TALLINN UNIVERSITY OF TECHNOLOGY
School of Information Technologies

Giorgi Okroshidze 215125IVCM

Comparison of SOC Workflow Automation Options for SMEs and Practical Impact Analysis

Master's thesis

Supervisor: Risto Vaarandi
PhD

Tallinn 2023

TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Giorgi Okroshidze 215125IVCM

KÜBERTURBE KESKUSTE TÖÖVOO AUTOMATISEERIMISE VÕIMALUSTE VÕRDLUS VÄIKESE JA KESKMISE SUURUSEGA ETTEVÕTETELE NING PRAKTILINE MÕJUANALÜÜS

magistritöö

Juhendaja: Risto Vaarandi
PhD

Tallinn 2023

Author’s declaration of originality

I hereby certify that I am the sole author of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Author: Giorgi Okroshidze

09.12.2023

Abstract

As the necessity for information security has increased, a Security Operations Center (SOC) has become a rather common occurrence in organizations of various sizes. However, many SOCs continue to face challenges: a high number of false-positive alerts, analyst fatigue, demotivation, etc. To combat these issues various SOC workflow automation tools have appeared on the market attempting to ease the work of SOC teams. However, most of these tools are either not advanced enough to deliver on their promises or are too advanced and out of reach for smaller organizations. This thesis aims to select and compare the features of available SOC automation options on the market that are accessible to small and medium-sized enterprises (SMEs). It will demonstrate the common characteristics of these tools and will illustrate how basic automation can positively impact the performance of an SOC using practical performance analysis. The outcome of this thesis will help organizations that are hesitating to switch to a more automated SOC by providing an understanding of the selected automation solutions and will encourage them to take the first steps.

This thesis is written in English and is 51 pages long, including 6 chapters, 23 figures and 0 tables.

Annotatsioon

Infoturbe nõudluse kasvu tagajärjel on Küberturbe Keskuse olemasolu erinevates suurustes organisatsioonides tavaline nähtus. Paljudel Küberturbe Keskustel on jätkuvalt raskusi suure arvu valepositiivsete hoiatuste, analüütikute kurnatuse, demotivatsiooni ja muuga. Nende probleemide lahendamiseks on turule jõudnud mitmeid erinevaid Küberturbe Keskuste töövoogude automatiseerimise tööriistu. Kuigi paljud tööriistad pole piisavalt arenenud, et tagada lubatud võimekust või on liiga võimekad ja pole kättesaadavad väiksematele organisatsioonidele. Selle lõputöö eesmärk on valida ja võrrelda erinevaid Küberturbe Keskuste töö automatiseerimise võimalusi turul, mis on kättesaadavad väiksematele ja keskmise suurusega ettevõtetele. Lõputöös demonstreeritakse nende tööriistade tavalisi omadusi ning illustreeritakse, kuidas lihtpärane automatiseerimine võib positiivselt mõjutada Küberturbe Keskuse sooritust, kasutades praktilist soorituseanalüüsi. Selle lõputöö lõpptulemus aitab organisatsioone, kes kõhklevad minna üle rohkem automatiseeritud Küberturbe Keskusele, tagades arusaama erinevatest automatiseerimislahendustest ja julgustades neid tegema esimest sammu.

Lõputöö on kirjutatud inglise keeles ning sisaldab teksti 51 leheküljel, 6 peatükki, 23 joonist, 0 tabelit.

List of abbreviations and terms

SOC	Security Operations Center
FP	False-Positive
SOAR	Security Orchestration, Automation and Response
AI	Artificial Intelligence
SME	Small and Medium-sized Enterprise
SIEM	Security Information and Event Management
IDS	Intrusion Detection System
IPS	Intrusion Prevention System
ML	Machine Learning
SANS	SysAdmin, Audit, Network and Security
IOC	Indicator of Compromise
API	Application Programming Interface
UI	User Interface
SaaS	Software as a Service information
N8n	Nodemation
NSA	National Security Agency
HTTP	Hypertext Transfer Protocol
URI	Uniform Resource Identifier
SMS	Short Message/Messaging Service
JSON	JavaScript Object Notation
IP	Internet Protocol
Regex	Regular Expression

Table of Contents

List of Figures

1 Introduction

In recent years, the landscape of cyber security has undergone various changes regarding the prevalence of both increasingly sophisticated cyber threats and defense solutions. As organizations and individuals have become more reliant on digital technologies and interconnected systems, the need for robust and responsive security measures has never been greater. This necessity has led to a more frequent adoption of Security Operations Centers (SOCs) in organizations.

Modern SOCs have the ability to keep track of logs from multiple sources and generate alerts according to implemented rulesets. This allows organizations to have a certain level of visibility over their infrastructure. Analysts working in an SOC team have the ability to investigate events and carry out incident response by initiating endpoint scans, wiping and isolating endpoints, and so on.

Although the rising frequency of cyber-attacks, data breaches, and software vulnerabilities has thrust SOCs into the spotlight, it has also shed light on common challenges SOC teams are facing in the modern cyber security climate. Analysts are overwhelmed by the volume of alerts, especially frequent False-Positive (FP) detections, which leads to burnout and demotivation. On top of this, the growing number of potential incidents requires a growing number of human resources, which is an issue because there is a shortage of skilled cybersecurity personnel.

To further evolve SOCs, and to deal with the issues outlined above, automation has been frequently proposed as a possible solution. In order to add automation to SOC systems various additional tools and approaches have been explored. One such solution is workflow automation which can be achieved by the implementation of Security Orchestration, Automation and Response (SOAR) tools, among other means. Further approaches have also been proposed involving deep learning and Artificial Intelligence (AI). That being said, the adoption of automation still remains a challenge for some organizations.

1.1 Problem Statement

As mentioned in the previous section, a common trend on the market regarding SOC workflow automation solutions is to guide organizations towards using SOAR tools. However, implementation of SOAR systems, and workflow automation in general, is not as straightforward for some organizations as it should be. There are multiple causes for this issue.

Firstly, there is a lack of standardized market definitions regarding SOAR, making it challenging for organizations to assess and select available SOAR solutions. Moreover, there is an absence of widely accepted frameworks and guidelines for workflow automation. For example, the phrase – SOAR – has multiple definitions according to the specific tool vendor: Some definitions include advanced AI and data visualization capabilities as an integral part of such systems, others only involve basic automation options. What should an organization expect from SOAR tools? What features are more important? Should investment be focused on higher-end options only or perhaps even simple automation tools are enough for most organizations' needs? This ambiguity can cause organizations to hesitate and scare away certain companies from pursuing investments regarding such automation tools.

Secondly, there has not been enough real-world analysis shared on how current entry-level SOAR or basic workflow automation implementations have affected the performance of SOC teams. What aspect of the team's performance can be improved? Is the difference worth the investment of time and effort?

All of these problems are even more applicable for small and medium-sized enterprises (SMEs) whose scope and budget for such a project is more modest compared to the budget required for some of the more popular, higher-end solutions available on the market.

1.2 Objectives and Scope of the Research

The goal of this thesis is to analyze the options for entry-level SOC workflow automation tools available to SMEs – organizations with fewer than 999 employees and less than $1 billion in annual revenue [1] . The feature requirements and comparison basis will be researched and determined for such tools. On top of that, the goal is to demonstrate what kind of effects correctly implementing the aforementioned automation features can have on the efficiency of existing SOC teams in such companies. This will be done on the basis of analyzing real-world performance data of the SOC team in company P.

The outcome of this thesis will provide a clearer picture regarding some of the entry-level automation tools available on the market. It will also demonstrate the benefits of using an in-house automation solution. Moreover, it will serve as concrete evidence and present the real-world improvements making a basic investment towards security operations automation can have on companies falling within the scope.

The scope of this thesis includes analyzing the features of entry-level security tools that offer workflow automation capabilities and are accessible to SMEs regarding their pricing and position on the market. The automation options that fall into the scope are ones that will work with and function in addition to existing SOC tools, such as Security Information and Event Management (SIEM) and will not act as a full replacement for them. Also, the scope includes analysis of an in-house scripting/software development approach that can replicate the automation features of these tools and achieve the same or even more advanced results regarding automation.

1.3 Novelty

The novelty of this work is that it will select and compare automation tools that have not been explored as frequently or in as much detail as the other options mentioned in the Literature Review section. This comparison will demonstrate what features can be expected from entry-level SOC automation tools, and how they can be implemented for security automation. On top of that, this thesis will provide performance statistics, since real-world analysis of SOC performance after applying basic automation in a company within this scope was not observed when performing the literature review. The thesis will also demonstrate that existing entry-level SOC automation tools have shared shortcomings that might prevent their application for advanced automation tasks. For this reason, developing an in-house automation solution might be beneficial if such functionality is needed, and the thesis describes one such production-grade SOC automation system.

1.4 Research Questions

The following research questions were identified:

1. What features should be expected from entry-level SOC workflow automation tools and how should these features be prioritized?
2. What tools can be selected that provide the aforementioned features and also fall into the scope of this work?
3. How do these tools compare with each other according to the prioritized features?
4. How can a basic utilization of the automation features these tools provide affect an organization's SOC team's performance?

2 Literature Review

2.1 SOC Components and Architecture

Regarding the architecture of an SOC [2] , it can be centralized, meaning all the data is aggregated into one SOC, distributed, where data is not aggregated in one place and is fetched from multiple locations, or decentralized, which is a combination of the previous two architectures.

Moving on to the technology stack, it includes an array of different interconnected security tools and monitoring solutions. These tools may involve an Intrusion Detection System (IDS), Intrusion Prevention System (IPS), SIEM system, firewalls, antivirus software, and various other network and endpoint monitoring solutions [3] . Additionally, threat intelligence providers and vulnerability assessment tools allow analysts to stay informed regarding newly emerging threat indicators and system/code vulnerabilities. All these technologies work together to effectively collect and analyze vast amounts of data and logs, providing an organization's SOC team much needed visibility over their company.

Along with basic tools, recently there have been experiments and proposals for SOCs to utilize more advanced technologies like Machine Learning (ML) to elevate simple tasks. For example, in a 2019 study by Gupta et al. [4] ML is proposed as an efficient method for alert classification and prioritization. Another example for using ML is demonstrated in a study by Van Ede et al. [5] where an algorithm is used to find correlations between event sequences from a specific device. The system DEEPCASE attempts to find context between events and decide if they are malicious. Moreover, a 2017 study by Feng et al. [6] demonstrates how the ML approach can be used to potentially lower the amount of FP alerts in a system. This would help analysts combat alert fatigue.

On top of the technical tools, personnel also play a pivotal role in a SOC's architecture. Typically, important SOC roles involve different tiers of analysts and dedicated managers, all bearing distinct responsibilities [7] . Tier 1 analysts are triage specialists, responsible for investigating, classifying, and prioritizing generated alerts. They decide whether or not an alert is a FP and are the first line of defense. Tier 2 analysts share some responsibilities with tier 1; however, they deal with more complicated alerts that are escalated from Tier 1 and require further investigation, containment, or response. Tier 3 analysts are the highest level analysts. They deal with the highest priority incidents and are also responsible for identifying security gaps, tuning existing alerting rules, and optimizing monitoring solutions. Finally, there is the SOC Manager role, which is in charge of supervising the team, creating processes, etc.

Along with the technology stack and personnel, pre-defined processes and incident response playbooks are integral to the correct operation of an SOC [8] . These pre-defined workflows and steps ensure that security incidents are managed effectively. Incident response playbooks are a very important part of an SOC, containing detailed steps for the analyst to follow regarding specific security alerts and how to initiate correct incident response. A playbook can be more general or more specific depending on its application. The approach differs according to the type of SOC services being offered. It is common for SOCs that offer their services to multiple customers to have separate playbooks to manage each customer's unique requirements.

The final key element of an SOC is collaboration. Collaboration between the different tools being used, collaboration between the SOC team, the different analysts tiers [9] , and other departments within an organization, (such as the Information Technology department, legal, etc.) and collaboration between the organization and its various customers.

To bring the basic process together:

- SOC architecture can be centralized, distributed, or decentralized.
- Various security tools are responsible for generating, receiving, accumulating, and allowing interaction with company data and logs, providing much needed visibility. Alerts are generated by these tools.

- The SOC team is comprised of various roles and is responsible to deal with the generated alerts. Most of the alerts are investigated by Tier 1 analysts.
- Alerts are dealt with according to pre-defined incident response playbooks which detail the steps an analyst must take for a given alert.
- A potential incident is solved once there is regular successful collaboration between the SOC team, the security tools at their disposal, the documented processes, and the necessary departments within or even outside of the organization.

2.2 Common Challenges SOC Teams Face

Having described a typical SOC system, we must also describe the challenges that such systems typically face. No system is ideal, and SOCs also face issues that are often multifaceted and can significantly impact their ability to protect against cyber threats.

The annual SOC surveys that are performed by SANS (SysAdmin, Audit, Network and Security) are reliable sources for observing the challenges SOC systems have been facing throughout more recent years. Figure 1 demonstrates the greatest SOC challenges faced by organizations in 2023.

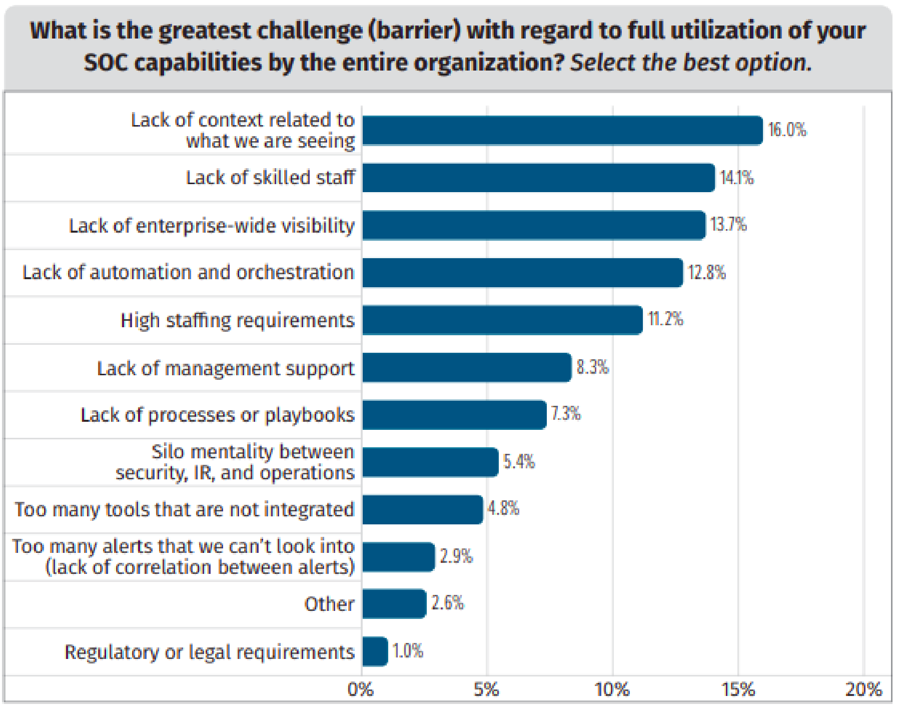
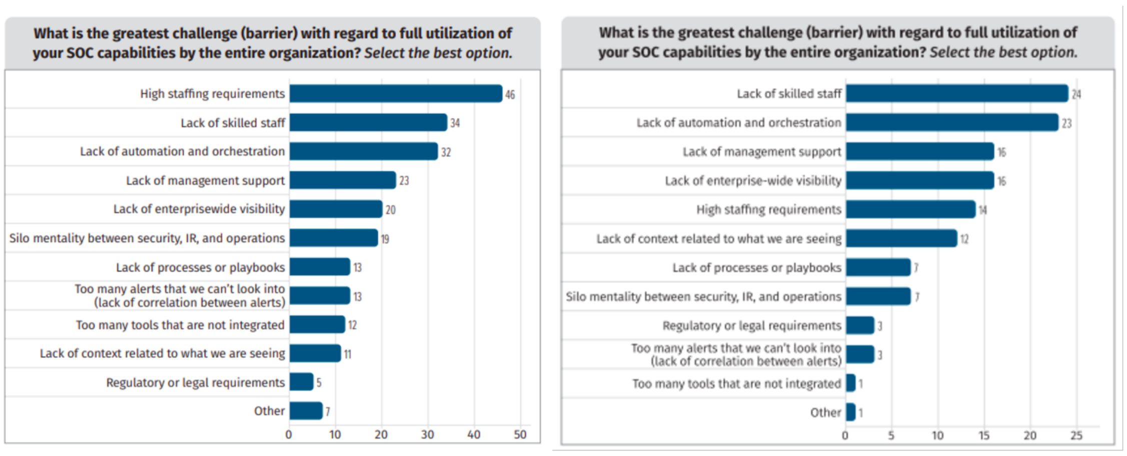


Figure 1: SANS Survey, Key Challenges, 2023

According to the 2023 SANS survey [10] , 16% of analysts reported that they did not have enough context related to the data they received and analyzed. Following this, the main challenge was lack of skilled staff with about 14.1%. On third place was lack of enterprise-wide visibility with 13.7%, and on fourth place lack of automation and orchestration was reported as an issue with 12.8%. In is important to note that the difference between percentages occupied by these issues is not very high.

Taking a look at previous years' SANS studies from 2022 [11] (Figure 2 a) and 2021 [12] (Figure 2 b), it is clear that the most important challenges organizations were facing are similar to the challenges in 2023: high staffing requirements and lack of skilled staff, along with lack of automation and orchestration. The 2021 survey stated that a possible solution to these challenges is to be "able to automate the mundane tasks."



(a) (b)

Figure 2: SANS Survey, Key Challenges, 2022 (a) and 2021 (b)

The same issue was also discussed in a 2020 study by Vielberth et al. [2] in which it is stated: "Regarding the people working in a SOC, we see a major challenge in recruiting and retaining staff." The challenge of retaining staff can in many cases be related to the many repetitive tasks Tier 1 analysts have to perform regularly and the burnout this kind of intense routine causes. The study also points out that there is a lack of industry-specific standards and guidelines which "prevents various SOC components from advancing since a common baseline of the status-quo has not yet been agreed upon." Couple this issue with the fact that most automation tools are not well detailed or sometimes advanced enough to deliver on their promises and the picture is clear as to why many companies, especially ones with a tighter budget, have hesitated when it comes to implementing automation solutions.

A 2022 study by Alahmadi et al. [13] also describes how analysts experience fatigue because of multiple security systems frequently generating alerts that are not truly threats or are not malicious. It also states that many alerts from these security tools lack the context needed for the analyst to decide if the alert is a genuine alarm. Moreover, many of the analysts that participated in the study revealed that many AI or ML driven solutions did not perform that well and generated their own FP alerts, adding to the analysts' struggle. The issue of frequent FPs by ML solutions in production environments is one of the reasons the current thesis focuses on the use of basic automation tools for easing the work of SOC teams.

2.3 Automation as a Potential Solution

According to multiple sources, automation of mundane and repetitive SOC tasks holds the promise of addressing several of the aforementioned challenges faced by SOC teams. Some even go so far to say that automation is one of the "key elements to consider when designing a modern SOC" [14] . Stating that automation tools can help streamline operations by reducing the workload of security analysts and improving response times. Other studies have shown that some operators believe even automating simple investigation steps can have a positive impact [15] . An example of this kind of basic automation is to simply query VirusTotal and enrich Indicators of Compromise (IOCs) in a given alert. A combined effort of such steps helps provide more context to an alert, reduce burnout, and helps organizations retain talent.

2.3.1 SOAR tools

SOC automation cannot be discussed without mentioning SOAR tools. SOAR as a term was created in 2017 by Gartner [16] . It has become a very popular market term describing tools that offer varying levels of workflow automation, along with other features. Regardless of the term's popularity, most SOAR tools do not follow a single pre-defined architecture or feature-set. There is little to no uniformity when it comes to solutions offered by different vendors [17] . Some vendors offer very advanced features as basic parts of a SOAR platform such as AI, ML, data visualization tools, and so on.

Others offer very basic automation options. In many cases these vendors do not live up to the expectations they create [2] . This makes navigating the market regarding SOAR tools fairly difficult for some organizations. Why are the solutions so different from one another?

In a recent study [16] orchestration is defined as follows: “orchestration specifically refers to the integration of separate tools with different functions into a single platform to streamline and accelerate the investigation of a threat.” Another survey presents other definitions of security orchestration [18] , which include connecting existing security tools together via API (Application Programming Interface) to streamline incident response. Accordingly, it would seem that orchestration does not involve workflow automation in itself. Conversely, further definitions in the same survey bundle automation as part of security orchestration. This is why the term SOAR is so complicated in today’s market and why vastly different tools still share the same name. Here is a broad description of SOAR that can cover a bigger range of solutions:

SOAR tools typically combine separate local or remote security systems into one, use a pre-defined ruleset to enrich and add context to an alert, decide what steps to take for investigation or escalation, and allow for sufficient automated response when necessary. In some implementations these steps may be achieved with a visual interface or without. Solutions described as SOAR tools may include additional AI, Deep Learning, visualization options, ticketing systems, or may only offer very basic automation features.

That being said, is SOAR effective? In his master’s thesis [19] Lalos states that utilizing automation in security investigations and procedures “can drastically reduce response time from the analysts, making them more effective.” On top of that, in the work by Brewer [20] it is also stated that SOAR systems can reduce mundane SOC tasks and high day-to-day workload. Moreover, SOAR tools were proven to make investigations more efficient in a survey that evaluated operator performance in multiple test environments [21] .

2.4 Similar Works and Research Gap

During the literature review process similar works to this thesis were identified. The goal of some of these works was to also compare several SOC automation options, mainly concerning SOAR tools. Some of the following works also demonstrated the effects these tools could have in use. Both of these objectives are in line with the goals of this thesis.

In a recent work from 2021 [16] , the authors decided to evaluate 11 SOAR tools since there was no research that had done so at the time. To achieve this, SOC operators were asked to complete multiple surveys. The first survey requested that the operators choose which aspects of SOAR tools are the most important. Following this, the operators were asked to evaluate SOAR tools after watching demonstration and overview videos. Finally, the tools were scored. However, the tools that were evaluated were not named and were left anonymized: “as a prerequisite, SOAR vendors agreed to providing licenses and support for this study in exchange for anonymity of their results and participation.“

A 2023 study [21] asked operators from US Navy SOCs and other experienced SOC teams to evaluate SOAR tools that were set up in a test environment. The study revealed valuable results regarding SOAR tools and what kind of positive and negative impacts they can have on investigations. However, as in the previous study, the tools were not named: “As required by the AI ATAC Challenges, SOAR tool contenders involved in this evaluation must be held anonymous.”

Having said this, there were also works found that named the SOAR tools that were analyzed. In 2022, Lalos performed an in-depth comparison of three SOAR tools [19] :

- Cortex XSOAR
- Splunk SOAR
- Siemplify SOAR

Another work that named the SOAR tools that were compared is from 2021 [22] . It selected and briefly compared the following tools:

- IBM Security SOAR
- PaloAlto Cortex XSOAR
- Simply
- Security Vision SOAR

Another work by Kantola [15] demonstrated how SOAR can be used with third-party sources like VirusTotal to enrich IOCs during investigation and studied how analysts reacted regarding its implementation in their workflows. This work was successful in demonstrating this basic automation approach, however, the other capabilities of this SOAR tool or other real-life improvements were not discussed.

After looking through these studies a clear research gap emerged: entry-level SOAR tool functionality and features have not been analyzed in detail since most of the studies dealt with more advanced, more expensive options. Furthermore, the surveys that went in-depth into evaluating SOAR tools chose to present the results as anonymized data. Because of this, there is uncertainty regarding available options of competitive entry-level tools and their feature-sets. On top of that, the actual impact these solutions can have on real-life SOCs has been rarely discussed.

3 Selection of Automation Tools

3.1 Choosing the Selection and Comparison Criteria

Before suitable automation tools can be selected for analysis, this section will delve into the fundamental considerations by which the efficacy and appropriateness of these tools will be evaluated.

3.1.1 Pricing of the Tools

Firstly, as this analysis is for entry-level tools that will be accessible to companies on a tight budget, one of the main considerations for selection is the price of the tools. Studies show that some small-to-medium size businesses have struggled with IT budget constraints recently [23] . In an aforementioned survey [10] although statistics are not divided according to company size, it is shown that the estimated annual budget for cyber-security related hardware, software, human capital, etc., was unknown for 20% of the companies that were questioned. Apart from this, about 35% of companies had an annual budget of less than 500,000 USD for all of their security-related needs. As many organizations are more willing to try out and take risks with free-to-use open-source tools, such solutions will naturally have a small edge during this selection process.

3.1.2 Basic Functionality of General Security Automation Tools

Secondly, to be eligible for analysis in the following section, the chosen tools must provide certain basic functionality that is commonly associated with security automation tools. To identify the necessary functionality multiple related works were analyzed.

Some of the defining aspects of SOAR tools were detailed in a recent work [21] , the main aspects being the following:

- Data Ingestion – from a wide variety of internal and external sources.
- Data Correlation and Prioritization – including data enrichment, highlighting in the tool's User Interface (UI).

- Process Automation – automation of alert triage and incident response processes via configurable workflows and playbooks.

In a work about SOAR platform analysis for SOCs [19] three SOAR tools were compared. According to the main comparison categories, it is clear that for a tool to be considered it must offer at least the following features:

- Data Enrichment
- Data Correlation and Forensic Analysis
- Data Visualization
- Playbooks and Automation

In the work by Norem et al. [16] SOC operators were asked to evaluate SOAR tools. Firstly, operators prioritized the ability to create playbooks and workflows that are easy to manage. In second place, was the ability to automate common tasks. Third place was given to providing tickets with additional context. On fourth place was alert prioritization. This was followed by the ability to ingest data from various sources. After the survey, it was decided by the operators that the most important aspects of a SOAR tool were the functionality of its playbooks and the ability to automate common tasks. These features were followed by ticketing and alert prioritization.

A very detailed work by Islam from 2020 [17] describes the following as part of high-level SOAR architecture:

- UI Layer: Provides interactive dashboards for visualization of data, Command Line, and Visual Interfaces basic interaction with the tool.
- Orchestration Layer: Responsible for allowing coordination and modification of configurations, monitoring status of processes and executions, triggers, automating execution of incident response playbooks, etc.
- Semantic Layer: Allows interaction with tool knowledge base according to a query language.

- Data Processing Layer: This layer processes the heterogeneous data from different sources and allows for data analysis.
- Integration Layer: This layer is designed for seamless management and integration of different security tools, via API or other means.
- Security Tool Layer: This is the various separate and independent security products that are connected with each other through SOAR.

The work by Gibadullin and Nikonorov [22] identified three important modules in SOAR tools:

- Incident module: Has the ability to create incident types, ability to provide a custom response, ability to assign an incident to an employee, etc.
- Scripting module: Has the ability to create, modify, delete scripts, ability to run scripts manually or on a schedule.
- Integration module: Has the ability to handle different types of integrations, likely relating to security tools.

After analyzing the aspects discussed in these works, some focusing on more specific functionality and others focusing on more unique features, it is clear that there is overlap between commonly expected and valued aspects of SOAR tools. It was decided that many of the aforementioned SOAR features will be excluded moving forward. One of the main reasons for removing these features as selection criteria is the fact that the scope of this thesis targets automation tools that do not have to necessarily replace basic parts of existing SOC infrastructure. In the scope of this thesis, automation is an addition on top of the existing infrastructure. For instance, this means that the querying and visualization capabilities of a SIEM tool or a ticketing system that are likely already in-place in typical SOCs do not have to be matched or replicated by the automation tool that will be selected. In addition to this, some of the outlined features can be considered already a part of other functionality that was discussed. For example, data enrichment, prioritization, scripting, incident response can all be considered sub-categories of playbook and workflow automation capability. Similarly, threat intelligence can be considered to be a part of data ingestion or data enrichment capability. On top of these aforementioned features, a main aspect that was not discussed as often is version control. For individuals developing software or generally writing code, version control is very important for their process [24] . For many advanced SOAR workflows, it is necessary to write snippets of code, or even complete scripts to correctly parse data, format output, and so on. Regardless of this, proper version control options seem to be missing from many SOAR tools [25] , often lacking in the ability to rollback or properly keep track of changes done to a workflow [26] . This thesis wishes to separately outline version control as an aspect of the selection criteria. Version control can be with regards to code snippets in workflows or with regards to general configurations of the workflows and the tool.

Because of all the reasons mentioned in this section, the aspects that will be focused on during tool selection and comparison steps are the following:

- Data Ingestion (including the various methods of receiving and fetching data)
- Workflow Automation (including data manipulation, scripting, logic statements, etc.)
- UI
- Orchestration (including configuration management, status monitoring, combination of different sources, triggers and schedules, etc.)
- Version Control

3.2 Tool Exclusions, Choices, and Reasoning

Firstly, this section will discuss the SOAR tools that were mentioned during the Literature Review section. Although these options mainly fulfill all the feature related aspects mentioned above, and in many cases offer features that go above and beyond the outlined scope of automation, they also have drawbacks in their pricing.

According to the price calculation feature on IBM's website, the pricing of IBM Security SOAR partially depends on the number of authorized users allowed to use the product [27] . Of course, this is just used for a rough estimation and the actual price also depends on the specific configuration, add-ons, etc. Even so, the estimated cost for a SaaS (Software as a Service information) solution with 11 authorized users amounts to a monthly cost of 22K USD, amounting to more than 250K yearly. 11 was selected as the number of users during the calculation of this price because according to the 2023 SOC SANS study, the most common SOC team size is between 11-25 [10] . Price estimation is harder for the following products: Cortex XSOAR, Splunk SOAR, and Siemplify SOAR (now known as Chronicle SOAR after acquisition by Google). However, sources indicate that prices are not low [28] [29] . All of these products offer a free tier or community edition which can be used for longer than the trial period, usually without time limitations or additional cost [30] . That being said, the functionality of these tools is limited instead, including restrictions on the number of API calls, playbook executions, number of integrations, and so on. For example, for Cortex XSOAR automation commands are limited to 166 command calls daily and incident history is restricted to the last 30 days. Such limitations indicate that these community editions are mainly useful for testing and experimentation purposes. Finally, Security Vision SOAR was also discussed in a few sources, however, this tool has very limited adoption outside of Russia [31] . Most of the organizations that have implemented and reviewed the tool are located in Russia, and descriptions of this tool are rarely found outside of Russian sources.

Moving on to the tools that were chosen for analysis, during research into low-cost, entry-level automation solutions, the following options were discovered:

- Shuffle
- Nodemation (N8n)

Shuffle is an open-source workflow automation tool that offers both cloud and self-hosted solutions. The tool has a free option that includes unlimited apps, workflows, users, with a limit of 10k app runs every month [32] . Apps in Shuffle are the nodes that run in workflows and perform various functions. For increasing the app run limit, one can use the Enterprise version of the tool, which starts from 100k app runs each month for 180 USD. This can also be scaled proportionally with the price according to the organization’s needs. According to Shuffle’s website, 100k app runs per month can support around 500 assets.

N8n is another almost open-source workflow automation tool that also offers cloud and self-hosting options [33] . The pricing of the tool is 20 euros per month for Starter edition, 50 euros per month for Pro, and has a custom price per organization for the Enterprise version [34] . According to a recent whitepaper from N8n, their pricing is affordable and competitive compared to other SOAR tools [35] .

Both of these tools fulfill the selection criteria for the basic tool features / aspects outlined in the previous section. They fit into the scope because of their entry-level price and because they are made to combine and work alongside existing security tools instead of replacing them fully. For testing, the free cloud version of Shuffle and the Pro cloud version of N8n were utilized.

4 Detailed Comparison of the Selected Tool Features

This chapter contains a side-by-side comparison of the selected automation tools: Shuffle and N8n. The features offered by these tools are described in-detail, along with various advantages and disadvantages of each tool's approach. This comparison reveals specific functionality that can be expected from automation tools in the entry-level category. Moreover, in chapter 4.3 the shared drawbacks of these tools are summarized which motivates the implementation of similar feature-sets in a custom in-house SOC automation solution.

4.1 Shuffle

Shuffle is an open-source SOAR project started in mid-2019 as a hobby [36] . It was based on the structure of National Security Agency's (NSA) WALKOFF SOAR, which was discontinued in 2019 [37] . Shuffle is a low-cost product that aims to provide the basic capability of connecting multiple tools together, allowing easy data ingestion and flow between processes and allowing automation of basic workflows. Because of its nature as an entry-level and open-source tool it does not provide complicated AI or ML features and it does not aim to replace a SIEM or similar security tools. However, this results in a product that is focused and delivers on the features it promises.

Here are some of the basic components of Shuffle [38] :

- Apps – are basically integrations with other platforms' APIs. Shuffle uses OpenAPI for easy app creation. Apps can be imported, exported, or shared. Many pre-existing apps are available right after creating a Shuffle account. Moreover, custom apps can be created via the tool's visual interface with no need for writing code. Shuffle also has an app that allows the user to run basic functions like printing a statement, filtering a list, looping, etc.

- Workflows – are what connect different apps/nodes and include automation logic. For example, a workflow can be created to automate the steps in an incident response playbook. They can be comprised of the following components:

  - Apps – can be used in workflows to fetch data via API endpoints or to run basic functions on data, like parse it, filter it, and so on.
  - Scripting – scripts written in Python can be injected into a workflow to cover the cases where pre-defined app functionality will not be enough to cover a task.
  - Variables – can be set in a workflow to hold data. There are Execution Variables that are temporary and set during runtime. There are Workflow Variables that static and set before execution. There is also the Execution Argument that is passed into the workflow when it is triggered.
  - Triggers – allow the user to run a workflow. Here are some of the main trigger options:
    - Webhook – runs when an Hypertext Transfer Protocol (HTTP) request is made on the workflow's endpoint.
    - Email – runs when an email is received in a connected Gmail or Office365 account.
    - Schedule – a workflow can be set to run on a schedule, similar to a cron job.
    - Subflow – a workflow can be inserted into a parent workflow as a subflow which will run when the parent is triggered.
    - Etc.
  - If conditions – determine which app/node will run during a specific execution of a workflow.
- Storage:
  - Datastore – Permanent key-value storage for useful data for automations.

  - Files – files can be uploaded to Shuffle and plugged into workflows for analysis.
  - Encrypted Authentication – authentication credentials are stored mainly for apps and are encrypted for safety. Decryption happens automatically during runtime.

Following this short overview of the tool’s features, this section will transition into in-depth analysis of each aspect that was outlined.

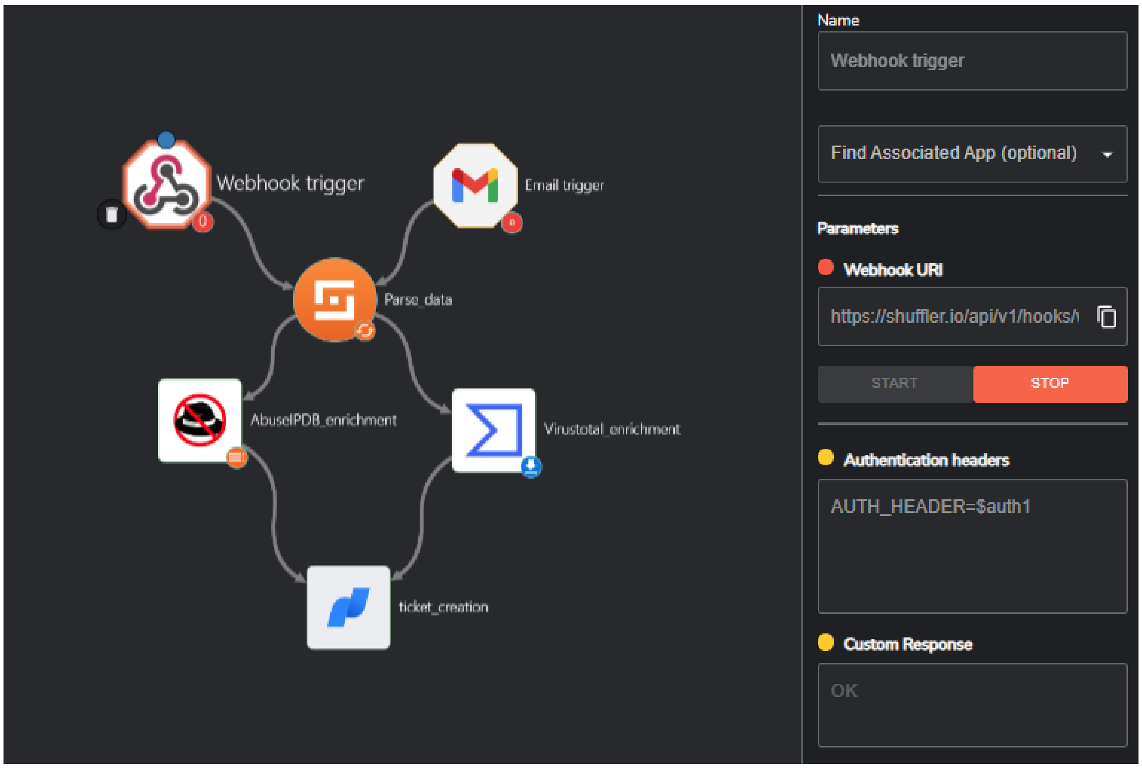
4.1.1 Data Ingestion

As mentioned above, there are many ways for data to be ingested into Shuffle. Firstly, data can be ingested into workflows via triggers:

- A workflow can receive data in the form of HTTP POST or GET requests from outside sources if Webhooks are correctly setup. They will have their own Webhook Uniform Resource Identifier (URI) which will receive these requests. Moreover, these endpoints allow the option to set up authentication, only allowing authorized sources to send data.
- The email trigger also sends data to Shuffle. This can include the email body, headers, and attachment info.
- Data can also be ingested via the User Input trigger which will run a workflow when a user manually chooses to via an Email or SMS (Short Message/Messaging Service).

Apart from triggers, data can be ingested into the tool using the various apps. Since many apps allow users to call any API endpoint they wish, they allow for virtually limitless external sources of data.

To demonstrate this capability, in Figure 3, a workflow is visualized that ingests data from two triggers, a Webhook trigger and an Email trigger, sends the data for enrichment via two remote sources, and creates a ticket with all the data combined.



Name: Webhook trigger

Webhook URI: https://shuffler.io/api/v1/hooks/

Authentication headers: AUTH_HEADER=$auth1

Custom Response: OK

Figure 3: Shuffle – data ingestion demonstration

4.1.2 Workflow Automation

As demonstrated above, triggers can be used in workflows to start workflow execution and to inject data into the workflow. Once input data is in the workflow, it can be handled in multiple ways.

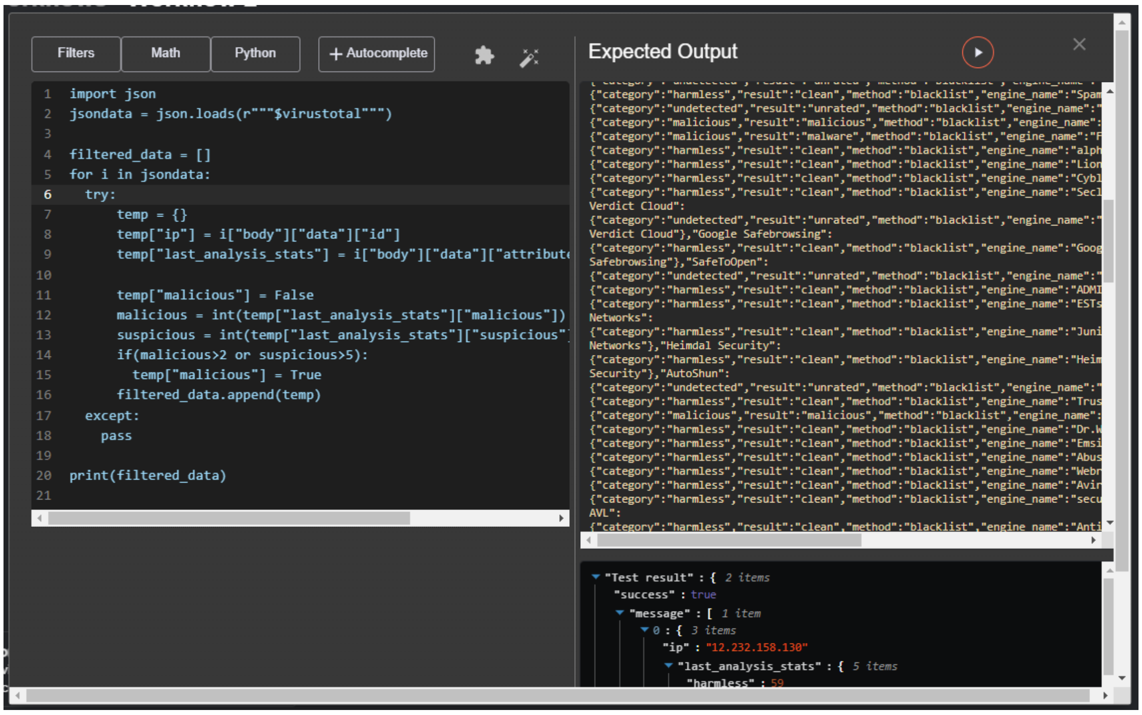
Firstly, data will be passed between apps/nodes. Mainly, Shuffle outputs data as a string, if it is a JSON (JavaScript Object Notation) string, it will be recognized as an object but still output as a string. The user has the option to pass the whole string to the next node as the execution argument, or to pass part of the string. If the object is a list, users can choose to iterate over the list entries – *(.#)*. For this, the user can use *$var.#* operator to perform a for-each loop on the string and pass each value separately to the next node. As an example, if a workflow contains the variable *$input*, which is a list of objects with attributes – *name, date, and IP (Internet Protocol)* address – this variable can be passed to a VirusTotal app for enrichment as a loop: *$input.#.ip*; This means a separate request will be made using the app for each input. The output of this node will be a list of VirusTotal responses for each IP. This is the main way looping is achieved in Shuffle outside of scripts. The following are the main options for this loop operator:

- *.#* – loop through the entire list.
- *.#0* and *.#1* – run only the first or second element of the list.
- *.#0-1* – run the first AND second element of the list.
- *.#.data* and *.#0.data* – run the specified list, get value of key “data” for each.
- *.#max.data* – run only the last element, get value of key “data” for it.

Secondly, once data is passed between the nodes, it will be manipulated and changed. An option to manipulate data is through the pre-existing “Shuffle Tools” app. This can be used to parse input data, execute custom Python scripts, filter lists, create files, etc. A common use-case regarding input data is to handle a list or an array of objects. The “Shuffle Tools” app offers the following pre-defined actions for handling list data:

- Filter List – filter according to object attributes. If a specific attribute equals an expected value, keep that list item.
- Parse IOC – filter out IOCs from strings, arrays, dictionaries, etc. The output will be in the format of a list of objects that contain attributes: item type and value.
- Parse List – turn a string into a list according to a splitter string.
- Add List to List – add two lists together.
- Merge Lists and Merge JSON objects – merge two dictionaries or JSON objects.
- Diff Lists – return an array of keys that are unique from both lists.
- Set JSON key – add key/value pair or change the value of an existing key in the dictionary/JSON string.
- Delete JSON key – remove key/value pairs from a dictionary/JSON string.

Building on top of these actions, if the functionality required in a workflow is not supported by one of the default apps or app actions, a custom Python script can be injected into the workflow using the “Shuffle Tools” app. This script can perform more advanced data parsing, looping, error handling, etc. For example, this is useful if a custom API response is difficult to parse with existing app actions. (Figure 4)



```python
import json
jsondata = json.loads(r"""$virustotal""")

filtered_data = []
for i in jsondata:
  try:
      temp = {}
      temp["ip"] = i["body"]["data"]["id"]
      temp["last_analysis_stats"] = i["body"]["data"]["attribut

      temp["malicious"] = False
      malicious = int(temp["last_analysis_stats"]["malicious"])
      suspicious = int(temp["last_analysis_stats"]["suspicious"
      if(malicious>2 or suspicious>5):
        temp["malicious"] = True
      filtered_data.append(temp)
  except:
    pass

print(filtered_data)
```

```
{"category":"harmless","result":"clean","method":"blacklist","engine_name":"Spam
{"category":"undetected","result":"unrated","method":"blacklist","engine_name":"
{"category":"malicious","result":"malicious","method":"blacklist","engine_name":
{"category":"malicious","result":"malware","method":"blacklist","engine_name":"F
{"category":"harmless","result":"clean","method":"blacklist","engine_name":"alph
{"category":"harmless","result":"clean","method":"blacklist","engine_name":"Lion
{"category":"harmless","result":"clean","method":"blacklist","engine_name":"Cybl
{"category":"harmless","result":"clean","method":"blacklist","engine_name":"Secl
Verdict Cloud":
{"category":"undetected","result":"unrated","method":"blacklist","engine_name":"
Verdict Cloud"},"Google Safebrowsing":
{"category":"harmless","result":"clean","method":"blacklist","engine_name":"Goog
Safebrowsing"},"SafeToOpen":
{"category":"undetected","result":"unrated","method":"blacklist","engine_name":"
{"category":"harmless","result":"clean","method":"blacklist","engine_name":"ADMI
{"category":"harmless","result":"clean","method":"blacklist","engine_name":"ESTs
Networks":
{"category":"harmless","result":"clean","method":"blacklist","engine_name":"Juni
Networks"},"Heimdal Security":
{"category":"harmless","result":"clean","method":"blacklist","engine_name":"Heim
Security"},"AutoShun":
{"category":"undetected","result":"unrated","method":"blacklist","engine_name":"
{"category":"harmless","result":"clean","method":"blacklist","engine_name":"Trus
{"category":"malicious","result":"malicious","method":"blacklist","engine_name":
{"category":"harmless","result":"clean","method":"blacklist","engine_name":"Dr.W
{"category":"harmless","result":"clean","method":"blacklist","engine_name":"Emsi
{"category":"harmless","result":"clean","method":"blacklist","engine_name":"Abus
{"category":"harmless","result":"clean","method":"blacklist","engine_name":"Webr
{"category":"harmless","result":"clean","method":"blacklist","engine_name":"Avir
{"category":"harmless","result":"clean","method":"blacklist","engine_name":"secu
AVL":
{"category":"harmless","result":"clean","method":"blacklist","engine_name":"Anti
```

```
"Test result" : { 2 items
  "success" : true
  "message" : [ 1 item
    0 : { 3 items
      "ip" : "12.232.158.130"
      "last_analysis_stats" : { 5 items
        "harmless" : 59
```

Figure 4: Shuffle – demonstration of the scripting interface

Finally, IF conditions can be used to add logic to the workflow and run or skip specific nodes. An IF condition is added to a connection between workflow components and can be used to check a value of the available variables and compare it to an expected result. Multiple conditions can be added to a single connection; however, the connection will execute the next node if even one of the conditions is true. The following comparison options, along with their inverses, are available in Shuffle IF condition:

- Equals/does not equal
- Starts with/ends with
- Contains/contains any of
- Matches Regex (Regular Expression)
- Larger than/less than
- Is empty

Figure 5 demonstrates a full workflow that starts with input data containing a suspicious IP address. IOCs are parsed out as a list from the input data using “Parse IOC” action. Following this, all the IPs are filtered out of the newly made list. The IPs are separately sent to VirusTotal for enrichment. The results from VirusTotal are handled by a custom script. Finally, if one of the IPs is determined to be malicious, an alert is created in the ticketing system.

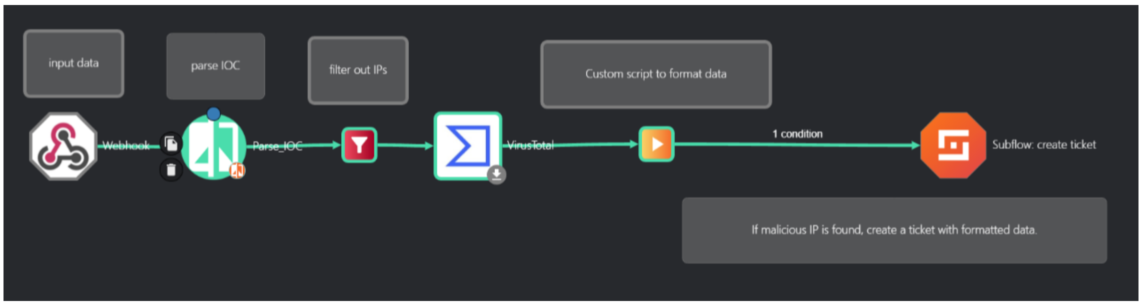
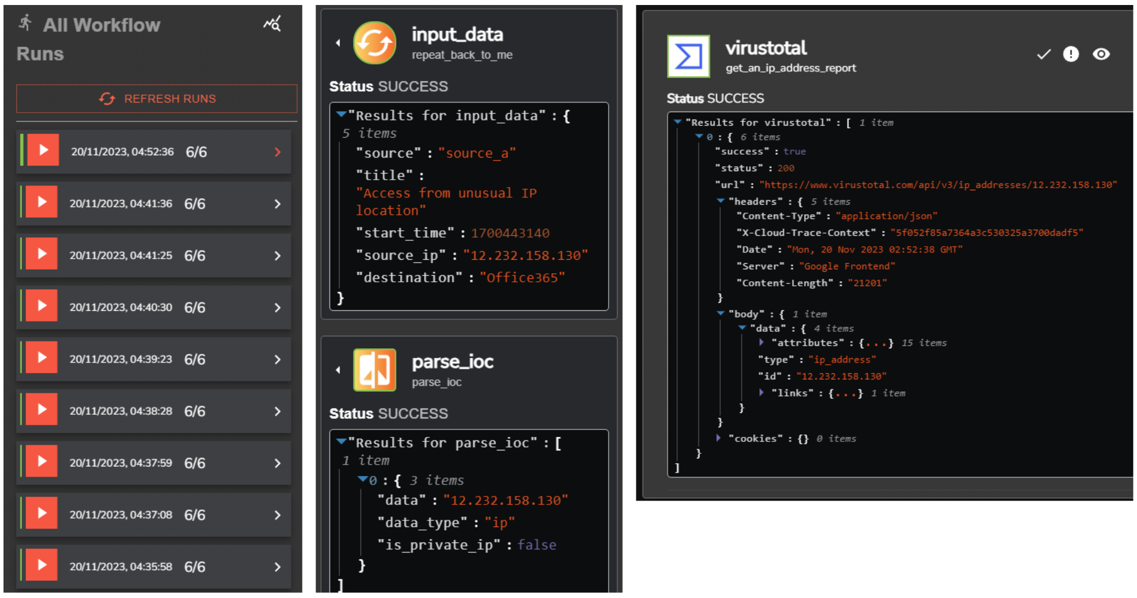


Figure 5: Shuffle – demonstration of a workflow

4.1.3 UI

Although the tool does not provide detailed dashboards for statistics like some of the higher-end SOARs, its user interface is very powerful.

When working with workflows, the UI allows users to place and connect apps intuitively. Workflow UI allows users to explore previous workflow executions (Figure 6 a) and to see changes every step of the way chronologically from top to bottom (Figure 6 b). This can be useful for debugging an execution and seeing the input and output of each node/app once expanded (Figure 6 c). Users are also able to control the variables that are passed into the apps. The variable path can be easily copied just by clicking the JSON representation of the output/variable data.


All Workflow Runs

REFRESH RUNS

```
20/11/2023, 04:52:36  6/6
20/11/2023, 04:41:36  6/6
20/11/2023, 04:41:25  6/6
20/11/2023, 04:40:30  6/6
20/11/2023, 04:39:23  6/6
20/11/2023, 04:38:28  6/6
20/11/2023, 04:37:59  6/6
20/11/2023, 04:37:08  6/6
20/11/2023, 04:35:58  6/6
```

input_data
repeat_back_to_me
Status SUCCESS

```
"Results for input_data" : {
5 items
  "source" : "source_a"
  "title" :
  "Access from unusual IP
  location"
  "start_time" : 1700443140
  "source_ip" : "12.232.158.130"
  "destination" : "office365"
}
```

parse_ioc
parse_ioc
Status SUCCESS

```
"Results for parse_ioc" : [
1 item
  0 : { 3 items
    "data" : "12.232.158.130"
    "data_type" : "ip"
    "is_private_ip" : false
  }
]
```

virustotal
get_an_ip_address_report
Status SUCCESS

```
"Results for virustotal" : [ 1 item
  0 : { 6 items
    "success" : true
    "status" : 200
    "url" : "https://www.virustotal.com/api/v3/ip_addresses/12.232.158.130"
    "headers" : { 5 items
      "Content-Type" : "application/json"
      "X-Cloud-Trace-Context" : "5f052f85a7364a3c530325a3700dadf5"
      "Date" : "Mon, 20 Nov 2023 02:52:38 GMT"
      "Server" : "Google Frontend"
      "Content-Length" : "21201"
    }
    "body" : { 1 item
      "data" : { 4 items
        "attributes" : {...} 15 items
        "type" : "ip_address"
        "id" : "12.232.158.130"
        "links" : {...} 1 item
      }
    }
    "cookies" : {} 0 items
  }
]
```

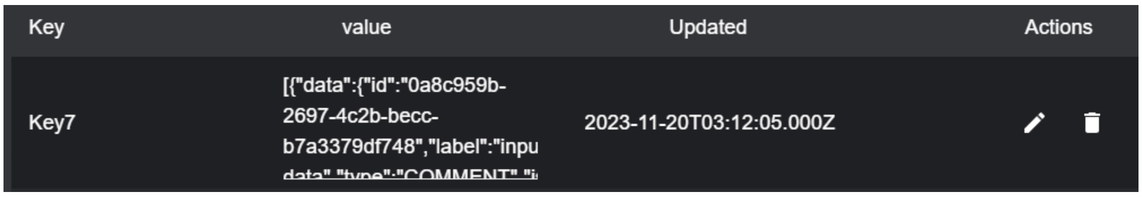
(a) (b) (c) .

Figure 6: Shuffle – workflow runs (a), run details (b), app result details (c)

One negative regarding the workflow UI is that connections between apps cannot be removed. To remove the connection one of the apps must be deleted.

The UI allows for full creation and configuration of the apps. This means API endpoints can be added and specific request types, with required variables, can be set up. Each API endpoint will become a separate app action. Moreover, most of the actions required in a workflow in Shuffle can be set up using the tool's UI. There is no necessity to write code or run commands to achieve most of the functionality. This is a big plus because it makes the tool more intuitive and accessible to people who do not have software development skills.

Another negative aspect of the UI is in the admin panel, specifically in the Files and Datastore sections. These sections can get overwhelming if there are many entries. Moreover, there is no search functionality to look through the storage via the UI. Another small issue is the formatting of the entries, as the ones with large data are not presented in a pleasing way and the time and date are also not formatted correctly (Figure 7).



Key	value	Updated	Actions
Key7	[{"data":{"id":"0a8c959b-2697-4c2b-becc-b7a3379df748","label":"inpu data","type":"COMMENT","i	2023-11-20T03:12:05.000Z	✎ 🗑

Figure 7: Shuffle – formatting of a JSON string in Datastore

Finally, as seen in figure 4, the UI for entering Python scripts inside of workflows has limited syntax highlighting. It also does not highlight errors in the syntax. The lack of syntax highlighting makes the development of larger Python scripts less convenient and increases the danger of introducing hard-to-spot errors in the code. However, since workflows are intended to automate critical security procedures, the bugs in relevant software might introduce additional risks and vulnerabilities into protected systems.

4.1.4 Orchestration

On the orchestration side, it was demonstrated that multiple different tools and data sources can be connected together seamlessly in a workflow using apps and triggers. Apart from triggers, workflows can be set on a schedule as well. Moreover, workflows can be connected with each other seamlessly as well using subflows. This makes data flow between different security tools easy and removes one of the typical barriers of automation. Workflow execution status and detailed information is visible via the UI, and light debugging is also possible through the UI. Custom apps can be used to manage other security tools. Persistent workflow variables can be used to set specific configurations for workflows.

4.1.5 Version Control

Unfortunately, options for version control in Shuffle are very limited. Shuffle does have a Version History feature. This mainly affects workflows and scripts inside of workflows. However, this is a beta feature and still has a way to go (Figure 8).

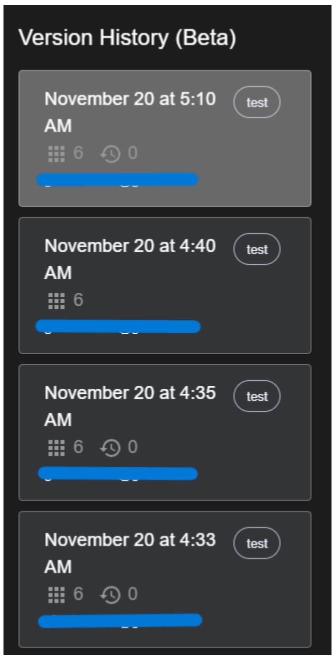


Figure 8: Shuffle – a screenshot of the Version History tab

The older versions of the workflow can be clicked to switch to the previous version. This can be used for workflow version rollback in cases of emergency. However, the interface is minimal, and use of this feature has resulted in losing progress in some cases during testing. On top of that, there is no good way to track exactly what changes have been made.

Another option for version control is to export workflows as JSON strings and upload them to a separate version control platform. Exporting workflows is a feature that Shuffle offers. However, this is not a convenient option and still does not provide an easily understandable solution for monitoring changes or for performing a rollback reliably.

4.2 N8n

N8n is node-based workflow automation tool that allows users to connect different APIs together and manipulate input/output data of different nodes primarily through using a visual interface. While Shuffle was specifically targeted towards security operations, N8n offers workflow automation for more general use-cases like automation for finding leads and CRM (Customer Relationship Management) integrations [35] . On top of this, N8n can also be used as a SOAR platform to automate more security-oriented workflows. The tool is fair-code licensed, meaning it is not fully open-source [39] . Although its source-code is openly available, it is generally free to use, and it can have public and private contributors, it is commercially restricted by its authors.

Here are some basic features offered by N8n:

- Nodes – are the key components of workflows. Nodes have two types of operations: they can act as triggers to start the workflow, and can perform actions to fetch, send and manipulate data. Many pre-existing nodes are available that are built-in or are shared by the N8n community. Also, N8n allows users to build their own nodes. Each node has the following setting toggles: Execute Once, Retry On Fail, Continue On Fail, Always Output Data – output an empty item even if there is no data.
- Workflows – are a collection of nodes that are connected and setup in specific ways to automate a process. Mainly, they can be comprised of the following components:
    - Nodes and Connections – Connections establish links between nodes. A connection between two nodes gives the first node's output to the second node as the input. The nodes themselves can be used to make API and app requests or run local actions to manipulate the data.
    - The Code node – allows users to write custom JavaScript or Python scripts and run them inside workflows. The scripts can access all the variables that the nodes can.
    - Variables – Users can create variables for specific workflows or can set general variables shared between workflows.
    - Triggers – will start executing workflows. There are various trigger options that can be set up according to specific use-cases. A Notable trigger that is not mentioned in detail in section 4.2.1 is the "On a schedule" trigger which can be set to go off in a specified time interval or according to a cron string. On top of this, multiple interval rules can be added to the same trigger.

    - Flow Logic – N8n allows users to implement advanced logic elements in workflows. These include:
        - Nodes for data manipulation and fetching data
        - Nodes for IF/Switch conditionals
        - Waiting nodes – allows pausing of workflow execution for a certain duration, until a specified time, or until a Webhook gets called.
        - Error handling nodes – allow for a certain action to be taken if an error is thrown.
- Credential Storage – Authentication credentials can be stored for apps/nodes. Saved credentials can also be shared with users.

4.2.1 Data Ingestion

Similarly to Shuffle, there are various methods for data ingestion into N8n. Firstly, as mentioned, data can be ingested into workflows via nodes that can be setup to act as triggers. Some of the main trigger options that ingest data into a workflow include the following:

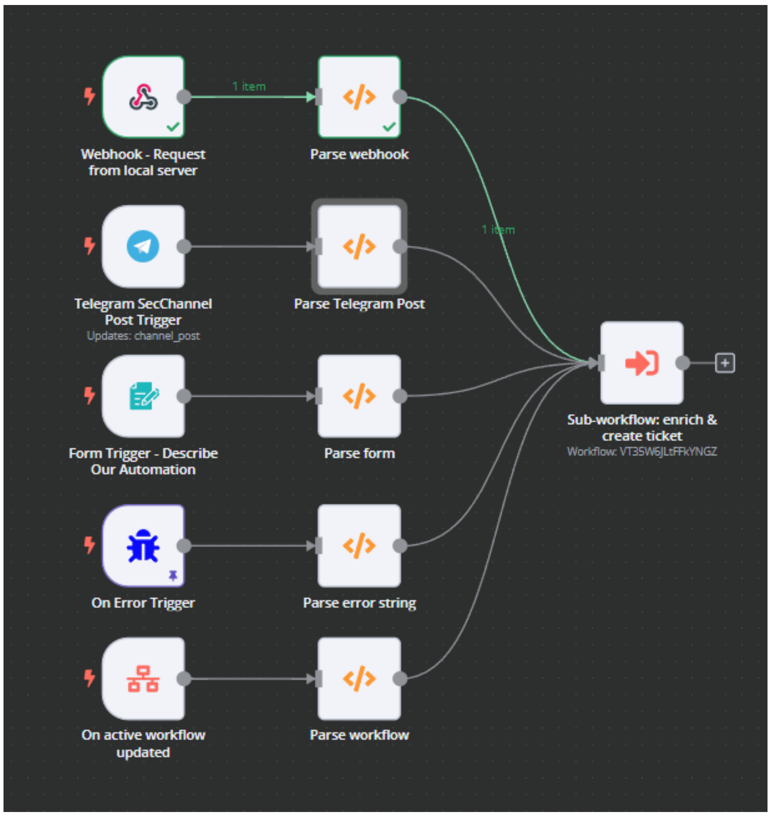
- On an app event – will trigger when a specified action occurs in a connected app. For example, when a message is received in a Telegram channel. Related data from the app will be passed into the workflow.
- On a Webhook call – Similarly to Shuffle, HTTP methods can be setup to work with N8n Webhooks along with header or basic authentication. Differently from Shuffle's Webhook integration, in N8n users can also choose when to respond immediately, when last node finishes, or at a custom time during workflow execution using the "Respond to Webhook" node.
- On form submission – N8n allows users to create their own web forms and trigger workflows once these web forms get submitted along with the form data.

- When called by parent workflow – A workflow can be set up to allow other workflows to execute it. In this case the parent can supply data to the sub-workflow.

- Error Trigger – This trigger executes a workflow when another workflow has an error. The error data is passed in as a variable. This feature is not available on Shuffle side.

On top of this, like in Shuffle, data can be ingested into workflows by using the various nodes to make calls to custom API endpoints or pre-existing app integrations.

To demonstrate the capability of all of the aforementioned triggers in action, in Figure 9, a workflow is visualized that ingests data from: a Webhook trigger waiting for a post request from a local server, a Telegram app that sends an event once a post is added to the specified channel, from a custom form submission, and ingests an error item thrown by a specified workflow. The data is taken from all of these triggers, parsed, and sent to a sub-workflow for enrichment and ticket creation.



Webhook - Request from local server
1 item
Parse webhook
1 item
Telegram SecChannel Post Trigger
Updates: channel_post
Parse Telegram Post
Form Trigger - Describe Our Automation
Parse form
On Error Trigger
Parse error string
On active workflow updated
Parse workflow
Sub-workflow: enrich & create ticket
Workflow: VT35W6JLtFFkYNGZ

Figure 9: N8n – data ingestion demonstration

4.2.2 Workflow Automation

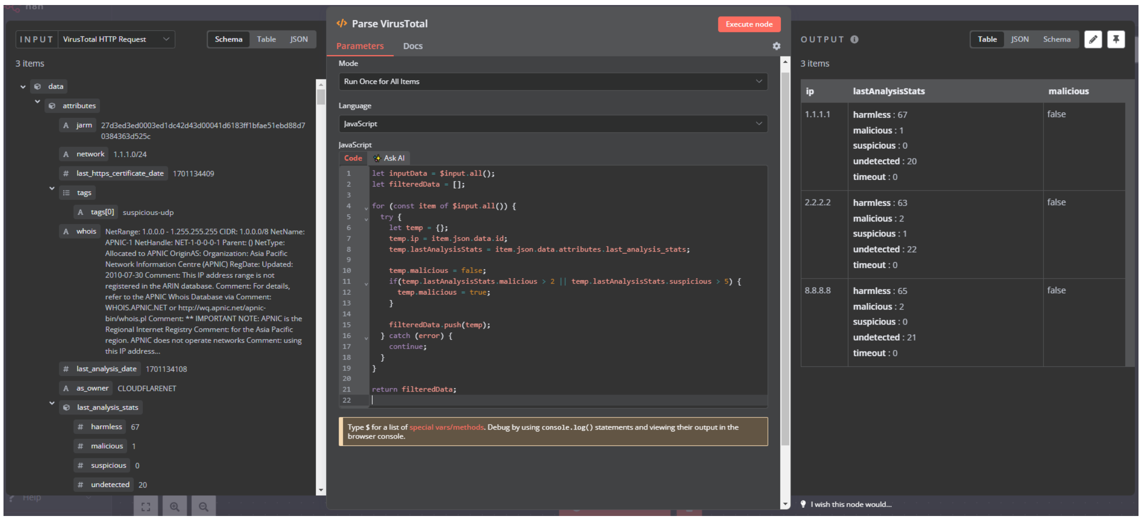
Once data is injected into a workflow, it can be manipulated in multiple ways. Firstly, data will be passed between nodes. However, this is handled differently in N8n compared to in Shuffle. In N8n, by default, every output is an item or a list of items, just a string or a number cannot be outputted. If a node outputs a list of items, the next node will receive these items separately, which is similar to using *$var.#* functionality in Shuffle. Flexibility regarding how these items will be processed consists of configuring the node to only process the first item with the "Execute Once" setting. This "Execute Once" setting can be used in combination with inline JavaScript and *JSON.stringify()* function to pass input to the next node as a string. Although this string must still be part of an object, like *$output.result*, it achieves the default functionality of passing output as strings in Shuffle. On top of this, the documentation of N8n highlights another method of performing loops because some nodes, like HTTP request, do not automatically iterate over items. To create such a loop in N8n one must take the output of a node and give it as input to one of the previous nodes. There should be an IF node in between also, which will decide if the loop is complete or not. A use-case for such a loop is to handle pagination during page load requests. Another example would be to use the loop as a retry limit during an API request. Example: if the expected result is not received, retry the request maximum 5 times. A variable *$runIndex* can also be used to track the number of times a node has been run and can be used as an index for the loop.

Regarding data manipulation, one way to manipulate list data is through the "Data transformation" nodes. Some of the main ones when it comes to working with lists outside of the code node include:

- Item Lists node – a helper node for working with lists (arrays) allowing for list concatenation, duplicate removal, sorting, limiting, etc.
- Rename Keys and Edit Fields nodes – allowing for modifying object keys and values.
- Filter node – allows users to perform an IF operation on an object's key value and filter out the matching entries.

- Merge node – allows for appending one input to another, combining two items together, etc.
- Compare Datasets node – allows for comparing two items according to a matching field. It will return what value for which keys overlaps between these items and what value is different.

On top of pre-defined data manipulation options, there is the Code node, which allows scripts to be run in JavaScript or Python. This is one of the most important nodes since it gives the most freedom to the users to deal with workflow data and handle errors. The script run by this node can perform advanced parsing, looping, error handling, and so on. For example, this is useful if a custom API response is difficult to parse with existing node functionality (Figure 10).



3 items

```
data
  attributes
    A jarm 27d3ed3ed0003ed1dc42d43d00041d6183ff1bfae51ebd88d70384363d525c
    A network 1.1.1.0/24
    # last_https_certificate_date 1701134409
    tags
      A tags[0] suspicious-udp
    A whois NetRange: 1.0.0.0 - 1.255.255.255 CIDR: 1.0.0.0/8 NetName: APNIC-1 NetHandle: NET-1-0-0-0-1 Parent: () NetType: Allocated to APNIC OriginAS: Organization: Asia Pacific Network Information Centre (APNIC) RegDate: Updated: 2010-07-30 Comment: This IP address range is not registered in the ARIN database. Comment: For details, refer to the APNIC Whois Database via Comment: WHOIS.APNIC.NET or http://wq.apnic.net/apnic-bin/whois.pl Comment: ** IMPORTANT NOTE: APNIC is the Regional Internet Registry Comment: for the Asia Pacific region. APNIC does not operate networks Comment: using this IP address...
    # last_analysis_date 1701134108
    A as_owner CLOUDFLARENET
    last_analysis_stats
      # harmless 67
      # malicious 1
      # suspicious 0
      # undetected 20
```

Mode: Run Once for All Items

Language: JavaScript

JavaScript

```javascript
let inputData = $input.all();
let filteredData = [];

for (const item of $input.all()) {
  try {
    let temp = {};
    temp.ip = item.json.data.id;
    temp.lastAnalysisStats = item.json.data.attributes.last_analysis_stats;

    temp.malicious = false;
    if(temp.lastAnalysisStats.malicious > 2 || temp.lastAnalysisStats.suspicious > 5) {
      temp.malicious = true;
    }

    filteredData.push(temp);
  } catch (error) {
    continue;
  }
}

return filteredData;
```

Type $ for a list of special vars/methods. Debug by using console.log() statements and viewing their output in the browser console.

3 items

ip	lastAnalysisStats	malicious
1.1.1.1	harmless : 67 malicious : 1 suspicious : 0 undetected : 20 timeout : 0	false
2.2.2.2	harmless : 63 malicious : 2 suspicious : 1 undetected : 22 timeout : 0	false
8.8.8.8	harmless : 65 malicious : 2 suspicious : 0 undetected : 21 timeout : 0	false

Figure 10: N8n – demonstration of Code node interface

N8n supports IF conditions similar to the ones in Shuffle, but on top of that, it also has a Switch node, which allows the user to account for more outcomes than just true and false with a single node. These options can be used to add logic to the workflow and run or skip specific nodes. Both IF and Switch conditions have access to the connected variables in the workflow. As in Shuffle, the IF node can include multiple sub-conditions. Unlike Shuffle, the node can be setup to return true once ALL of the conditions are met, or if ANY of the conditions are met. The comparison options for these N8n nodes include:

- Equal/Not Equal
- Contains/Not Contains
- Starts With/Not Starts With
- Ends With/Not Ends With
- Regex Match/Regex Not Match
- Is Empty/Is Not Empty

Figure 11 demonstrates a full workflow that starts with input data from a Webhook containing a suspicious IP address. The IPs are parsed out by the Code node. The IPs are separately sent to VirusTotal for enrichment. The results from VirusTotal are handled by a custom script in another Code node. Finally, if one of the IPs is determined to be malicious, a ticket is created by a sub-workflow.

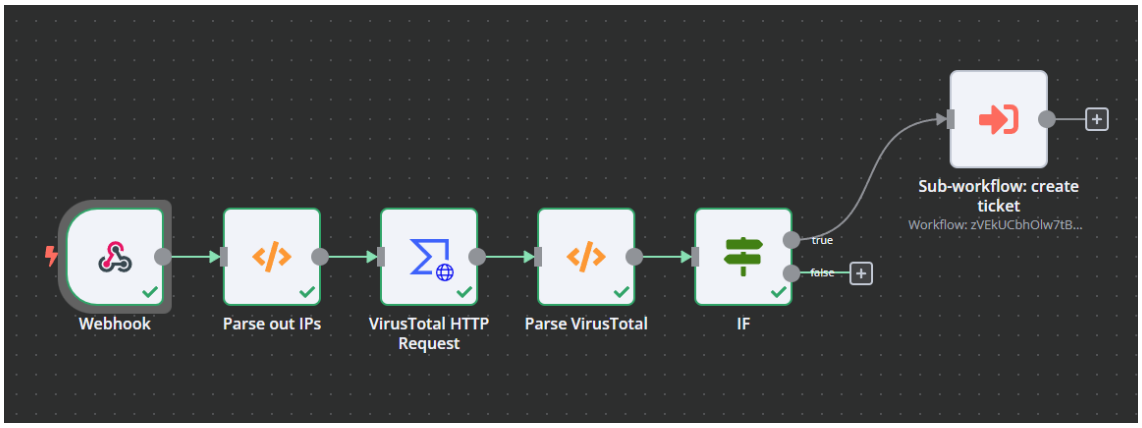
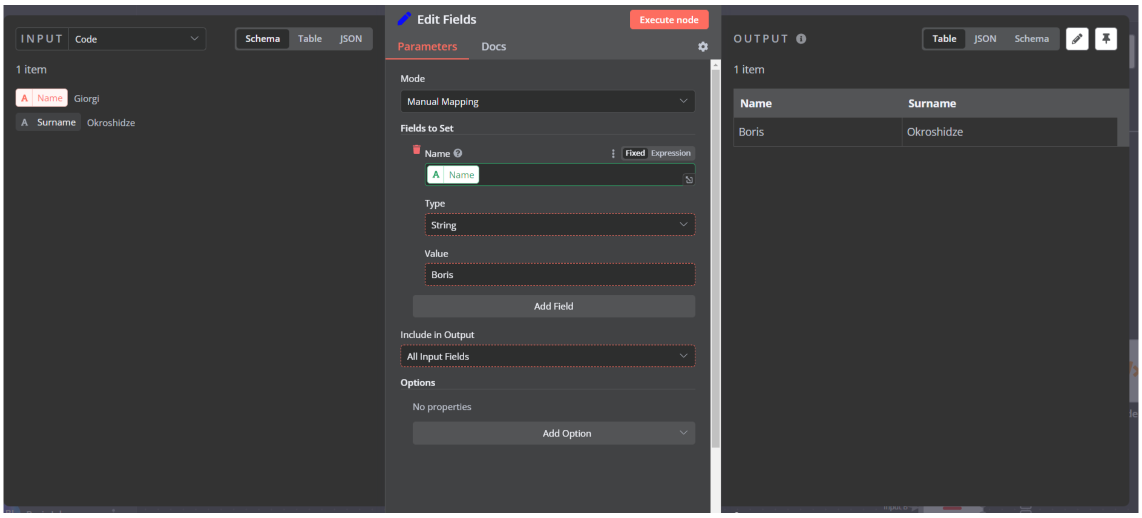


Figure 11: N8n – demonstration of a workflow

4.2.3 UI

Just like the interface of Shuffle, instead of detailed statistics and dashboards N8n's user interface is mainly focused on providing a clean workflow builder experience.

As in Shuffle, the UI allows users to place and connect nodes. Unlike Shuffle, in N8n the connections between nodes can easily be removed without having to remove the nodes themselves. Users are also able to explore previous workflow executions and to see the path of the execution. Users are able to click into each node and see the input and output on the left and right side of the node view (Figure 12). Users are also able to control the variables that are passed into the apps. Adding variables as input is as simple as dragging and dropping the variable from the left side variable panel into the input field of a node.



1 item

Name: Giorgi
Surname: Okroshidze

Mode: Manual Mapping

Fields to Set

Name (Fixed | Expression): Name

Type: String

Value: Boris

Include in Output: All Input Fields

Options: No properties

1 item

Name	Surname
Boris	Okroshidze

Figure 12: N8n – demonstration of node input and output

The highlights of N8n’s workflow UI are IF condition and Switch nodes, which allow users to connect multiple branches to a single condition node (Figure 13).

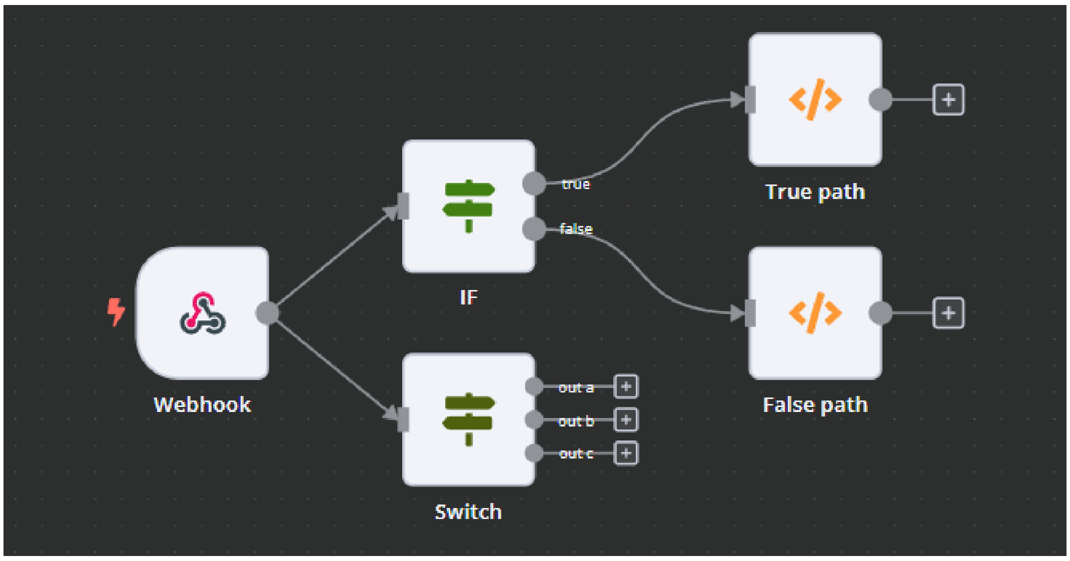


Figure 13: N8n – demonstration of IF condition and Switch nodes

It is possible to create custom nodes in N8n. This is a multi-step process and includes running a local N8n instance, Git, writing the node code, building the node, and deploying the custom node. As this requires interactions outside of the cloud automation tool, it is outside of the scope of this thesis. However, it is important to mention that there is no user interface for creating custom nodes in N8n. This is one of the minuses regarding N8n’s UI since Shuffle offers app creation options through its UI.

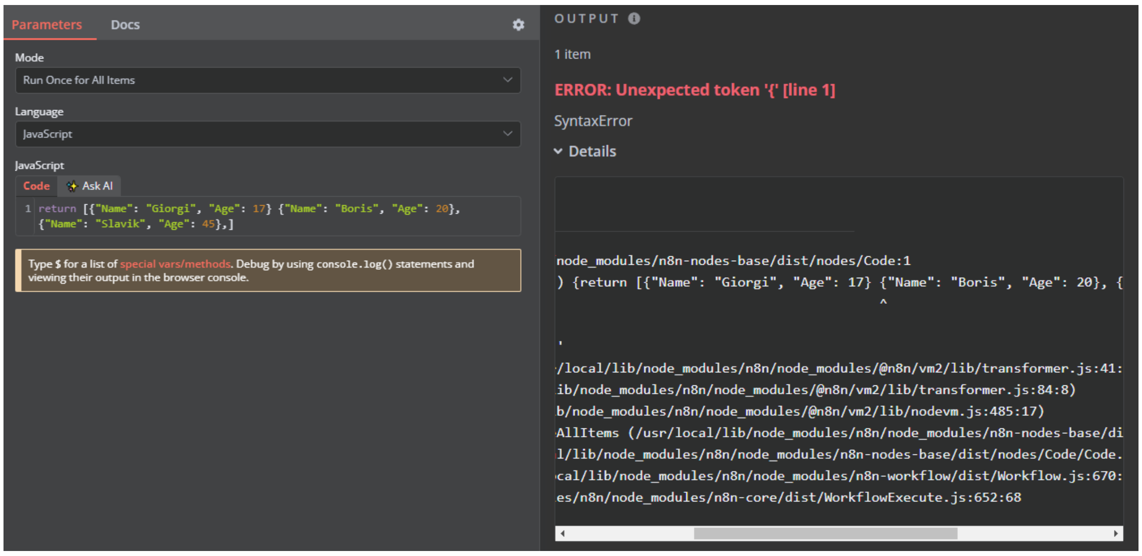
The UI also offers easy ways to store credentials for nodes and to store global variables that are accessible to workflows.

Also, as seen in figure 10, the UI for entering scripts inside the code node of a workflows has marginally better syntax highlighting than the Shuffle code editor. However, like Shuffle, it also does not highlight errors efficiently in the syntax.

4.2.4 Orchestration

According to the previous sections, it was demonstrated N8n is capable of easily connecting multiple different tools and data sources with each other using its workflow automation, triggers and data handling nodes/apps. Workflows and apps can be configured individually allowing for precise control. Detailed information about workflow executions is visible via the UI, along with inputs and outputs of nodes allowing for brief debugging options. This is all very similar to what Shuffle offers. However, N8n offers many more options regarding error handling outside of the Code node.

An error handler workflow can be created with the “Error Trigger” which can be connected to typical workflow. With this kind of configuration, once an error is thrown the error handler workflow will be triggered which will receive the error information. This workflow can be set up to attempt a rerun or to forward this error towards an external source, like a ticketing system, for example. This option increases the ability for users to monitor their integrations, debug errors, and respond to issues on time. On top of this, the “Stop and Error” node can be used to manually throw errors in a workflow. Figure 14 is an example of a Code node throwing an error because of a syntax issue. An error handler workflow with the “Error Trigger” automatically gets initiated with this error input and can handle the it as needed. Figure 15 shows this error input caught by the “Error Trigger.”



Run Once for All Items

JavaScript

```
return [{"Name": "Giorgi", "Age": 17} {"Name": "Boris", "Age": 20},
{"Name": "Slavik", "Age": 45},]
```

Type $ for a list of special vars/methods. Debug by using console.log() statements and viewing their output in the browser console.

1 item

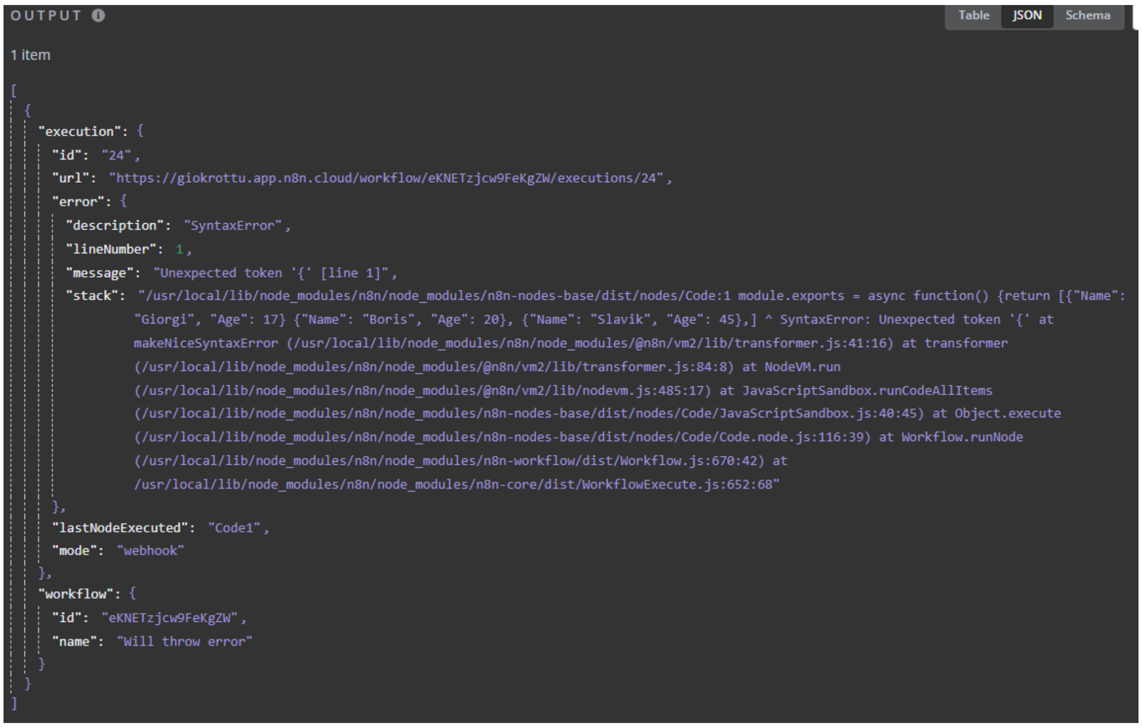
ERROR: Unexpected token '{' [line 1]

SyntaxError

```
node_modules/n8n-nodes-base/dist/nodes/Code:1
) {return [{"Name": "Giorgi", "Age": 17} {"Name": "Boris", "Age": 20}, {
                                         ^

'
/local/lib/node_modules/n8n/node_modules/@n8n/vm2/lib/transformer.js:41:
ib/node_modules/n8n/node_modules/@n8n/vm2/lib/transformer.js:84:8)
b/node_modules/n8n/node_modules/@n8n/vm2/lib/nodevm.js:485:17)
AllItems (/usr/local/lib/node_modules/n8n/node_modules/n8n-nodes-base/di
l/lib/node_modules/n8n/node_modules/n8n-nodes-base/dist/nodes/Code/Code.
cal/lib/node_modules/n8n/node_modules/n8n-workflow/dist/Workflow.js:670:
es/n8n/node_modules/n8n-core/dist/WorkflowExecute.js:652:68
```

Figure 14: N8n – a syntax error thrown by Code node



1 item

```
[
  {
    "execution": {
      "id": "24",
      "url": "https://giokrottu.app.n8n.cloud/workflow/eKNETzjcw9FeKgZW/executions/24",
      "error": {
        "description": "SyntaxError",
        "lineNumber": 1,
        "message": "Unexpected token '{' [line 1]",
        "stack": "/usr/local/lib/node_modules/n8n/node_modules/n8n-nodes-base/dist/nodes/Code:1 module.exports = async function() {return [{"Name":
                 "Giorgi", "Age": 17} {"Name": "Boris", "Age": 20}, {"Name": "Slavik", "Age": 45},] ^ SyntaxError: Unexpected token '{' at
                 makeNiceSyntaxError (/usr/local/lib/node_modules/n8n/node_modules/@n8n/vm2/lib/transformer.js:41:16) at transformer
                 (/usr/local/lib/node_modules/n8n/node_modules/@n8n/vm2/lib/transformer.js:84:8) at NodeVM.run
                 (/usr/local/lib/node_modules/n8n/node_modules/@n8n/vm2/lib/nodevm.js:485:17) at JavaScriptSandbox.runCodeAllItems
                 (/usr/local/lib/node_modules/n8n/node_modules/n8n-nodes-base/dist/nodes/Code/JavaScriptSandbox.js:40:45) at Object.execute
                 (/usr/local/lib/node_modules/n8n/node_modules/n8n-nodes-base/dist/nodes/Code/Code.node.js:116:39) at Workflow.runNode
                 (/usr/local/lib/node_modules/n8n/node_modules/n8n-workflow/dist/Workflow.js:670:42) at
                 /usr/local/lib/node_modules/n8n/node_modules/n8n-core/dist/WorkflowExecute.js:652:68"
      },
      "lastNodeExecuted": "Code1",
      "mode": "webhook"
    },
    "workflow": {
      "id": "eKNETzjcw9FeKgZW",
      "name": "Will throw error"
    }
  }
]
```

Figure 15: N8n – a syntax error caught by a workflow with the "Error Trigger"

In the Enterprise version of N8n, Log Streaming can be setup which allows the tool to forward logs towards a selected destination, which can be a Syslog server, a generic Webhook, and so on. Users can set up which logged events are sent to the destination. Streamable events include:

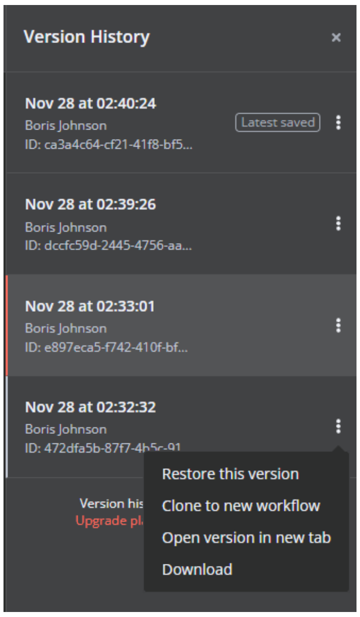
- Workflow – started, success, failed
- Node executions – started, finished

- Audit logs:
  - User creation, update, deletion, etc.
  - Package installed, updated, created
  - Workflow created, deleted, updated

4.2.5 Version Control

The non-enterprise versions of N8n offer the following options regarding version control of implemented workflows:

- Downloading/exporting implemented workflows as JSON files and importing saved workflow JSON files into the tool. Just like in Shuffle, one option for version control would be to keep track of these JSON files in Git. However, as mentioned in section 4.1.5, manually carrying out this process is not a convenient method of performing version control.
- A version history of each workflow is kept in N8n, accessible through the ”Version History” tab (Figure 16). This tab allows users to see when each version was saved, download each version, restore an older version, and so on. This tab also shows who each version was edited by.



Version History

Nov 28 at 02:40:24
Boris Johnson
ID: ca3a4c64-cf21-41f8-bf5...
Latest saved

Nov 28 at 02:39:26
Boris Johnson
ID: dccfc59d-2445-4756-aa...

Nov 28 at 02:33:01
Boris Johnson
ID: e897eca5-f742-410f-bf...

Nov 28 at 02:32:32
Boris Johnson
ID: 472dfa5b-87f7-4b5c-91

Version his
Upgrade pl

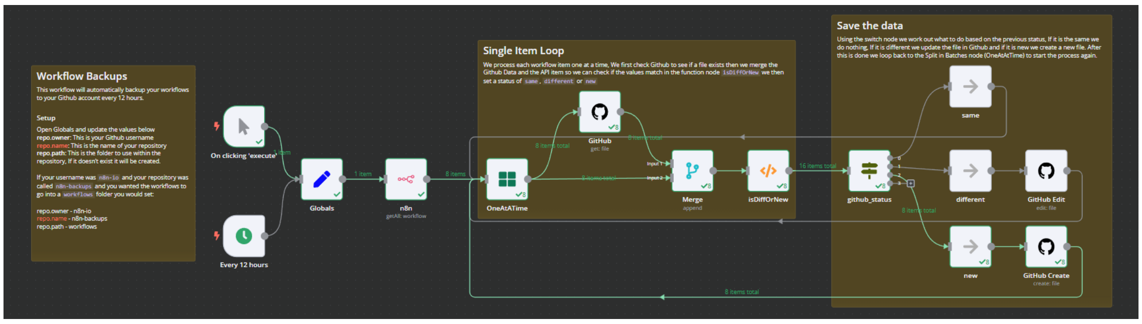
Figure 16: N8n – a screenshot of the Version History tab

However, non-enterprise versions of N8n are highly limited in terms of the amount of days workflow history can be stored. For example, the Pro version of the tool stores workflow history for just 5 days. On top of this, if the workflow has multiple components, keeping track of what was changed with each version update is very difficult through the tool's interface since it simply loads the previous workflow without highlighting the changes.

The enterprise version of N8n offers options for setting up environments and Git connectivity. Users can set up development and production instances of N8n. Regarding Git integration, N8n uses three Git processes: Push, Pull, and Commit. When changes are committed to Git, the following is included:

- Workflows – including tags, email of the workflow owner
- Credential and Variable stubs – ID, name, type

A way to overcome some of the workflow history and version control issues without upgrading to the Enterprise version is to have a separate workflow that will push existing workflows into Git. To achieve this, the "N8n" node can be implemented to get the workflows and "GitHub" node can be used to push to Git. Figure 17 demonstrates a pre-existing workflow that was imported from N8n's website called "Back Up Your N8n Workflows to Github":


Workflow Backups

This workflow will automatically backup your workflows to your Github account every 12 hours.

Setup

Open Globals and update the values below
repo.owner: This is your Github username
repo.name: This is the name of your repository
repo.path: This is the folder to use within the repository, If it doesn't exist it will be created.

If your username was `n8n-io` and your repository was called `n8n-backups` and you wanted the workflows to go into a `workflows` folder you would set:

repo.owner - n8n-io
repo.name - n8n-backups
repo.path - workflows

Single Item Loop

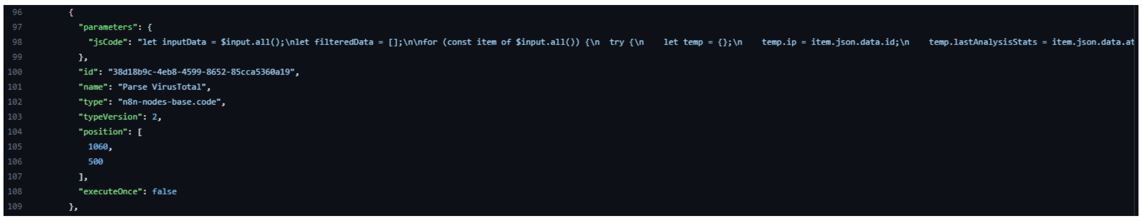
We process each workflow item one at a time, We first check Github to see if a file exists then we merge the Github Data and the API item so we can check if the values match in the function node `isDiffOrNew` we then set a status of `same`, `different` or `new`

Save the data

Using the switch node we work out what to do based on the previous status, If it is the same we do nothing, If it is different we update the file in Github and if it is new we create a new file. After this is done we loop back to the Split in Batches node (OneAtATime) to start the process again.

Figure 17: N8n – demonstration of Git push workflow

The outcome of this workflow results in the specified workflow JSON files being pushed to Git. A negative of this particular approach is that the script code that is written inside the "Code" node is not as clearly readable as pushing a plain script into GitHub. As seen in Figure 18, this will make reviewing the changes more cumbersome.



```
    {
      "parameters": {
        "jsCode": "let inputData = $input.all();\nlet filteredData = [];\n\nfor (const item of $input.all()) {\n  try {\n    let temp = {};\n    temp.ip = item.json.data.id;\n    temp.lastAnalysisStats = item.json.data.at
      },
      "id": "38d18b9c-4eb8-4599-8652-85cca5360a19",
      "name": "Parse VirusTotal",
      "type": "n8n-nodes-base.code",
      "typeVersion": 2,
      "position": [
        1060,
        500
      ],
      "executeOnce": false
    },
```

Figure 18: N8n – demonstration of Code node script after GitHub push

4.3 Summary of Pros and Cons of the Tools

First of all, after analyzing the capabilities offered by these tools it is clear that both of them are exceptionally good at what they set out to do. They offer a wide range of data ingestion options and allow users to manipulate data via UI and scripting to a level that results in workflows that can automate both simple and more advanced steps. This outcome is highly commendable. Having said this, no tool is perfect in its execution and feature-set.

Starting with data ingestion, both tools are flexible when it comes to ingesting and fetching data. N8n has more trigger options that are missing in Shuffle, which include “on app event”, “error trigger”, a more adjustable “schedule trigger” regarding intervals, etc. Moreover, N8n can choose when to respond to the “Webhook” trigger during workflow execution. Shuffle does not offer such flexibility. On the other hand, while both N8n and Shuffle support custom apps and nodes, no custom node creation UI is available in N8n, while Shuffle allows users to create custom apps and actions relatively simply using its UI. This allows users to call specific API endpoints with preset authentication and variable fields in a convenient and re-useable way. Shuffle also has more pre-existing security-related apps, like VirusTotal, AlienVault, AbuseIPDB, that offer more default actions compared to the ones in N8n. This is a substantial advantage for Shuffle.

Moving on to workflow automation, both tools give users flexibility regarding inputting and outputting data between nodes. N8n has a small edge regarding the connections between apps, since its connections can be removed without having to remove the nodes themselves. On the topic of manipulating data inside workflows, like parsing and list filtering outside of scripts, both tools offer a wide variety of options. However, in Shuffle, the data handling actions are all bundled together in the “Shuffle Tools” app, as opposed to being scattered in different nodes N8n. Also, Shuffle already has built-in actions in “Shuffle Tools” for security related filters, like IOC filtering. From the author’s experience, Shuffle’s UI was more intuitive in this regard.

Another advantage for Shuffle is regarding its file and data storage options, allowing users to easily save data in the Shuffle tool itself. N8n does not offer a similar feature outside of its Enterprise version.

N8n has an advantage when it comes to inserting custom scripts since it supports both JavaScript and Python, while Shuffle only supports Python. The scripting UI is very basic in both tools. However, syntax highlighting is better in N8n compared to Shuffle. Also, N8n can recognize code components, like for loops and try-catch statements, and suggest auto-fill options. N8n allows users to resize the code input window, while Shuffle is very limited in this case. Error highlighting is also better in N8n’s interface. However, error highlighting is still limited in both cases, missing to point out basic syntax errors in Shuffle and arbitrarily missing to highlight syntax errors before execution in N8n also. Evidently these tools are made to be mainly used with their UI for creating automations. Even so, it is important to mention that, in terms of writing code, the experience cannot compare to code editor tools.

Regarding the workflow IF conditions, they are handled by a separate node in N8n which means only one node is necessary in the UI to account for both true and false outcomes. The matching options for the rules applied to this node can be set up to match either all or any of the logic statements. On top of this, the Switch node is not limited to true and false outcomes, giving users even more outcome options from a single node. In Shuffle, the IF conditions must be added as part of connections between apps. Because of this, the condition must be entered twice for two app connections to account for both true and false outcomes. Moreover, if multiple conditions apply to one connection, by default Shuffle executes when any of the conditions match. There is no option to change this default behavior.

Looping in workflows is handled differently by default in these tools, but as mentioned, the same results are achievable for both tools if the workflow and nodes are configured properly. Having said this, N8n has more options when it comes to creating a loop between multiple nodes.

Finally, regarding error handling and version control, both tools are limited. (Because of a higher price, this section does not consider the N8n enterprise features like different environments, Git integration, and log streaming.) Both tools allow users to view workflow execution status and failure points in their UIs and can implement error handling in scripts. N8n has the edge in terms of error handling because of the "Error Trigger" and "Stop and Error" nodes that can throw and catch errors. It also has the edge because of the node configuration options like "Retry On Fail", "on error stop workflow" and "continue using error output." Unfortunately, there are no log streaming options for entry-level versions of these tools to get a more detailed view of the error logs. Both tools also offer very limited workflow version switching through their UIs. N8n also has an edge in terms of version control using the "N8n" and "GitHub" nodes to push workflows. However, this is not an ideal solution to version control since the workflow scripts are not easily readable in the pushed JSON files, which makes reviewing changes for more advanced workflows more difficult.

To summarize, both tools share key drawbacks regarding error handling, log sharing, version control options and custom scripting environments. These shortcomings are addressed by the in-house solution that is proposed in chapter 5.

5 In-house SOC Automation in Company P and Real-World Impact on Performance

5.1 Why Use an In-house Solution

As discussed in the previous chapter, the existing automation tools have several shared drawbacks. Therefore, this chapter describes an alternative path for implementing an SOC automation system which involves the development of an in-house solution. For the clarity of the discussion, the in-house SOC automation system of company P is described. The author of this thesis took part in implementation and testing of various automation scripts contributing to the performance impact described in chapter 5.3.

The in-house automation platform satisfies the selection and comparison criteria described in section 3.1. This is demonstrated in section 5.2. Moreover, it has multiple benefits over the tools mentioned in the previous section. The benefits include:

- Freedom of choice regarding the scripting environment and programming language configuration. Also, plain scripts are easier to integrate with existing test environments.
- An in-house solution eliminates error handling and syntax highlighting issues, as was the case in Shuffle. Error handling, in general, is more straightforward to implement in the chosen solution compared to the larger number of steps needed to achieve the same in N8n.
- Version control for workflows is not an issue since the in-house solution can work freely with GitHub. Moreover, any change before deployment can be reviewed easily in GitHub because of the clearly understandable format of the scripts, as opposed to the JSON string format of exported N8n workflows.
- The in-house solution has the ability to utilize more advanced scripts and can achieve a high level of automation without running into the same issues of

complexity and version maintenance problems as the aforementioned tools. Also, there is much less complexity regarding adding custom re-useable objects compared to the complexity of adding custom nodes in N8n.

- An in-house solution is also completely free, assuming the organization considering this approach is already employing software developers or security engineers who have development skills.

That being said, an in-house approach also has its drawbacks, which include:

- Naturally, this tool has to be hosted and maintained by the organization.
- The tool is more limited in terms of existing workflows and plugins, which the aforementioned options offer to help companies get started.

The unescapable reality of implementing security operations automation is that custom automation elements, like scripts, custom apps, and so on, will always be necessary. Moreover, the count and complexity of these custom elements will likely increase over time. If a tool does not offer a convenient way to implement and manage these custom attributes, it will be less desirable to some organizations. As demonstrated, the lack of features and simplicity in terms of error handling, version control, and scripting in Shuffle and N8n was considered a drawback in this regard.

5.2 Implemented Automation Structure

The in-house automation solution incorporates the key features of Shuffle and N8n described in chapter 4. The scripts utilized by the in-house solution mainly fall into the following categories:

- Controllers – are scripts which provide endpoints that can be called to take input data and trigger predefined playbooks. Expected data is defined in controller scripts and parsed before handed over to the playbook scripts. Controllers can be set off by third-party tools or by manual triggers. This functionality is similar to workflow triggers used in Shuffle and N8n.

- Playbooks – are scripts consisting of multiple steps that carry out specific actions and deal with predefined use-cases. Playbooks can be setup to ingress and enrich alerts from a remote source, and to forward alerts or manual actions towards other third-party tools. Playbooks use plugin scripts for communicating with third-party tools. This functionality is similar to the workflows that users can create while using tools like Shuffle and N8n.

- Plugins – consist of scripts specifically for communicating with and fetching data from remote tools. This can include tools for data enrichment, like VirusTotal, AbuseIPDB, MxToolbox, URLSCAN.io or other tools for communication and ticketing. Plugins are very similar to apps in Shuffle and N8n that also allow communication with third-party tools.

- Services – are scripts that can be enabled to run on a schedule, similar to using a cronjob trigger for a workflow in Shuffle or N8n. Certain playbook or plugin scripts can be utilized to run as a service on a schedule.

5.2.1 Data Ingestion

In the in-house solution controller and plugin scripts are used for data ingestion. To achieve that goal, controller scripts take data from remote sources and execute a playbook. Plugin scripts integrate third-party sources into a playbook and fetch data from them. As mentioned, this functionality is the same as the functionality of triggers and apps in Shuffle and N8n.

5.2.2 Workflow Automation

Workflow automation in the in-house solution is handled by playbooks and services. Playbooks take input from controllers, import plugins that interact with third-party security tools and pass the input data into the remote tools. Services are similar to playbooks, however, they are run on a schedule. Playbooks and services offer similar functionality as workflows in Shuffle and N8n.

5.2.3 UI

The most limited part of the in-house automation solution is its UI. However, the UI offers important features, including:

- Allowing users to view statistics from workflow execution steps and alert components.
- Allowing users to perform modifications to basic workflow configuration variables.
- Allowing users to trigger workflow executions and view execution status.

Because of the nature of the tool, the logic of the playbook and plugin scripts cannot be modified via the UI as in Shuffle and N8n.

5.2.4 Orchestration and Version Control

Regarding orchestration, it is clear that the in-house solution is able to easily combine data from multiple sources. Moreover, the performance of the in-house tool can easily be monitored via event monitoring and log aggregation systems since exporting logs from the tool is not limited, unlike in the case of Shuffle and N8n. This allows for more straightforward error handling and performance monitoring.

Regarding version control, naturally, there is no limitation affecting integration with Git or with local testing environments.

5.3 Perceived Impact on SOC Performance

Company P has had various security automation procedures already implemented for the last 2 years. The statistics shown below focus on specific automation procedures implemented since the beginning of October of 2023 via the in-house solution. In this period, no other modifications were made regarding alert sources, alerting rules, or tuning processes. This has had a positive impact on the following aspects:

- The total number of generated cases
- The average time needed for closing cases
- The average duration of a vulnerability management process

In order to demonstrate the benefits of automation, the aforementioned metrics were measured in the year 2023 when recent automations were not yet implemented:

- From July through September regarding case generation number metric
- From January through September regarding average case close time metric

Afterwards, the same metrics were measured from October through December of 2023 after the deployment of recent automations.

5.3.1 The total number of generated cases

The following graphs show the number of cases created between July and September of 2023 vs the cases created between October and December of the same year. The data points in the following graphs are visualized per ISO week.

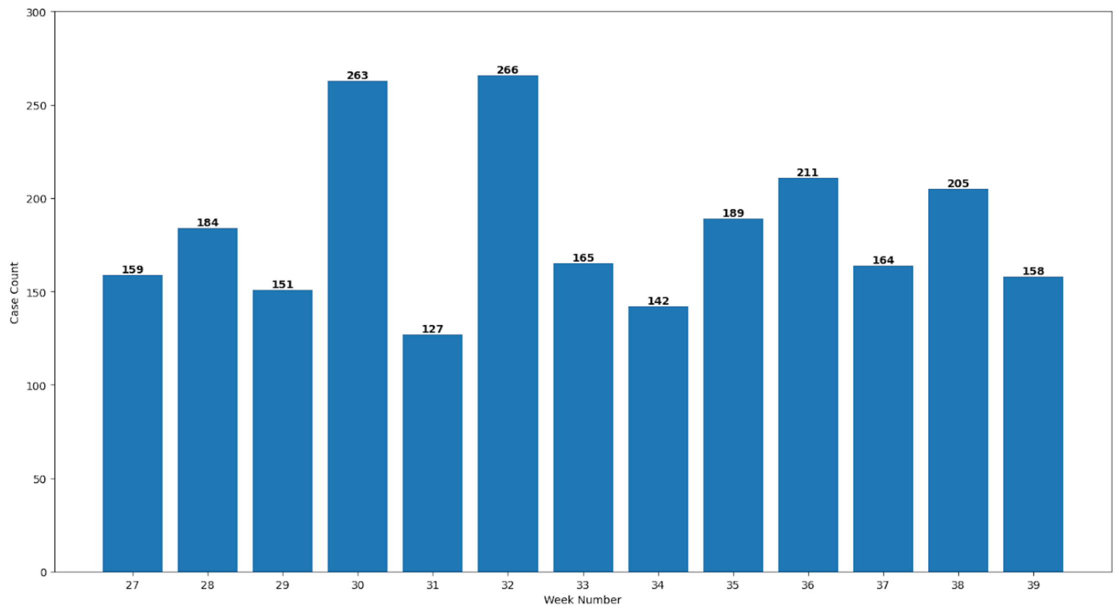


Figure 19: Total cases per week for July-September (weeks 27-39)

In 2023, between the months of July and September, 27th and 39th weeks, the total amount of created cases was 2384 (Figure 19). Over the same length period between October and December, 40th and 52nd weeks, the number of generated cases dropped to 1317 (Figure 20). This means that there were approximately 44.76% less cases created after implementing automation for performing external checks prior to creating cases.

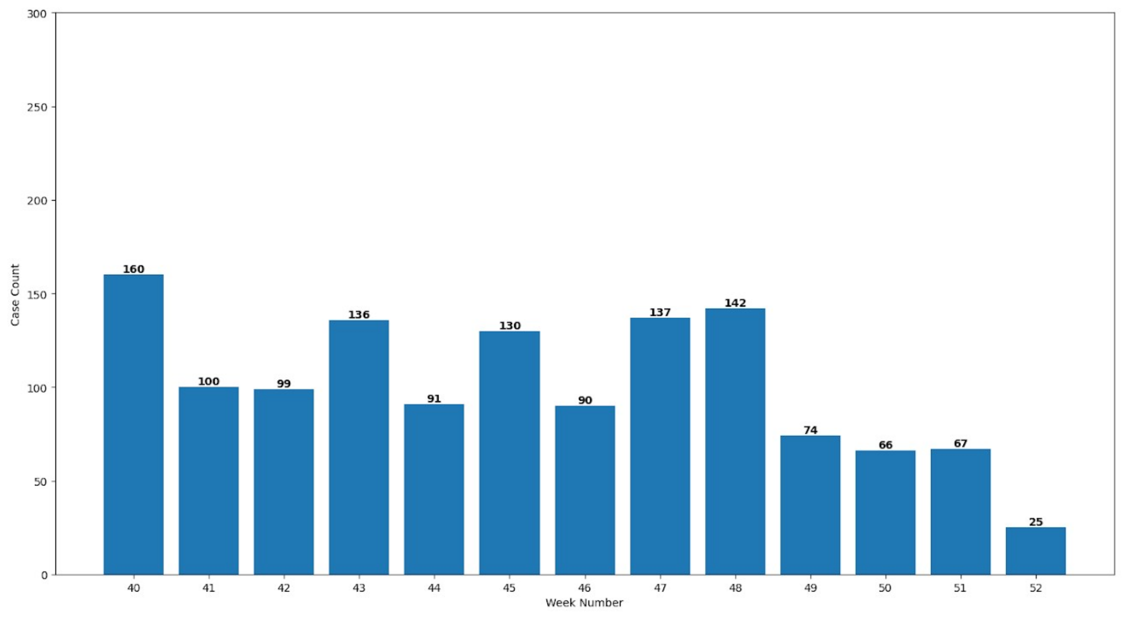


Figure 20: Total cases per week for October-December (weeks 40-52)

5.3.2 The average time needed for closing cases

The following graphs demonstrate the average amount of time required, in hours, for closing cases. The data points in the following graphs are also visualized per ISO week. The average hours printed in the following graphs were rounded to full numbers for visual clarity.

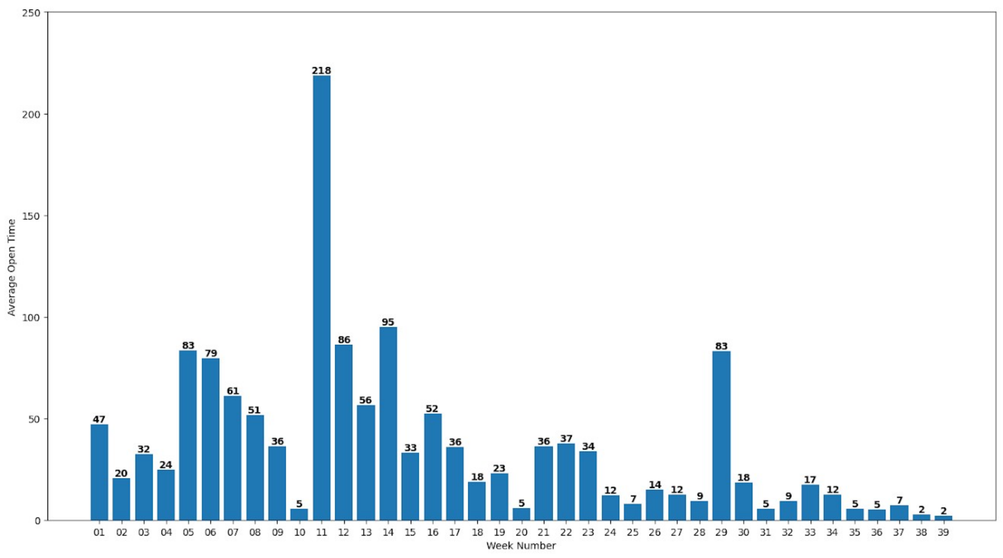


Figure 21: Average hours until case resolution weeks 01-39 (January-September)

Between January of 2023 until September of the same year, the average number of hours until case resolution was approximately 35.93 per week (Figure 21). Specific automated processes were tested during September and were fully put in place since the beginning of October. This is reflected in the lower average hours during the month of September. Between October and December of 2023, the average number of hours until case resolution dropped to approximately 7.78 per week (Figure 22). This is a decrease of approximately 78.35% after implementing automation procedures like alert enrichment.

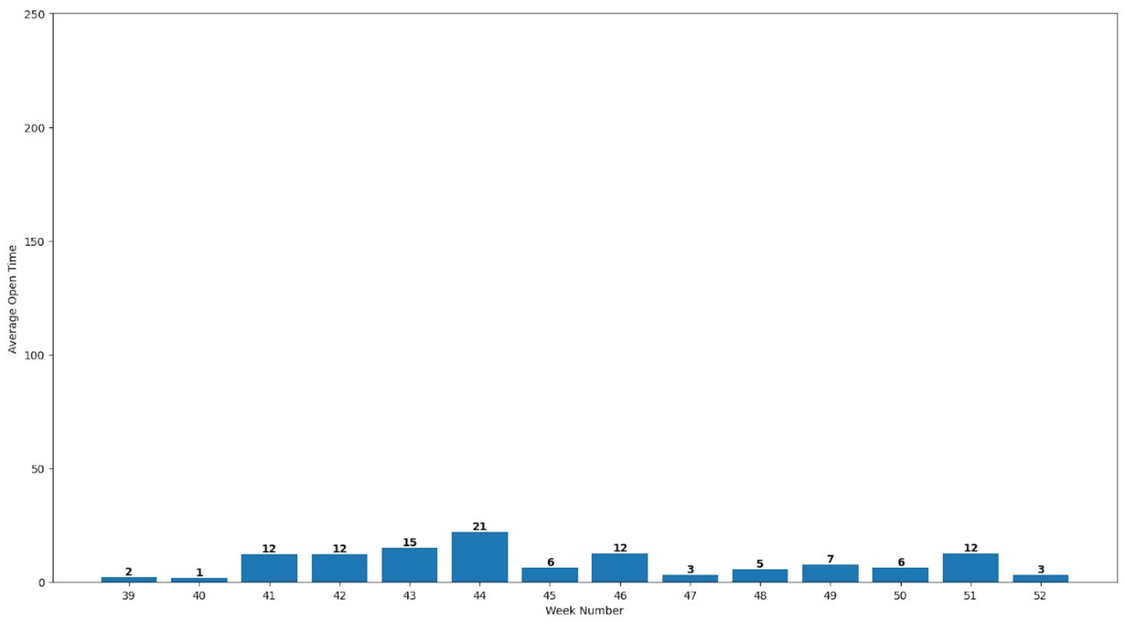


Figure 22: Average hours until case resolution weeks 39-52 (October-December)

5.3.3 The average duration of a vulnerability management process

The following figure demonstrates the impact of an automation that was added to the process of vulnerability management in the beginning of October. Prior to this implementation, alert investigation regarding the existence of potential vulnerabilities in assets was done mainly manually. After the implementation, various potential fixes are automatically applied to the target asset.

Since there is no data describing the manual procedure duration from the previous months, duration was recorded in October of 100 alerts being triaged manually and via the automation procedure. The attempt duration is recorded in seconds. The seconds printed in the following graph were rounded to full numbers for visual clarity. On average, the automation process shows a 61.22% improvement in triage time (Figure 23).

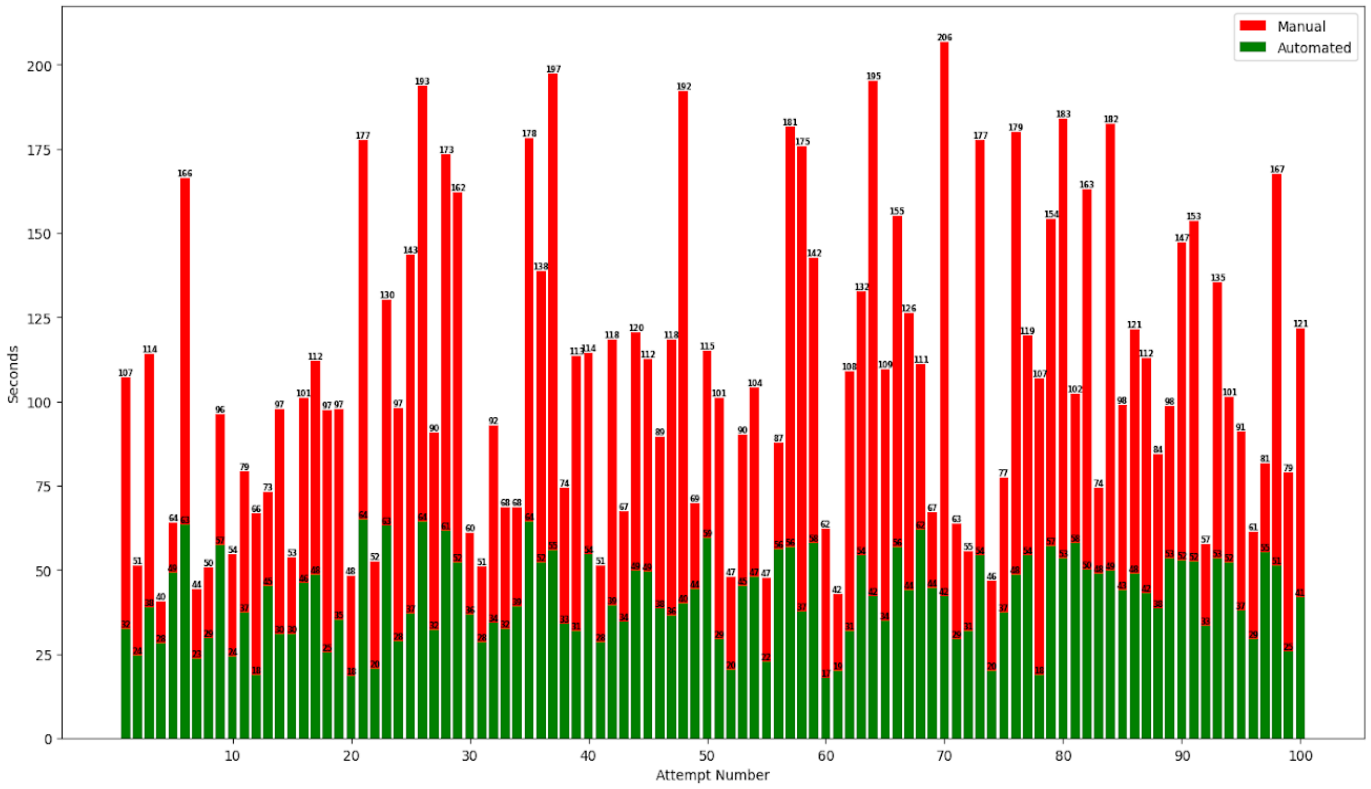


Figure 23: Manual vs automated vulnerability management process

6 Summary

This master's thesis carried out a comprehensive investigation into entry-level SOC workflow automation tools suitable for SMEs. The research involved identifying and defining feature requirements and comparison criteria for such automation tools. Afterwards, two tools that satisfied the selection criteria were chosen and analyzed in detail. After the pros and cons of the aforementioned tools were described, an alternative in-house SOC automation solution was proposed. On top of this, impact assessment was performed by leveraging real-world performance data from the SOC team of company P. After analyzing the statistics, the study successfully demonstrated the efficiency gains a basic automation system with the aforementioned features can have on SOC teams within the context of SMEs. This research contributes valuable insights to the field of cyber security, aiding SMEs in making informed decisions for enhancing their security operations.

References

[1] Gartner, “Small And Midsize Business (SMB)”, Gartner Glossary, Information Technology. [Online]. Available: https://www.gartner.com/en/information-technology/glossary/smbs-small-and-midsize-businesses. Accessed on: Jan. 02, 2024.

[2] Manfred Vielberth, Fabian Böhm, Ines Fichtinger, and Günther Pernul, “Security Operations Center: A Systematic Study and Open Challenges,” IEEE Access, vol. 8, 2020, pp. 227756–227779.

[3] J. Muniz, “The Modern Security Operations Center,” 2021.

[4] N. Gupta, I. Traore and P. M. F. de Quinan, "Automated Event Prioritization for Security Operation Center using Deep Learning," 2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, 2019, pp. 5864-5872, doi: 10.1109/BigData47090.2019.9006073.

[5] T. v. Ede et al., "DEEPCASE: Semi-Supervised Contextual Analysis of Security Events," 2022 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 2022, pp. 522-539, doi: 10.1109/SP46214.2022.9833671.

[6] C. Feng, S. Wu and N. Liu, "A user-centric machine learning framework for cyber security operations center," 2017 IEEE International Conference on Intelligence and Security Informatics (ISI), Beijing, China, 2017, pp. 173-175, doi: 10.1109/ISI.2017.8004902.

[7] O. Cassetto, "Security operations center roles and responsibilities," Exabeam, Foster City, CA, USA, Tech. Rep., 2019

[8] A. Applebaum, S. Johnson, M. Limiero and M. Smith, "Playbook Oriented Cyber Response," 2018 National Cyber Summit (NCS), Huntsville, AL, USA, 2018, pp. 8-15, doi: 10.1109/NCS.2018.00007.

[9] D. Crémilleux, C. Bidan, F. Majorczyk, and N. Prigent, “Enhancing collaboration between security analysts in security operations centers,” in Risks and Security of Internet and Systems, vol. 11391. Cham, Switzerland: Springer, 2019, pp. 136–142.

[10] C. Crowley, B. Filkins, and J. Pescatore, "SANS 2023 SOC Survey," June 2023.

[11] C. Crowley and B. Filkins, "SANS 2022 SOC Survey," May 2022.

[12] C. Crowley and J. Pescatore, "A SANS 2021 Survey: Security Operations Center (SOC)," October 2021.

[13] Bushra A. Alahmadi, Louise Axon, and Ivan Martinovic, “99% False Positives: A Qualitative Study of SOC Analysts’ Perspectives on Security Alarms,” 2022 USENIX Security Symposium, pp. 2783–2800.

[14] A. A. Mughal, “Building and Securing the Modern Security Operations Center (SOC)”, IJBIBDA, vol. 5, no. 1, pp. 1–15, Jan. 2022.

[15] T. Kantola, "Exploring VirusTotal for security operations alert triage automation," Bachelor’s Degree Programme in Information and Communications Technology, February 2022.

[16] S. Norem, A. E. Rice, S. Erwin, R. A. Bridges, S. Oesch, and B. Weber, “A mathematical framework for evaluation of SOAR tools with limited survey data,” in Computer Security. ESORICS 2021 International Workshops. Springer, 2022.

[17] C. Islam, "ARCHITECTURE-CENTRIC SUPPORT FOR SECURITY ORCHESTRATION AND AUTOMATION," The University of Adelaide, School of Computer Science, 2020.

[18] C. Islam and M. A. Babar, "A Multi-Vocal Review of Security Orchestration," ACM Computing Surveys, vol. 52, no. 2, p. 45, 2019.

[19] D. Lalos, "Analysis on Security Orchestration Automation and Response (SOAR) platforms for Security Operation Centers," University of Piraeus, School of Information and Communication Technologies, 2022.

[20] R. Brewer, "Could SOAR save skills-short SOCs?" Computer Fraud & Security, vol. 2019, no. 10, pp. 8–11, Oct. 2019.

[21] R. Bridges, A. Rice, S. Oesch, J. Nichols, C. Watson, K. Spakes, S. Norem, M. Huettel, B. Jewell, B. Weber, C. Gannon, O. Bizovi, S. Hollifield, S. Erwin "Testing SOAR tools in use," Computers & Security, Volume 129, pp. 1-28, 103201, ISSN 0167-4048, 2023.

[22] R. F. Gibadullin and V. V. Nikonorov, "Development of the System for Automated Incident Management Based on Open-Source Software," 2021 International Russian Automation Conference (RusAutoCon), Sochi, Russian Federation, 2021, pp. 521-525, doi: 10.1109/RusAutoCon52004.2021.9537385.

[23] A. Chidukwani, S. Zander and P. Koutsakis, "A Survey on the Cyber Security of Small-to-Medium Businesses: Challenges, Research Focus and Recommendations," in IEEE Access, vol. 10, pp. 85701-85719, 2022, doi: 10.1109/ACCESS.2022.3197899.

[24] N. Deepa, B. Prabadevi, L. B. Krithika and B. Deepa, "An analysis on Version Control Systems," 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), Vellore, India, 2020, pp. 1-9, doi: 10.1109/ic-ETITE47903.2020.39.

[25] A. Anderson, "The State of SOAR: Tines Survey Reveals the Pros and Cons of SOAR Platforms," November 8, 2022. [Online]. Available: https://www.tines.com/blog/the-state-of-soar-tines-survey-of-security-professionals-reveals-pros-and-cons. Accessed: Nov. 15, 2023.

[26] R. S., "Caveats when using SOAR and Git," May 22, 2020. [Online]. Available: https://medium.com/@slick_security/caveats-when-using-soar-and-git-25ee490671f2. Accessed: Nov. 15, 2023.

[27] IBM. "Pricing." IBM QRadar SOAR. Available: https://www.ibm.com/products/qradar-soar/pricing. Accessed: Nov. 16, 2023.

[28] J. Miller, "What is your experience regarding pricing and costs for Palo Alto Networks Cortex XSOAR?" Palo Alto Networks Cortex XSOAR Reviews. [Online]. Available: https://www.peerspot.com/products/palo-alto-networks-cortex-xsoar-reviews. Accessed on: Nov. 18, 2023.

[29] PeerSpot, "Splunk SOAR Reviews." [Online]. Available: https://www.peerspot.com/products/splunk-soar-reviews. Accessed on: Nov. 18, 2023.

[30] Palo Alto Networks. "Community Edition FAQ." Cortex XSOAR. [Online]. Available: https://start.paloaltonetworks.com/rs/531-OCS-018/images/Cortex_XSOAR_Community_Edition_FAQ.pdf. Accessed: Nov. 16, 2023.

[31] "They work with us," Security Vision, Available: https://www.securityvision.ru/work_with/. Accessed: Nov. 16, 2023.

[32] Shuffler. "Pricing." [Online]. Available: https://shuffler.io/pricing. Accessed: Nov. 16, 2023.

[33] n8n. "Self-hosting n8n." [Online]. Available: https://docs.n8n.io/hosting/. Accessed: Nov. 16, 2023.

[34] n8n. "Pricing." [Online]. Available: https://n8n.io/pricing/. Accessed: Nov. 16, 2023.

[35] n8n. (2023). "Streamlining SecOps: Enhancing security and operations with n8n." [Whitepaper]. Available: https://n8niostorageaccount.blob.core.windows.net/n8nio-strapi-blobs-stage/assets/n8n_streamlining_secops_2397b62e64.pdf

[36] Frikky, "Introducing Shuffle — an Open Source SOAR platform part 1," Shuffle Automation, May 20, 2020. [Online]. Available: https://medium.com/shuffle-automation/introducing-shuffle-an-open-source-soar-platform-part-1-58a529de7d12. Accessed on: Nov. 17, 2023.

[37] R. Špitsmeister, "Implementing a SOAR Solution in a Security Operations Center on the Example of Cybers," TalTech, Tallinn, 2023.

[38] Shuffle Features, "Shuffle Features," [Online]. Available: https://shuffler.io/docs/features. Accessed on: Nov. 18, 2023.

[39] Fair-code. “Fair-code,” [Online] Available: https://faircode.io/ Accessed on: Nov. 18, 2023.

Appendix 1 – Non-exclusive licence for reproduction and publication of a graduation thesis[1]

I Giorgi Okroshidze

1 Grant Tallinn University of Technology free licence (non-exclusive licence) for my thesis “Comparison of SOC Workflow Automation Options for SMEs and Practical Impact Analysis,” supervised by Risto Vaarandi

 1.1 to be reproduced for the purposes of preservation and electronic publication of the graduation thesis, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright;

 1.2 to be published via the web of Tallinn University of Technology, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright.

2 I am aware that the author also retains the rights specified in clause 1 of the non-exclusive licence.

3 I confirm that granting the non-exclusive licence does not infringe other persons' intellectual property rights, the rights arising from the Personal Data Protection Act or rights arising from other legislation.

09.12.2023

1 The non-exclusive licence is not valid during the validity of access restriction indicated in the student's application for restriction on access to the graduation thesis that has been signed by the school's dean, except in case of the university's right to reproduce the thesis for preservation purposes only. If a graduation thesis is based on the joint creative activity of two or more persons and the co-author(s) has/have not granted, by the set deadline, the student defending his/her graduation thesis consent to reproduce and publish the graduation thesis in compliance with clauses 1.1 and 1.2 of the non-exclusive licence, the non-exclusive license shall not be valid for the period.