed in SAPPAN:** ML-based Phishing URL classifiers are developed in the scope of WP3 with promising results. Those classifiers return a confidence value between 0 and 1, where 0 means it is unlikely to be a phishing URL and 1 is the highest likelihood. We aim to integrate those classifiers to evaluate cases.



**Improving confidence scoring system:** One of the considered improvements is analyzing more complicated decision-making strategies. We only use a simple formula in the current prototype, which can be extended using better and more comprehensive heuristics. We can improve it by simulation or real testbed evaluation scenarios. We could also employ other information sources such as similarity metrics, time-related metrics, and response-based lessons learned to improve the system decisions based on the previous cases.

**Templates for integration:** Currently, the points of interaction for the workflow and other systems are kept very open so users can implement their own solutions for these problems. Nevertheless, it might be beneficial to create templates for common solutions to these interactions to keep the implementation similar across deployments. This template creation requires an in-depth real-world study of the showcases. These templates can be later shared across organisations.

**More details in logs:** Extending the amount of information available to the workflow could be used to provide more detailed alerts. It benefits the operator by giving a clearer picture for approving manual cases. Further, it allows for more detailed reviews later.

### 4.3 DGA Showcase Prototype

A simple proof of concept for the automation of response and recovery actions has been created based on the DGA detection showcase. The creation of this prototype was treated as a learning opportunity to identify what questions arise when automating response and recovery steps.

The overview of the prototype's design and the main requirements are discussed at the beginning of this chapter (SAPPAN Response Automation Prototypes). In the remainder of this section, we propose a system architecture for the DGA response automation prototype, and then discuss the identified questions and proposed answers in detail.

#### 4.3.1 System Architecture

Figure 12 depicts the overall architecture of the prototype, and Table 6 shows the components, their description and interactions.

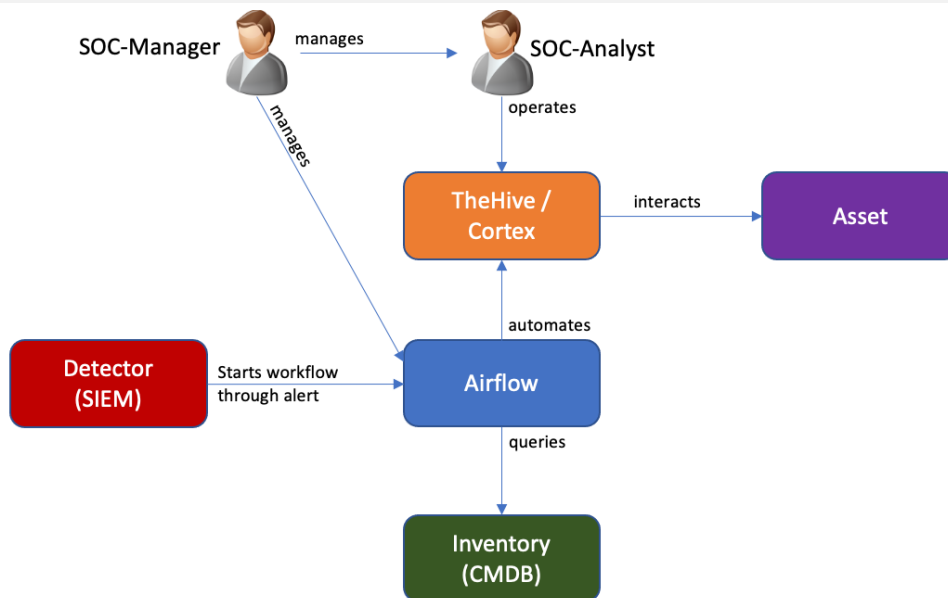


Figure 12: The overall architecture for the DGA Showcase prototype.

Table 6: Description of the components of the DGA showcase prototype.

Component/Entity	Description	Interactions
SOC-Manager	The SOC-Manager is responsible for security operations and fulfils a variety of managerial and organisational tasks.	Manages a team of SOC-Analysts.  Manages automation through Airflow.
SOC-Analyst	The SOC-Analyst is responsible for analysing alerts and working cases, and making decisions based on facts, as well as external inputs (dependent on the organisation and affected assets).	Operates the case management platform.
Detector (SIEM)	A detector monitors an organisation and produces alerts based on what it sees. In the context of this prototype, the detector is an instance of a binary DGA detector (RESNET or CNN, cf. SAPPAN Deliverable D3.4) integrated with a SIEM solution, in this case, Dreamlab's CySOC solution.	Triggers workflows based on alerts.
TheHive / Cortex	A specialized case management solution offering so-called responders and analyzers to SOC-analysts. Both responders, as well as analyzers, are ways of interacting with the environment through, for instance, API calls.	Interacts with affected asset, possibly through some kind of agent.

Airflow	The workflow engine, utilized to run the defined SOAR workflow.	Automates processing of alerts and cases in TheHive.  Queries asset inventory for information on affected assets.
CMDB (asset inventory)	An asset database containing specific SOAR controls to guide the execution of workflows.	
Asset	The affected asset can be a workstation or a more critical device like a mail server, domain controller, etc.	

### 4.3.2 Workflow Description

The first prototype version implements the workflow depicted in Figure 13 below. The individual activities are explained in Table 7 below.

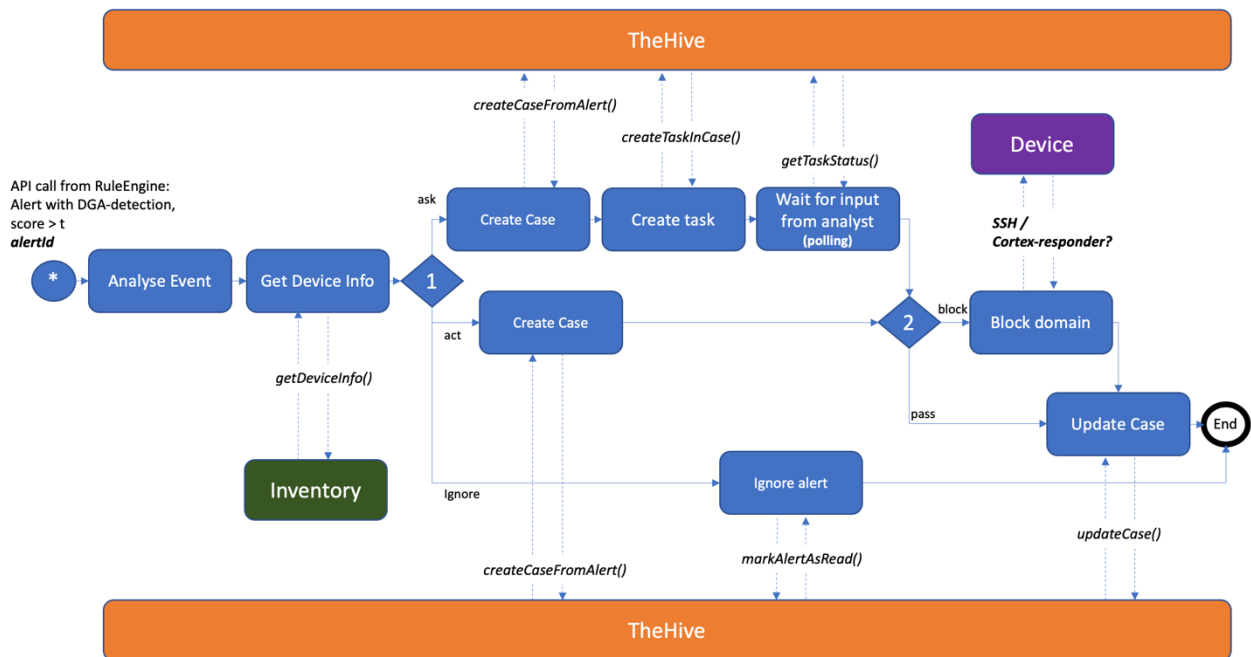


Figure 13: The workflow of the DGA Showcase prototype.

**Table 7: Description of the workflow steps for the DGA Showcase prototype.**

Workflow Activity	Description
(*) - The trigger condition	<p>The trigger for the workflow is an alert coming from a detection mechanism. In this case, this is an alert coming from the binary RESNET-based DGA detector mentioned in SAPPAN Deliverable D3.4. (The original detector was trained on non-resolving traffic only. We retrained the said detector to work on resolving and non-resolving DNS/domains traffic with high success.) The detector takes a domain, for example <a href="http://www.bluewin.ch">www.bluewin.ch</a>, as input, and delivers a score/float between 0 and 1 expressing the certainty of the entered domain being algorithmically generated (i.e. malicious) as output.</p> <p>The workflow starts if the score lies above a certain threshold <b>t</b>. It takes a JSON file describing the event that triggered the alert in the first place. The JSON file contains a portion "enrichment", which can, in principle, contain a bunch of helpful information to be processed at a later stage:</p> <pre> ... "enrichment": {   "domain_hostname": {     "dga": {       "score": 0.997,       "family": "locky"     },   }, }, ... </pre>
Analyze Event	The data received from the alert is parsed and analyzed for necessary information about the asset that is affected (for example, its IP address).
Get Device Info	<p>Based on the asset's IP address, the Asset inventory (a CMDB) is queried for additional information. This is implemented in the <b>getDeviceInfo()</b> API call. To control the SOAR and satisfy requirement R1 and R2, two pieces of information are queried from the asset database:</p> <ul style="list-style-type: none"> <li>• <b>asset.action_preference</b>: The action preference for the given asset (this can be either "Always ask" or "Act autonomously" and is used to fulfil the guarantee required by R2);</li> <li>• <b>asset.importance</b>: The asset importance score (this is again a float between 0 and 1 and is used to fulfil requirement R1).</li> </ul>

Decision <1>	<p>Here a first decision based on the data at hand is taken. If the detection score lies below the asset importance (<b>detection_score &lt; asset.importance</b>), the alert is ignored. This is to ensure that important assets are only affected if the confidence of the alert being a true positive is high enough.</p> <p>If the detection score is high enough, a second decision is taken, whether the asset is too important to be acted upon autonomously (think a domain controller) or whether a response action can be carried out without further checks. For that, the <b>asset.action_preference</b> value from the previous steps is utilized:</p> <ul style="list-style-type: none"> <li>• If the action preference is "always ask", then a case is created in TheHive and a task is added for a SOC-analyst to provide feedback on further actions.</li> <li>• If the action preference is "act autonomously", then a case is created in TheHive and the response action is carried out.</li> </ul>
Create Case	This creates a case in TheHive based on a given alert.
Create Task	This creates a task inside a case in TheHive. Tasks are used to specify jobs for analysts in TheHive and represent an already known way to interact with SOC-analysts. We're using the "Responder" mechanism in TheHive, something with which SOC-analysts will already be familiar, as well.
Wait for input from analyst	This polls for the task SOC-analyst to produce a result based on the feedback task. (To be more precise, a custom field of the TheHive case is polled for a change, which is made through the "responder" mechanism with which the SOC-analyst is interacting).
Decision <2>	This is the gateway to actual response actions being carried out. Depending on previous decisions (especially the interaction with the SOC-analyst), this decision branches into blocking communication to the suspicious domain and closing the case or just closing the case.
Ignore Alert	This ignores a given alert because it is not deemed certain enough.
Block Domain	This carries out the response action. In the example, this is thought of as blocking communication to an identified domain through the utilization of an agent (e.g. Wazuh) on the endpoint (cf. the section on response actions under "Lessons learned").
Update Case	The case is updated to contain the latest information, including decisions made and actions that were carried out.

### 4.3.3 Illustrations

The following screenshots (Figures 14 to 17) show some examples taken during several run-throughs of the DGA Showcase prototype.

**M** Case # 46 - Possible DGA-activity over http(s) from 10.0.2.15 - automated response

Created by admin Thu, Oct 14th, 2021 9:18 +02:00 1 alert

Details Tasks 0 Observables 0

### Summary

Title	Possible DGA-activity over http(s) from 10.0.2.15 - automated response
Severity	M
TLP	TLP:AMBER
PAP	PAP:AMBER
Assignee	admin
Date	Thu, Oct 14th, 2021 9:18 +02:00
Tags	DGA xczjhkgdsadsa.ch BLOCKED

Additional information +

No additional information have been specified

### Description

Possible DGA-activity has been detected coming from host 10.0.2.15 to domain xczjhkgdsadsa.ch.

- The investigation is automated by airflow according to the DGA-playbook.  
Airflow DAG-Id for your reference: <DagRun complete\_flow\_test @ 2021-10-14 07:18:36.399624+00:00: manual\_\_2021-10-14T07:18:36.399624+00:00, externally

Merged with alert #DGA\_1634195921.630422 Possible DGA-activity over http(s) from 10.0.2.15

This alert was auto-created by AirFlow.

**Figure 14: An example for an automated response case in TheHive. This alert affected an asset that had the action preference "act autonomously" (hence the "automated response" in the case title) and communication to the respective domain was blocked on the asset.**

**M** Case # 47 - Possible DGA-activity over http(s) from 10.0.2.15 - interactive response

Created by admin Thu, Oct 14th, 2021 9:24 +02:00 **1 alert**

Details Tasks **1** Observables **1** xczjhkgdsadsa[.]ch

### Summary

<b>Title</b>	Possible DGA-activity over http(s) from 10.0.2.15 - interactive response
<b>Severity</b>	<b>M</b>
<b>TLP</b>	TLP:AMBER
<b>PAP</b>	PAP:AMBER
<b>Assignee</b>	admin
<b>Date</b>	Thu, Oct 14th, 2021 9:24 +02:00
<b>Tags</b>	DGA xczjhkgdsadsa.ch PASSED

Additional information +

### Description

Possible DGA-activity has been detected coming from host *10.0.2.15* to domain *xczjhkgdsadsa.ch*.

- The investigation is automated by airflow according to the DGA-playbook.  
Airflow DAG-Id for your reference: `<DagRun complete_flow_test @ 2021-10-14 07:24:21.645591+00:00: manual__2021-10-14T07:24:21.645591+00:00, externally`

Merged with alert #DGA\_1634196264.3411467 Possible DGA-activity over http(s) from 10.0.2.15

This alert was auto-created by AirFlow.

**Figure 15: An example for an interactive response case in TheHive. This alert affected an asset that had the action preference "always ask" (hence the "interactive response" in the case title) and the analyst decided that the alert is a false positive and no action is required (hence the case carries the tag "passed").**

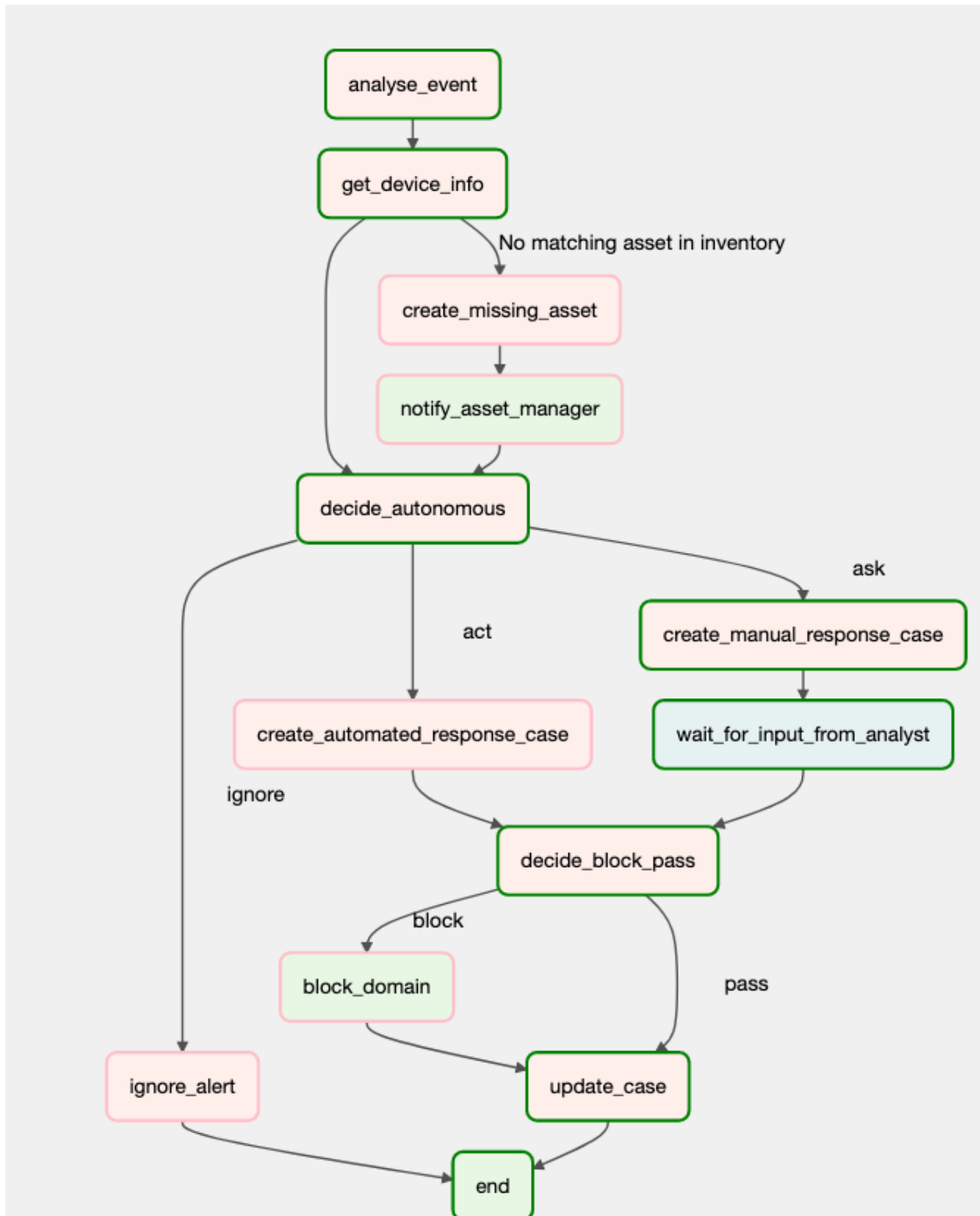
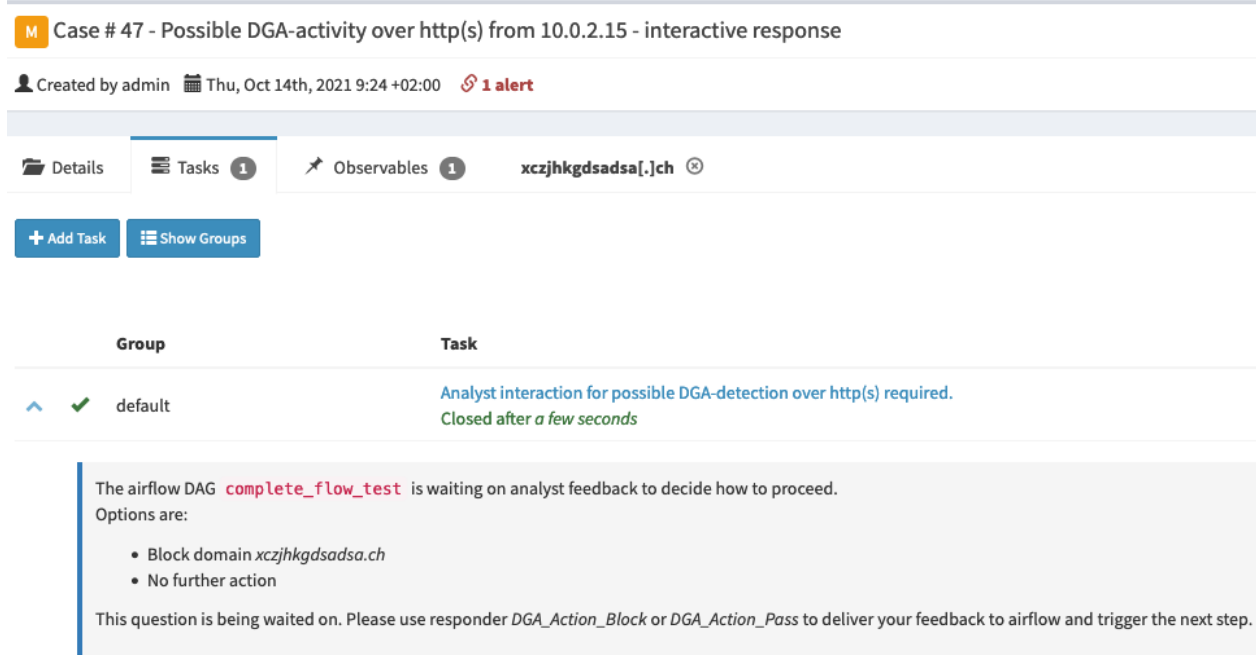


Figure 16: After clicking on "(Link)" in Figure 15, the analyst is led to the DAG run in Airflow.





**M** Case # 47 - Possible DGA-activity over http(s) from 10.0.2.15 - interactive response

Created by admin Thu, Oct 14th, 2021 9:24 +02:00 1 alert

Details Tasks 1 Observables 1 xczjhkgdsadsa[.]ch

+ Add Task Show Groups

Group	Task
default	Analyst interaction for possible DGA-detection over http(s) required. Closed after a few seconds

The airflow DAG `complete_flow_test` is waiting on analyst feedback to decide how to proceed.  
Options are:

- Block domain `xczjhkgdsadsa.ch`
- No further action

This question is being waited on. Please use responder `DGA_Action_Block` or `DGA_Action_Pass` to deliver your feedback to airflow and trigger the next step.

Figure 17: Example of a task assigned to an analyst and answered to, by the said analyst.

#### 4.3.4 Lessons learned

##### 4.3.4.1 Platform Architecture

The architecture described in Figure 12 works well for our use case.

We observe:

- the SOC-analyst is presented with familiar interfaces;
- Airflow provides a summarizing view (the tree view, cf. Figure 16) that provides an overview of all automation in the SOC to the SOC-manager;
- the automation engine (in this case Airflow) "simulates" actions of an analyst through API calls, but in principle, the workflow can be carried out manually because known interfaces are utilized;
- the construct using Cortex' responders and Analyzers allows for a lot of flexibility when constructing workflows;
- the architecture has to support the automation and vice versa (e.g. there has to be an inventory of assets that contains information to help controlling the workflow and managing the risk of automation).

**We conclude that an architecture, such as the one we constructed for the DGA Showcase prototype, will be suitable to automate an arbitrary SOC workflow.**

##### 4.3.4.2 Automation and Heuristics

The tradeoff between automating as much as possible while managing the risk of automating response actions is difficult to resolve and involves more dimensions than meets the eye at first:

- One would like to have all-encompassing protection and response and especially protect high-value assets while at the same time keeping the load of manual work as low as possible in the SOC.
- The potential impact of carrying out an action that is unsuited or, in the worst case, plain wrong and disruptive on an important asset may weigh as heavy as not detecting a certain threat that's affecting said asset (imagine completely disrupting a webserver vs. removing a cryptominer that is running on just one core of the webserver's (virtual) CPU), at least for a while. In other words: The more important the asset, the higher quality detections, certainty and checks and balances are required.

This fits the well-travelled maximum of "automate the simple things to free up the time to concentrate on the difficult parts" well, but also calls for an extension of that statement, to include the asset importance as a factor.

**So we can draw the conclusion that the idea to "automate actions regarding low-risk assets to free up time to concentrate on alerts affecting high-value assets" should be used to refine goal #3 (G3), which can then be postulated as a generally applicable design goal for SOC automation.**

The automation itself should be as short as possible and contain the least amount of interruptions (i.e., interactions with humans) as possible to increase the throughput of alerts. Assuming one has a way to distinguish assets by their importance, this is an easy and efficient approach for low-risk/low-importance assets. The case of high-value/high-risk assets is trickier: Since there is a high risk associated with the asset, one wants to be sure not to miss any threats potentially affecting said asset.

In the case of the DGA Showcase prototype, this was resolved by the two-step heuristic involving a detection score and a preferred action for a given asset. The limiting factor here is that one does not want to miss any alerts/threats potentially affecting high-risk assets. This requires that the threshold of alerts to be considered potentially malicious is set rather low, for example, 50-60%. This significantly increases the number of alerts, the potential for false positives, as well as the number of automated response actions carried out on low-risk assets. It also increases the number of alerts regarding high-risk assets, but as there are significantly fewer high-risk than low-risk assets, there is - depending on the chosen threshold value - enough leverage to free SOC-analyst's resources to treat the alerts regarding high-value assets.

For this to work, some underlying assumptions need to hold:

1. There are a lot more low-risk/low-value assets than high-risk ones.
2. All assets get attacked roughly the same amount of time.

**We learn from this that the chosen heuristic still is very simple, is somewhat comparing apples (a detection score) and oranges (an assigned asset importance value) and has potential for improvement and tuning of the parameters.**

Involving an asset inventory into the workflow automation also has some interesting corner cases, such as the treatment of an asset that is not yet in the inventory. The DGA Showcase prototype solves this case by adding a previously unseen asset to the inventory, with a low importance score (we don't want to miss anything) and the action

preference "always ask". This may trigger a large number of alerts for the newly added asset in the SOC and mean an additional strain on the SOC team, but it also has the effect of adding pressure on the IT operations team to update the asset inventory. Thinking this a little further, it may even be feasible to start building a new asset inventory from monitoring the network in this way. In the case where an organisation does not have an inventory to begin with (something the author has seen in various network monitoring projects), this may well be an efficient and synergetic way to build one.

**We learn from this that synergetic effects are possible and worth pursuing when constructing the workflow automation in a SOC.**

#### 4.3.4.3 Response Actions

Choosing suitable response actions heavily depends on the type of alert for which automation is implemented. Even in the case of the DGA showcase there are several options with their pros and cons that have to be considered.

As the use case at hand is about detecting outgoing HTTP(S) connections to suspicious domains, an obvious choice for suitable actions is blocking said outgoing communication from the affected asset. This can be achieved in several ways:

- blocking communication on the client level;
- blocking communication on the network level.

Blocking communication on the client level can be achieved by native tools like iptables (in the case of Linux), or the Windows Firewall, but using some kind of endpoint detection and response (EDR) solution will be much more comfortable and give the handler much more control. An additional plus is that in the case of TheHive, there are already Responders that are able to interact with popular open-source EDR solutions like "Wazuh" or "Velociraptor". Action on the client level will likely be tied to the client that is affected by a given alert.

An alternative solution is blocking communication on a network security device like a firewall. While this approach can be restricted only to affect one single device, it will typically be used to block communication to (or from) a given destination for a whole subnet (meaning every asset residing within that subnet will reap the benefits or drawbacks).

In both approaches, it is not necessarily clear what exactly should be blocked. Talking about an outgoing HTTP(S) connections, there are at least the following two possibilities:

- blocking the given domain;
- blocking the IP address a given domain resolves to.

Both possibilities can make sense for different reasons. Blocking an IP can make sense if malware uses a domain generation algorithm to generate a sequence of URLs that all resolve to the same IP address. On the other hand, if a malicious URL resolves to an IP address from a range of services such as Cloudflare, Google or Amazon, there is a big possibility that the IP address is dynamically assigned and resolved and an IP

address that is hosting malicious services today will be hosting benign content tomorrow. The same argument holds for IP addresses that are shared between different websites or services. Blocking domains incurs less risk of affecting benign traffic and services, but in case of an actual malware running on an asset inside the network, the chances are higher that communication to the malware handler will resume (in case, the next domain generated by the domain generating algorithm is not detected by the detection mechanism used). Another drawback is an increased number of alerts in the SOC.

Another important difference to consider is the cost (in terms of computational load that is put on the infrastructure): Blocking domains on a general purpose network security device like a firewall can become prohibitively expensive because each connection has to be inspected for blacklisted domains. There are other specialised solutions, like DNS firewalls, but those are not available in some organisations. There is, however, no big difference between blocking domains or IP addresses on the client level regarding the cost.

**In the case of the DGA Showcase, we thus conclude that blocking both the domain that triggered the alert, as well as the associated IP address, on the client level is the best option.**

Blocking communication is not necessarily the best option, though. Depending on an organisation's architecture, size and prevalence of individual types of alerts for its network, it might be more reasonable to simplify the process by just informing the IT operations team through email about alerts (for example, in the case of high-confidence detections that rarely occur, or if the SOC does not have the authority to act upon alerts).

Finally, there is an additional layer to consider: Blocking communication to a malicious domain or IP does not answer the question whether malware is running on a given client inside the network, nor does it remedy such a situation. This suggests that another approach, where in a first step, only domains are detected and blacklisted. A second step would then require assets that generate a suspiciously high amount of alerts to be investigated further. In this way, the property of malware utilizing a domain generation algorithm for communication with its handler might be detected more precisely. This would have the benefit that one can keep gathering information about possibly infected hosts whilst already mitigating potential threats by blocking malicious communication as it occurs. Of course, the effort to implement such a two-step process is much more complicated and time and resource consuming than the simpler one-step process outlined and implemented above.

**We conclude from this that the intricacies of a given threat (such as how malware may use domain generation algorithms) can and should be reflected in the workflow design to improve detection and efficiency.**

As this will eventually lead to a candidate list of potentially malicious binary (that are causing communication), this would be a good handover point to the approach described in the next chapter (malware response automation platform).

## 4.4 Malware Response Automation Platform Prototype

Implementation of this prototype is motivated by repeating steps during malware analysis. The aim of this prototype is not to undergo a deep analysis of malware behaviour but rather to quickly evaluate a suspicious sample and automate the initial response. Moreover, we wanted to test the automation of the whole process, including the mitigation steps and discovering potentially infected machines in the organization's network.

Source codes of malware analysis platform prototype are located at the SAPPAN GitLab repository: <https://gitlab.fit.fraunhofer.de/sappan/malware-evaluator>.

### 4.4.1 Malware Evaluator Workflow and Architecture

The overview of system architecture is depicted in Figure 18 below. It shows the system component and their mutual interactions.

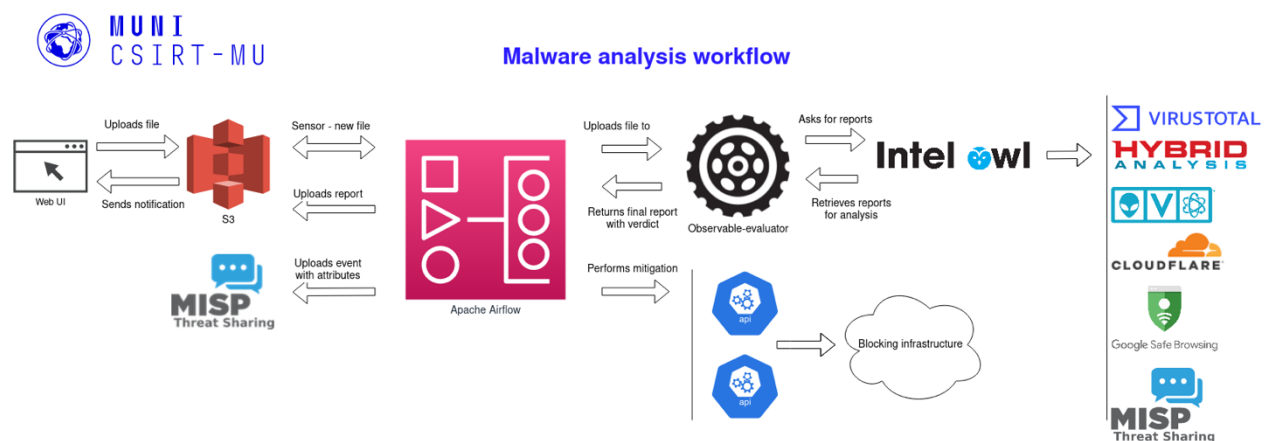


Figure 18: Malware evaluator prototype architecture.

#### 4.4.1.1 System Components

The system consists of several components, as depicted in the architecture overview picture above. Here, we provide a short description of the components:

- **WebUI** - WebUI is a simple React form application. It allows users to upload malware samples for evaluation and to read finished reports.
- **S3** - we are utilizing S3 compatible object storage MinIO as the storage of malware samples and reports. We are currently using three buckets: malware samples to be analyzed, malware samples that have already been analyzed, and final reports. The main advantages of MinIO are: it is lightweight, simple to interact with, and has official Docker and Apache Airflow support.
- **Apache Airflow** - Apache Airflow is an open-source Python-based workflow and orchestration platform. It offers an easy way of defining your own workflows thanks to many official operators (units of work) and hooks (connectors to external tools). In addition, it is also very easy to create new custom operators and hooks, if necessary. Apache Airflow contains the orchestration logic of our process. It periodically checks new uploads in the MinIO bucket and sends malware samples to Observable Evaluator, if there are any. Furthermore, it obtains analysis results and triggers necessary reporting and mitigation procedures.

- **Observable Evaluator** - Observable Evaluator is our custom tool used for malware analysis. It sends samples to IntelOwl (see next section) and collects analysis results. The main idea behind this tool is that IntelOwl simply uploads samples to third-party tools and returns their raw results. It performs neither “recursive analysis” (meaning analysis of observables, e. g., IP addresses, in malware sample) nor analysis evaluation (verdict or score). Observable Evaluator does these things: it collects reports returned by supported third-party tools via IntelOwl and tries to evaluate each sample based on predefined criteria. It also performs “recursive analysis”, which means it evaluates not only the given sample but also observables collected from the said sample.
- **IntelOwl** - IntelOwl is an OSINT solution integrating many analyzers (both local and external) and, therefore, allows users to collect information from many sources via a single query. It is not an evaluator, only a concentrator. This means that IntelOwl does not perform any additional analysis of the collected reports.
- **VirusTotal** - VirusTotal is a well-known website concentrating results from various antivirus software, as well as data from users. It can be used for the evaluation of both malware samples and observables such as IP addresses and domains. Its report provides a lot of information: collected results from anti-malware software, user votes, collected IoCs. From this point of view, VirusTotal is one of the more valuable sources. Observable Evaluator supports only public VirusTotal API, and it was not tested on the premium version.
- **Hybrid Analysis** - Hybrid Analysis is a free malware analysis service utilizing both static and dynamic analysis via Falcon Sandbox by CrowdStrike. Its reports are also quite rich, similar to VirusTotal. The main records in the report are given verdict, threat score, and list of connected domains (in the case of file report).
- **OTX AlienVault** - OTX AlienVault is a crowd-sourced computer security platform. It contains threat intelligence data about malware samples, as well as other observables provided by users. Its report is slightly limited in the sense that the most useful information is the number of pulses – information collected from users. We are also not able to collect observables from the malware sample via IntelOwl, only their numbers (such numbers also have a wide variety between pulses).
- **Google Safebrowsing** - Google Safebrowsing is an URL evaluating service provided by Google. We are obtaining only a direct verdict: malicious or harmless.
- **CloudFlare** - CloudFlare DNS is an URL evaluating service provided by CloudFlare. We are obtaining only a direct verdict: malicious or harmless.
- **MISP** - MISP is an open-source threat intelligence sharing platform. It is used both as a source of data, as well as the target of data, since in the final stages of the reporting phase we are creating MISP events.
- **Mocked Blocking Infrastructure** - to demonstrate the mitigation capabilities of the process, we are using custom mocked blocking infrastructure that is loosely based on the real-world infrastructure. It consists of IP blocking API and domain blocking API.
- **Mocked NetFlow Database** - we would like to be able to potentially query NetFlow data in an SQL-like fashion in the future. We are mocking this functionality with a simple single-table database in PostgreSQL. The provided sample contains anonymized and customized data, so we are able to showcase this feature on selected malware files. However, a real-world use case would be quite similar - to find other potential victims of specific malware.

#### 4.4.1.2 Malware Processing Workflow

The evaluation of malware samples is based on Apache Airflow orchestration, which consists of multiple DAGs that cover the whole process:

1. Starter DAG - when a user uploads a malware sample into the WebUI, it is persistently stored into an S3 bucket with new submissions. The Starter DAG continuously monitors the bucket, and when a new file appears, it launches Malware DAG.
2. Malware DAG - the main part of the workflow. It runs the analysis of a new sample by sending it to the Observable Evaluator. The evaluator performs the check of the file through the IntelOwl service (and hence through all connected tools). If it obtains any observables from the analysis, it repeats the analysis for each observable separately. When all results are collected, they are sent back to the Malware DAG. It then moves the malware sample from the S3 bucket with new samples to a bucket with processed ones, creates a MISP event with the collected information, and triggers DAGs to react to observables. After the subsequent DAGs finish, it creates a pdf report summarizing the findings and stores the report into an S3 bucket with reports.
  - Block multiple domains/emails/IPs DAG - three DAGs responsible for blocking all malicious observables through their respective tools or blocking APIs.
  - Observable DAG - performs a lookup for all communication with malicious observables in the traffic records.
3. When the final pdf report is stored in the S3 bucket, it is presented to the user via the WebUI for download.

#### 4.4.2 Deployment and Configuration

It is possible to deploy this project in two ways: on a local machine or on a local VirtualBox machine orchestrated by Vagrant and Ansible. Configuration stays the same for either way of deployment.

The whole project is configured in such a way that it is not necessary to do many configurations. It is necessary to insert API keys of external analyzers into the `.env_keys` configuration file, but all other configuration is strictly voluntary.

**Compulsory configuration** – the `.env_keys` file contains **secrets** part from the official IntelOwl `.env` configuration file. You can use the `.env_keys.template` as a starting point.

- `#secrets` – this section contains `MISP_URL` of used MISP instance. Change **ONLY** if you want to use your own MISP instance.
- `#Supported tools` – this section contains API keys for VirusTotal, Google Safebrowsing, Hybrid Analysis, and MISP.
- `#REST of IntelOwl supported tools` – IntelOwl provides even more tools; however, these tools are not supported by the project. It is possible to provide a module with your own implementation.

#### Local deployment using Docker:

- `docker-compose up`

### Local deployment using Vagrant + VirtualBox:

- `vagrant up`

After the malware evaluator is deployed, the endpoints listed in Table 8 are available.

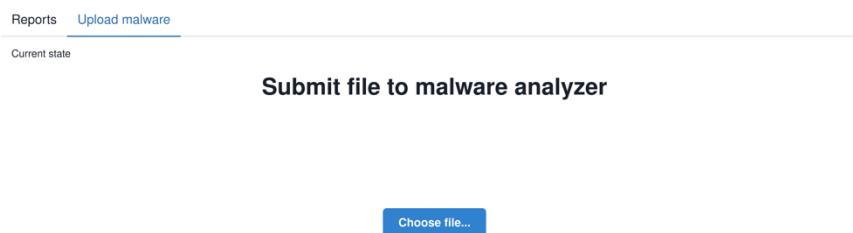
**Table 8: Malware evaluator endpoints.**

Tool	Address	Credentials
IP blocker	<a href="http://localhost:8996/api/ip-addresses">http://localhost:8996/api/ip-addresses</a>	None
Email blocker	<a href="http://localhost:8996/api/emails">http://localhost:8996/api/emails</a>	None
DNS blocker	<a href="http://localhost:8997/api/?action=list&amp;zone=black">http://localhost:8997/api/?action=list&amp;zone=black</a>	None
Whitelist	<a href="http://localhost:8998/api">http://localhost:8998/api</a>	None
Apache Airflow	<a href="http://localhost:8999/airflow/">http://localhost:8999/airflow/</a>	airflow:supertestovaciheslo
S3 Minio	<a href="http://localhost:9001">http://localhost:9001</a>	minioadmin:minioadmin
IntelOwl	<a href="http://localhost:9003">http://localhost:9003</a>	root:supertestovaciheslo
Malware uploader	<a href="http://localhost:9006/">http://localhost:9006/</a>	None
MISP	<a href="http://localhost:9007">http://localhost:9007</a>	<a href="mailto:admin@admin.test">admin@admin.test</a> :supertestovaciheslo

#### 4.4.3 System Usage Demonstration

The primary point of interaction with the user is the WebUI with malware uploader application, as depicted in Figure 19.

##### Malware Gateway



**Figure 19: Screenshot of Malware evaluator upload page.**

When the sample is uploaded, Apache Airflow DAGs will register and process it, as depicted in Figure 20, with details shown in Figure 21.



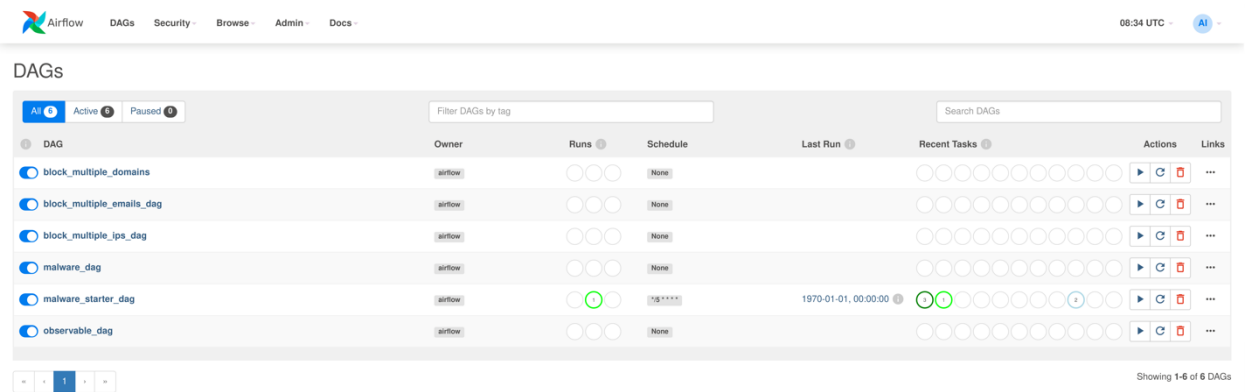


Figure 20: Screenshot of Malware evaluator DAGs in Apache Airflow.



Figure 21: Screenshot of malware processing DAG in Apache Airflow.

The progress can be monitored in the IntelOwl GUI, too, as shown in Figure 22.

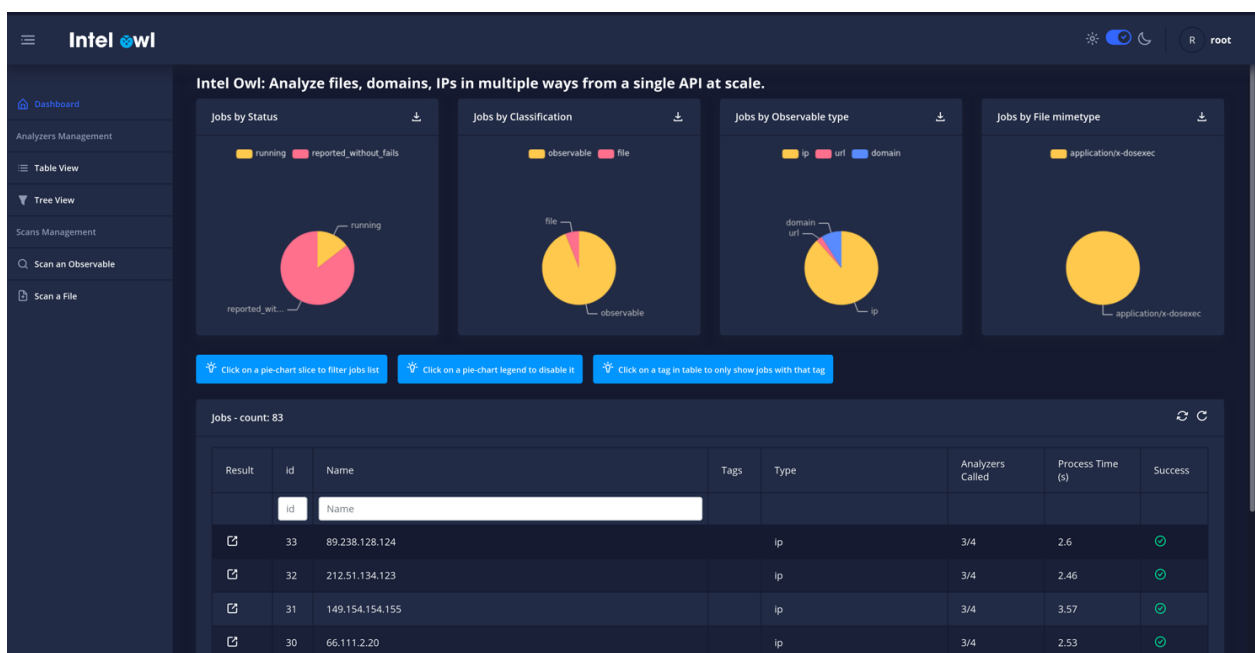


Figure 22: Screenshot of IntelOwl showing the progress of malware analysis.

After the analysis is complete, the WebUI offers it to the user, as depicted in Figure 23.

Reports [Upload malware](#)

## Report is ready

ed01ebfbc9eb5bbea545af4d01bf5f1071661840480439c6e5babe8e080e41aa.exe-report.pdf

Download

New upload

**Figure 23: Screenshot of Malware evaluator page with analysis report.**

The report contains a summary of the analyzed file and details about the detections and observables. The selected parts of the report are shown in Figures 24 to 26.

### Report for ed01ebfbc9eb5bbea545af4d01bf5f1071661840480439c6e5babe8e080e41aa.exe

#### Result

Malware-pipeline evaluated file ed01ebfbc9eb5bbea545af4d01bf5f1071661840480439c6e5babe8e080e41aa.exe as malicious.

#### Basic information

Name		ed01ebfbc9eb5bbea545af4d01bf5f1071661840480439c6e5babe8e080e41aa.exe
Hash	MD5	84c82835a5d21bbcf75a61706d8ab549
	SHA1	5ff465afaabcbf0150d1a3ab2c2e74f3a4426467
	SHA256	ed01ebfbc9eb5bbea545af4d01bf5f1071661840480439c6e5babe8e080e41aa

**Figure 24: Malware analysis report header.**

	ad1.exe
	8RK█(K1█+█).exe
	wannacry-ransomware
	WannaCry.infected!
Tagged as malicious	VirusTotal, OTX Alien Vault, HybridAnalysis
Tagged as harmless	

**Figure 25: Malware analysis report evaluation of sample.**

#### Found observables

Observable	Type	Malicious	Detected by	Tagged malicious	Tagged harmless
173.194.192.99	ip	True	VirusTotal	OTX Alien Vault	Google Safebrowsing
74.125.124.113	ip	True	VirusTotal	OTX Alien Vault	Google Safebrowsing
8.8.8.8	ip	False	VirusTotal		Google Safebrowsing, OTX Alien Vault
171.25.193.9	ip	True	VirusTotal	OTX Alien Vault	Google Safebrowsing
127.0.0.1	ip	False	VirusTotal		Google Safebrowsing, OTX Alien Vault

**Figure 26: Malware analysis report observables' evaluation details.**

#### 4.4.4 Evaluation of Automation Impact on Response

To evaluate the benefits of using the malware evaluator, we have carried out a measurement of the malware analysis process. We asked four professionals from the Computer Security Incident Response Team (CSIRT) of MU who are responsible for daily operations and incident response to go through the malware response process. Each analyst was given the same samples to process. Sample 1 was an MS Word document created by the test authors to serve as a benign sample. Samples 2 and 3 were taken from “theZoo - A Live Malware Repository” (<https://github.com/ytisf/theZoo>) and are samples of real malware.

The goal was to decide whether a sample is malicious, as sometimes users report suspicious files that are benign. For the malicious samples, the analyst was asked to obtain network Indicators of Compromise (IoC), i.e., communication of the malware with the C&C server or other network activity. For each IoC, it must be, again, decided whether the IoC is malicious or not, as the malware could communicate with legitimate services, e.g., Google public DNS. The next task was to block communication with malicious IoCs on the organization level. Finally, the analysts were asked to identify potentially compromised machines in the network by searching for the communication in network traffic records.

We designed the tests so that the analysts processed the samples manually first, and then the same sample using the malware evaluator. As the second process is almost fully automated, the analyst knowledge of the sample and IoCs did not play any role in the processing. The tasks were set so that they were following the same workflow for both manual and automated analysis. We measured the time needed to process each sample and list the results below. At the start, the analysis VM with malware evaluator is booted up. The planned deployment is that it is running continuously on a server as a service, and the platform will be available for incident responders. The initialization of the malware evaluator from scratch takes around 30 minutes (deployment on six years old desktop PC - Intel(R) Core(TM) i7-4790 CPU, 16 GB RAM, Windows 10 Pro version 21H1). After the system is up and running, the incident responder opens the ticketing system RTIR with the malware report.

##### 4.4.4.1 Manual Analysis

The evaluation of manual analysis of suspicious samples by human analysts is designed so that it follows similar principles as the automated workflow. The analyst only uses the selected tools and works with their outputs without any manual malware analysis or code inspection. The tasks they performed during the testing are as follows:

1. Download a suspicious sample from RTIR.
2. Upload the sample to VirusTotal.
3. Upload the sample to Hybrid Analysis.
4. Get results from both platforms and summarize network Indicators of Compromise, i.e., IP addresses, domains, URLs.
5. Look up the IoCs and classify them as benign or malicious.
6. Block the malicious IoCs.
7. Look up communication with IoCs in network traffic data.
  - Create a query to find communication with IoCs in the last 24 hours.

- Time for query evaluation is not measured, as the automatic workflow prototype does not have access to the same data source and the results would not be comparable.

The time measurement started with downloading the sample from the ticketing system (RTIR), and the times in the table correspond to the time when the analyst finished the given task. The comment summarizes important findings that affected the analysis and measurement process. The results are summarized in Table 9 below.

**Table 9: Measurements of manual sample analysis.**

		Upload to VirusTotal	Upload to Hybrid Analysis	Collect IoCs	Evaluate IoCs	Block IoCs	IoC traffic query	Total time	Comment
<b>Analyst 1</b>	Sample 1	00:28	01:17	08:55	x	x	x	08:55	Previously unseen file, doing complete analysis in the sandbox.
	Sample 2	00:27	01:18	03:52	06:20	07:04	08:32	08:32	Known malware. Taken results based on the hash match.
	Sample 3	00:24	00:59	02:40	03:45	05:01	06:10	06:10	Known malware. Taken results based on the hash match.
<b>Analyst 2</b>	Sample 1	00:20	01:07	01:42	x	x	x	01:42	Known sample. Taken results based on the hash match.
	Sample 2	00:21	00:46	02:12	03:04	x	x	03:04	Evaluated all IoCs as benign.
	Sample 3	00:15	01:02	01:53	02:20	04:01	05:24	05:24	Known malware. Taken results based on the hash match.
<b>Analyst 3</b>	Sample 1	00:20	00:56	01:20	x	x	x	01:20	Known sample. Taken results based on the hash match.
	Sample 2	00:12	00:30	03:15	04:29	x	x	04:29	Evaluated all IoCs as benign.
	Sample 3	00:10	00:26	01:37	01:57	02:53	03:30	03:30	Known sample. Taken results based on the hash match.

<b>Analyst 4</b>	Sam-ple 1	00:54	02:21	02:41	x	x	x	02:41	Known sample. Taken results based on the hash match.
	Sam-ple 2	00:16	01:24	02:24	04:31	x	x	04:31	Evaluated all IoCs as benign.
	Sam-ple 3	00:15	01:27	02:08	03:40	04:38	05:38	05:38	Known sample. Taken results based on the hash match.

#### 4.4.4.2 Analysis Using Malware Evaluator

The same analysts as in the previous section were tasked to analyze the same samples using the malware evaluator. The tasks had the same goals, but using only the evaluator:

1. Download a suspicious sample from RTIR.
2. Upload the sample to the malware evaluator.
3. Get network Indicators of Compromise (IoC), i.e., IP addresses, domains, URLs.
4. Classify the IoCs as benign or malicious.
5. Block the malicious IoCs.
6. Look up communication with IoCs in network traffic data.
  - o The evaluator prepares queries for data look-up, but the prototype implementation does not have a connection to production data and works only on the limited dataset. Under such conditions, the time measurement of query processing would be misleading, and was omitted.

The results are summarized in Table 10.

**Table 10: Measurements of sample analysis using malware evaluator.**

		Upload to evaluator	Collect IoCs	Evaluate IoCs	Block IoCs	IoC traf-fic	Total time
<b>Analyst 1</b>	Sample 1	00:25	01:29	x	x	x	01:29
	Sample 2	00:31	02:02	02:02	02:02	02:02	02:02
	Sample 3	00:15	02:17	02:17	02:17	02:17	02:17
<b>Analyst 2</b>	Sample 1	00:24	02:28	x	x	x	02:28
	Sample 2	00:14	01:58	01:58	01:58	01:58	01:58
	Sample 3	00:13	01:45	01:45	01:45	01:45	01:45
<b>Analyst 3</b>	Sample 1	00:20	01:24	x	x	x	01:24
	Sample 2	00:16	02:43	02:43	02:43	02:43	02:43
	Sample 3	00:12	02:52	02:52	02:52	02:52	02:52
	Sample 1	00:48	01:51	x	x	x	01:51

<b>Analyst 4</b>	Sample 2	00:45	01:56	01:56	01:56	01:56	01:56
	Sample 3	00:35	01:52	01:52	01:52	01:52	01:52

#### 4.4.4.3 Discussion and Lessons Learned

Downloading malware samples is complicated. It is blocked directly by web browsers. After an exception is set in the browser, the file is detected by antivirus and immediately deleted. The whole process should be contained in an isolated virtual environment.

Notes from manual analysis testing:

- Even though it took them approximately 10-20 seconds to complete, analysts complained about CAPTCHA during Hybrid Analysis file upload.
- Malware samples are often analyzed by other teams, and the results are stored on the platforms used, which can significantly speed up the process.
- Evaluation of whether an IoC is malicious or benign is not an exactly defined process, and the results vary between the different analysts.

Notes from malware evaluator testing:

- The analysts acknowledged that using the malware evaluator can lead to a reduction of possible mistakes as a human can overlook or forget an IoC.
- Analysts can upload the suspicious file and wait for the analysis results while doing other work. On the other hand, manual analysis requires their full attention the whole time.
- The decision of whether an IoC is malicious is the biggest weakness of the malware evaluator. During the testing, multiple Google services (e.g., public DNS server 8.8.4.4, website [\\*.google.com](https://www.google.com), Google mTalk, and others) were marked as malicious, and would be blocked if deployed in the production environment. Such a response would, however, cause more harm to the organization than the malware itself. These results also pose severe questions about the usability of open-source intelligence sources for incident response automation.
- The results are consistent across analysts, and the analysis times are similar with lower differences.

## 5 KPI Evaluation

In this section, we revisit key performance indicators (KPIs) defined for SAPPAN that are relevant to the content of task T4.4. The main goals of response automation are discussed in the overview section of the SAPPAN Response Automation Prototypes chapter and studied for the DGA showcase prototype. The most important KPI for response automation is the impact of the automation action on the response time and effort, as the preliminary evaluation experiments are included for the Malware evaluator prototype, showing significant improvement in the response time for the CSIRT members. The complete evaluation will be accomplished within the respective deliverable in WP6. Table 11 lists all the KPIs relevant to the works done within the scope of this deliverable and respective task T4.4.

**Table 11: Relevant KPIs for cyber security response and recovery automation.**

ID	KPI	Level of performance			SAPPAN objectives	Related WPs
KPI4	Performance overhead caused by SAPPAN innovations after enabling sharing features compared to SAPPAN in a baseline configuration.	< 80%	< 40%	< 10 %	O4, O5	WP3, WP4, WP5
KPI6	Processing delay between the arrival of sufficient data to detect the threat and report of the threat	>5min	>1min	<1min	O1, O2	WP3, WP4, WP5, WP6
KPI9	Time reduction of a CSIRT member to respond to threats (w/ and without SAPPAN)	>0%	>20%	>40%	O6, O7	WP4, WP6, WP7
KPI21	Does SAPPAN make it easier to interpret threat intelligence, support response and recovery actions, and provide accountability of recommendations? (assessed by the survey, 1 - no, 10 - yes)	>0	>6	>8	O7	WP3, WP4, WP7

## 6 Conclusion

This deliverable is the final version of our research on automating response and recovery steps for cybersecurity incidents. As part of this deliverable D4.7, we revisit the cybersecurity playbook specification and vocabulary with a focus on automated response actions.

We overview CACAO playbook specification and compare it to our developed vocabulary. We decided to progress towards the standardisation in the domain by contributing to the CACAO. We introduce our current contributions regarding CACAO standardisation activities, including our feedback on the specification and development of utilities for CACAO vocabulary usage and integrations.

Further, we overview workflow automation by providing an introduction to SOAR and an overview of the current automation tools. Then we describe our contributions regarding workflow automation on Apache Airflow, plus the development of a translator between CACAO and Airflow.

Later, we introduce our prototypes for response automation. We revisit the requirements, identified showcases (phishing and DGA), risk and confidence metrics for automation, and the general framework offered in the initial version of the deliverable (D4.6). The prototypes for phishing and DGA showcases are developed using Airflow and TheHive. We describe the general architecture and workflows and offer simple decision-making heuristics for automation of actions. We develop these prototypes as a learning opportunity to identify the problems and requirements for automating response and recovery steps. We discuss the outlook and lessons learned for the prototypes.

Besides, we develop a more mature prototype as a Malware response automation platform. We define the system architecture and describe the deployment and configuration steps in this deliverable. Additionally, we briefly demonstrate the prototype and evaluate the results by domain experts to analyze the performance improvement for CSIRT members. The preliminary results show a significant improvement in the response time. We will achieve the final KPI evaluation of task T4.4 within the scope of WP6.



## 7 References

- [1] OASIS Collaborative Automated Course of Action Operations (CACAO) for Cyber Security TC. CACAO Security playbooks specification v1.0. Committee Specification 01. 12 January 2021. Available online: <https://docs.oasis-open.org/cacao/security-playbooks/v1.0/cs01/security-playbooks-v1.0-cs01.html>
- [2] OASIS Open Command and Control (OpenC2) TC. Specification for JSON Abstract Data Notation (JADN) Version 1.0. Committee Specification 01. 17 August 2021. Available online: <https://docs.oasis-open.org/openc2/jadn/v1.0/cs01/jadn-v1.0-cs01.html>
- [3] MISP - Open Source Threat Intelligence Platform & Open Standards For Threat Information Sharing, available online: <https://www.misp-project.org>
- [4] MISP Taxonomies (overview and taxonomy repository), available at <https://github.com/MISP/misp-taxonomies>
- [5] MISP Galaxy, available at <https://circl.lu/doc/misp/galaxy>
- [6] Pedregosa et al., Scikit-learn: Machine Learning in Python, JMLR 12, pp. 2825-2830, 2011. <http://jmlr.csail.mit.edu/papers/v12/pedregosa11a.html>