Designing a Cyber-Secure Infrastructure for Belgian SMEs:

A Modular Approach aligned with the CyFun Framework, with Focus on the Accounting Sector

Arnaud Querinjean

Research and Development project owner:
Dr. Jérôme Dossogne

Master thesis submitted under the supervision of
Dr. Jérôme Dossogne

in order to be awarded the Degree of
Master in Cybersecurity
System Design

Academic year
2024 – 2025

I hereby confirm that this thesis was written independently by myself without the use of any sources beyond those cited, and all passages and ideas taken from other sources are cited accordingly.

02/06/2025

Title: Designing a Cyber-Secure Infrastructure for Belgian SMEs

Author: Arnaud Querinjean
Master in Cybersecurity – System Design
Academic year: 2024 – 2025

Abstract

In a context where small and medium-sized enterprises are becoming prime targets of cyberattacks, and these companies face more and more cybersecurity regulations and compliance requirements, it becomes essential to provide them with accessible solutions adapted to their operational reality.

This thesis proposes the adaptation of a pre-existing infrastructure from the JANUS project to meet the requirements of the CyFun framework, developed by the Centre for Cybersecurity Belgium. The main objective was to make this infrastructure compliant with the Identify function of the CyFun framework at the Basic level, focusing on small accounting firms in Belgium. To ground the study in real-world needs, semi-structured interviews were conducted with five Belgian accounting SMEs. Their feedback shaped the functional requirements and validated the usability of the proposed solution.

JANUS is an adaptive infrastructure that uses paradigms like Infrastructure as Code and Configuration as Code. It is entirely built with free and open-source tools. The infrastructure is highly modular and adaptable, allowing it to be used in various use cases, from a cyber-range to a production-ready infrastructure. The project focused on automating asset collection, organizing information, modeling threats, and documenting risks through both technical and manual documentation. Each sub-function of the Identify pillar was implemented using Ansible playbooks, YAML files, and markdown-based organizational policies, to align with standard frameworks.

The solution is mainly based on a modular approach, executable locally, with particular attention to readability and ease of adoption by organizations with low digital maturity. This thesis therefore helps small and medium-sized enterprises get a concrete and operational starting point for applying the CyFun framework and opens the door to future extensions to cover the other functions of the framework (Protect, Detect, Respond, Recover), with the ambition to offer a complete cybersecurity infrastructure for SMEs.

Keywords: modular infrastructure, CyFun, SMEs, Identify function, compliance, adaptive

Preface

This thesis represents the result of a journey full of learning, both technically and personally. It allowed me to strengthen my skills in cybersecurity, in infrastructure engineering, and also to better understand the real challenges small businesses face when trying to be compliant.

What made this thesis especially meaningful was the connection to real-world needs. I regularly interact with small businesses in my daily life, and I have seen firsthand how difficult it can be for them to keep up with growing cybersecurity expectations and regulatory pressure. Having the chance to speak directly with some accounting firms gave the work an extra layer of realism and purpose. These exchanges were some of the most valuable moments of the entire process, helping me shape a solution that is both practical and grounded in real operational challenges.

This experience made me realize how many opportunities still exist to support small and medium-sized businesses with tailored solutions. It also allowed me to put theoretical knowledge into practice through a concrete and meaningful project. Working independently pushed me to make critical decisions and helped me develop a more structured and strategic way of thinking.

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my supervisor, Professor Jérôme Dossogne, for his valuable guidance, availability, and academic rigor throughout this project. His feedback was essential in helping me frame my approach and structure my ideas effectively.

I also wish to thank the members of the jury, Professor Gaël Hachez and Professor Charles Cuvelliez, for their time, availability, and willingness to evaluate my work. A special thanks to my colleagues and classmates, with whom I had the opportunity to exchange ideas, test solutions, and share both doubts and progress during this journey. I am particularly grateful to the five accounting firms who agreed to participate in this study. Their trust, time, and honest feedback were invaluable in shaping a solution that is both relevant and practical.

A heartfelt thank you goes to my family, whose unwavering support helped me stay focused throughout this process. I am especially grateful to my partner, whose belief in me and constant encouragement gave me the strength to stay motivated during the more challenging moments.

Finally, I would like to thank Ruben De Smet and Nick Van Goethem, the authors of the thesis template, as well as all the contributors of open-source resources that greatly facilitated the formatting and presentation of this document.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

CaC	Configuration as Code
CCB	Centre for Cybersecurity Belgium
CD	Continuous Deployment
CI	Continuous Integration
CIA	Confidentiality, Integrity and Availability
CIS	Center fo Internet Security
CPU	Central Processing Unit
CVE	Common Vulnerabilities and Exposures
CWE	Common Weakness Enumeration
CyFun	Cybersecurity Fundamentals (Belgian Framework)
ENISA	European Union Agency for Cybersecurity
ERP	Enterprise Resource Planning
FaaS	Function as a Service
GDPR	General Data Protection Regulation
IaaS	Infrastructure as a Service
IaC	Infrastructure as Code
ISO	International Organization for Standardization
JSON	JavaScript Object Notation
MFA	Multi-Factor Authentication
NIST	National Institute of Standards and Technology
OS	Operating System
OWASP	The Open Worldwide Application Security Project
PaaS	Platform as a Service
RAM	Random Access Memory
SaaS	Software as a Service
SMART	Specific, Measurable, Achievable, Relevant, Time-bound
SMEs	Small and Medium-sized Enterprises
VM	Virtual Machine
VPN	Virtual Private Network
YAML	Yet Another Markup Language
ZAP	Zed Attack Proxy

Chapter 1
Introduction

1.1 Context and Objectives

Accounting SMEs in Belgium need support to control their cyber-risk In a world where cyber threats continue to increase, small and medium-sized enterprises (SMEs) are a prime target for attackers due to their limited resources and often insufficient cybersecurity maturity. This is even more true for accounting firms, where digitalisation is accelerating.

A majority of European SMEs are lacking in cyber According to the 2021 ENISA report *Cybersecurity for SMEs: Challenges and Recommendations* [23], only 28% of the 249 surveyed European SMEs had formally assigned the role of Information Security Officer within their organization. The same report highlights several other challenges faced by SMEs, including a general lack of cybersecurity awareness, insufficient financial and human resources, and the absence of dedicated cybersecurity guidelines. From these observations, we can infer that a considerable proportion of European SMEs operate without the governance structures necessary to manage cyber risk effectively. This is supported by ENISA's statement that "dealing with security threats is hard for non-IT related SMEs" and that there is a "shortage of skills in the population regarding cybersecurity."

Most Belgian business are SMEs Statbel, the Belgian national statistics office, reports that 95.9% of registered businesses in Belgium are micro-enterprises[1]. This figure is based on their national economic survey conducted in 2022 [49]. As such, micro and small enterprises represent not only the vast majority of the economic fabric but also a critical population when considering national-level cybersecurity readiness. Given their dominant presence, the security posture of these entities directly impacts the overall resilience of the country's economic infrastructure.

In Europe, legislation exists to increase cyber-resilience and privacy European regulations, such as the General Data Protection Regulation (GDPR)[2] and the NIS2 Directive [22], impose strict requirements on data protection, risk management, and incident response for organizations of all sizes. The NIS2 directive has been in force across EU member states since October 2024[3]. While these frameworks aim to strengthen cyber-resilience and harmonise standards, many SMEs face difficulties in achieving compliance due to a lack of budget, limited in-house expertise, and insufficient guidance tailored to their context. These challenges have been documented by ENISA, which notes that the cost of implementation, shortage of cybersecurity specialists, and low management support remain key obstacles to effective cybersecurity practices among SMEs [23].

[1] Micro-enterprises are defined by Statbel as companies with fewer than 10 employees and an annual turnover or balance sheet total not exceeding €2 million.

[2] `https://gdpr.eu`

[3] See official directive text: `https://eur-lex.europa.eu/eli/dir/2022/2555/oj`

Lack of compliance is an increased risk Traditional cybersecurity frameworks such as ISO/IEC 27001 are often too expensive and rigid for SMEs, while generic solutions lack the adaptability and automation needed for efficient adoption [6]. The Cyber Fundamentals framework, developed by the Centre for Cybersecurity Belgium, offers a progressive and accessible approach aligned with European requirements and adapted to SME constraints [10]. At the same time, Infrastructure as Code and Configuration as Code tools, illustrated by the JANUS cyber range, make it possible to design modular, secure-by-design infrastructures, offering an alternative for organizations with low technical maturity.

This master thesis addresses this problem by aiming to design a secure and modular infrastructure for Belgian accounting firms, building on the JANUS project and specifically implementing the Identify function of the CyFun framework level basic. This function, which is the first step in the cybersecurity lifecycle, helps identify critical assets, risks, organizational roles, and the policies and procedures needed for effective security governance. It is essential for building the foundations of information security. The objective is to develop a reproducible model to strengthen the resilience of SMEs facing both threats and regulatory obligations.

The proposed infrastructure emphasizes the automation of controls, documentation of processes, and alignment with real-world regulatory and operational requirements.

1.2 Motivations

Despite progress and the abundance of cybersecurity frameworks, Belgian SMEs, and especially accounting firms, remain underprotected. The solutions proposed for SMEs are still unsuitable or too expensive. These frameworks often do not consider limited resources, a lack of qualified staff, and often no cybersecurity awareness. This can potentially lead to financial losses, damage to reputation, or even a complete shutdown of the business [13, 26, 53].

One of the main motivations for this thesis lies in the persistent gap between the economic importance of SMEs and their low level of preparedness against cyberattacks. While large companies often have substantial budgets and dedicated cybersecurity teams, SMEs do not have the necessary capabilities to apply and interpret existing standards due to their complexity and cost [43, 47].

A second driver for this research is the gap between DevSecOps theory and real-world practice. Field investigations show that only 12% of SMEs trigger an automated security scan at each commit, mainly due to tool complexity, implementation difficulties, and a lack of internal skills [11]. Even though it is well known that automation through IaC and CaC helps companies reduce costs, there is still a real need to offer reproducible, maintainable, and well-documented playbooks [44]. This is precisely the goal of the JANUS project, which provides a secure-by-design private cloud foundation to be adapted to the organizational and regulatory constraints of accounting firms.

Another motivation for this research is the lack of solutions tailored to the specific needs of SMEs. Existing tools are often effective for large enterprises with the necessary budgets and expertise, but they are too technical, expensive, and rigid for SMEs. There is a need for simplified solutions that reflect the real level and often nonexistent cybersecurity skills in small businesses, especially in accounting firms, where there are often no technical IT skills at all. Most studies are limited to general recommendations or one-time audits,

without offering concrete tools for daily implementation [47]. This research aims to explore strategies to strengthen the resilience of these companies while taking their context into account.

Finally, regulatory compliance, such as GDPR and the new NIS2 directive, is a critical concern. These regulations impose strict obligations, even though SMEs often lack the knowledge to comply [54]. By addressing these challenges, this thesis seeks to contribute to the design of more appropriate tools that promote accessible cybersecurity for accounting firms, thus enhancing the resilience of SMEs.

1.3 Project statement & contributions

In this thesis, we focus on strengthening the cybersecurity of Belgian small and medium-sized enterprises, especially accounting firms, by proposing a modular and automated infrastructure using IaC and CaC. The goal is to support compliance with the Identify function of the CyFun framework at the Basic level.

The project is based on adapting the existing JANUS platform. This adaptation will help meet the requirements of the Identify sub-functions. This function is often neglected in SMEs, where the focus is sometimes on other areas of cybersecurity. However, this creates a problem for having an effective strategy, because we cannot protect what we do not know.

There is a fundamental need for companies to first understand their inventories and the risks they face before trying to implement solutions that may not be appropriate once the real risks are identified. The Identify pillar includes identifying critical assets, defining roles and responsibilities in terms of security, performing risk analysis, and setting governance policies.

Important note

Please note that the CyFun framework does not have a separate Governance pillar like NIST CSF 2.0 [39]. Governance-related sub-functions are included directly under the Identify pillar.

To meet this objective, the project delivers several concrete outcomes:

- **Empirical validation with accounting firms:** To validate the project hypothesis, a qualitative survey was conducted with five accounting firms in Wallonia. Structured interviews helped better understand their needs, ways of working, and infrastructure, as well as their level of awareness and technical or organizational constraints. The feedback directly influenced the design of the proposed tools.
- **Automation of the Identify function:** Using Ansible playbooks, the infrastructure can partially automate the inventory of assets such as virtual machines and the services running on them. This helps generate structured documentation files that support compliance with different regulations.
- **Provision of ready-to-use checklists:** Several checklists were created to help SMEs implement each CyFun sub-function. Since 100% automation is not always possible, and to remain modular and compliant, we provided markdown checklists named after each sub-function (e.g., `ID.AM-1`). These offer concrete steps, verification criteria, real examples, and improvement recommendations.

- **Modular design and focus on reusability:** Our approach ensures easy integration in various SME environments. Components can be activated or disabled based on the company's context and which services it needs to use.

- **Clear and contextualized documentation:** Each implementation step is supported by accessible, non-technical documentation to allow independent usage without the need for advanced technical expertise.

This thesis does not aim to cover all CyFun framework functions, nor to offer a complete cybersecurity solution. It deliberately focuses on the initial Identify phase, using a tested and in-depth implementation logic. The Protect, Detect, Respond, and Recover functions are considered in the overall structure, but are not implemented in this work. They could be developed in future work to complete CyFun implementation.

The project therefore provides the beginnings of a realistic and replicable concept for a minimalist but effective infrastructure. It takes into account the actual needs and resources of small accounting firms while staying modular. This work is progressive and also helps bridge the gap between theoretical frameworks and real-world implementation by clearly describing the steps to follow to meet the first CyFun function at the Basic level.

1.4 Research Questions

In addition to the objectives outlined above, this thesis aims to address the following research questions:

- **Question 1**: Is it possible to design a secure, modular infrastructure that complies with the CyFun framework, while remaining adapted to the technical and organizational constraints specific to small accounting firms in Belgium ?

- **Question 2**: To what extent can the automation of cybersecurity functions, particularly the Identify function, help compensate for the lack of specialized personnel within SMEs ?

- **Question 3**: Can the JANUS project be adapted to serve as a reusable infrastructure model for other SME sectors, while ensuring scalability, understandable documentation, and security by design ?

1.5 Organization of this document

This thesis is structured into nine chapters, each addressing a key step in the scientific process. The document includes:

- **Chapter 2 - Background**: This chapter provides the necessary technical concepts to understand the project, including the principles of Infrastructure as Code, the JANUS platform, and the CyFun framework.

- **Chapter 3 - State of the Art**: It offers an overview of related papers, highlighting the cybersecurity challenges of SMEs, the limitations of current approaches, and the opportunities for innovation. This literature review helps position the thesis in an academic context.

- **Chapter 4 - Project's Mission, Objectives and Requirements**: This chapter presents the concrete goals of the project, the field needs identified through interviews, and the constraints to follow.
- **Chapter 5 - Experimentation & Data Collection**: It details the method used to build the solution, the tools implemented, and the qualitative survey conducted with five accounting firms.
- **Chapter 6 - Implementation of the CyFun Identify Function**: This chapter explains how the proposed solution was designed and implemented to support CyFun Basic Identify compliance.
- **Chapter 7 - Cybersecurity Analysis of the Proposed Solution**: This part critically evaluates the proposed solution and its implementation.
- **Chapter 8 - Discussion**: It provides a detailed reflection on the results, a comparison with related work, and discusses the limits of the solution.
- **Chapter 9 - Conclusion**: The final chapter summarizes the key contributions of the thesis and the main takeaways from the research.

Chapter 2
Background

2.1 Introduction

This chapter presents the foundations needed to understand the project. It allows us to understand the design of a cyber-secure infrastructure intended for small and medium-sized enterprises, especially accounting firms. It introduces the essential concepts to understand the organizational context of SMEs, the technical principles used in the proposed solution, and an explanation of the JANUS project. This project relies on a modular architecture and tools like Vagrant, Ansible, Docker, virtualization, and a "secure by design" approach.

2.2 Small and Medium-Sized Enterprises

Small and medium-sized enterprises (SMEs) represent a large part of the European and global economy. According to the official definition by the European Commission, a company is considered an SME if it meets three criteria: it employs fewer than 250 people, has an annual turnover of less than 50 million euros, or a balance sheet total not exceeding 43 million euros [20, 21].

In Belgium, more than 99% of businesses fall into this category, showing their fundamental importance [33]. SMEs operate in many sectors such as retail, industry, and professional services like accounting firms. Despite their economic weight, they usually have limited resources to face cybersecurity requirements, whether technical, human, or financial. Note that the definition of SMEs may vary depending on the region and local government.

2.3 Virtualization

Virtualization is a key technology in modern system architecture, especially in the deployment of secure, isolated, and reproducible environments. It consists of running multiple operating systems or software environments on the same physical hardware, by creating "virtual machines" (VMs) that share the host's resources while remaining logically independent [50].

Virtualization relies on a hypervisor, a software layer that partitions physical resources and allocates them to different virtual machines. There are two main types of hypervisors:

- **Type 1:** A hypervisor that runs directly on the physical hardware, also called a bare-metal hypervisor [57].
- **Type 2:** A hypervisor that runs on top of a host operating system, like VirtualBox or VMware Workstation.

In this project, virtualization is used for several reasons. First, we want to isolate components in different VMs, for example separating the test environment from the production environment, reducing the risk of cross-compromise.

Then, virtual machines allow us to test and replicate infrastructure in a deterministic way, which is important for this project in order to have automation that works reliably, regardless of the company where we deploy it.

Finally, we want maximum flexibility: by changing only a few parameters, we can adapt the infrastructure for deployment in a company or as a cyber-range, simply by modifying configurations.

Virtualization is important for implementing Infrastructure as Code (IaC) principles, as it allows us to dynamically provision environments using tools like Vagrant or Ansible. It also plays a strategic role in the secure-by-design approach [15] by enabling the creation of isolated environments for applications and data, reducing the attack surface and improving security.

2.4 Docker

Docker is a lightweight virtualization tool based on containers. It allows packaging an application and all its dependencies into a portable and executable unit [19]. Unlike traditional virtual machines that include a full operating system, containers [18] share the host system's kernel while remaining isolated from one another. This approach allows faster startup, lower system footprint, and better portability between environments.

A Docker container is based on an image, which describes the complete state of a system and can be instantiated as needed. This "build once, run anywhere" philosophy is central to modern DevOps and IaC practices. In our project, it is used for each service or component to ensure clear separation of responsibilities.

Docker ensures the portability of services running in our project, as configurations tested on a development machine can be replicated exactly in production without behavioral differences. Docker also allows us to automate tasks and deploy a full set of ready-to-use services in seconds. Finally, it helps limit the attack surface of each service and enforce restriction policies.

2.5 Vagrant

Vagrant is an open-source tool developed by HashiCorp that manages the full lifecycle of virtualized development environments. It is an environment manager designed to work with virtualization providers like VirtualBox, VMware, or Hyper-V [28].

Vagrant automates the creation, configuration, and destruction of virtual machines using a simple text file named `Vagrantfile` [27]. This file defines the base image (e.g., Ubuntu 22.04), allocated resources, open ports, synchronized folders, and configuration scripts.

In our project, we use Ansible scripts as configuration, which are executed during initialization. The benefits of using Vagrant include the ability for each developer to generate a local environment identical to the production one, improving reproducibility. Each environment is also isolated to ensure software independence. Finally, Vagrant is designed for easy adaptation and automation with tools like Ansible, enabling fast deployment. In summary, Vagrant serves as the local orchestration layer.

2.6 Ansible

Ansible is a tool for automating configuration, deployment, and infrastructure management. It was originally developed by Michael DeHaan and is now maintained by Red Hat [5]. It uses an agentless approach, meaning that no software needs to be installed on target machines. Only SSH and Python are required, making it easy to adopt.

Ansible allows writing playbooks [4] in YAML format, describing the desired state of a system. For example, which packages should be installed, which services should be running, etc. This Configuration as Code logic ensures reproducibility by minimizing human error.

In our project, Ansible helps us standardize system configurations, deploy more securely, and automate the collection of artifacts.

2.7 JANUS

JANUS is a modular infrastructure platform initially developed as a private cyber range. Its original goal was to create a virtualized environment to simulate cybersecurity scenarios for training, educational, or technical validation purposes. The project is fully based on the IaC and CaC paradigms, using tools like Vagrant, Ansible, VirtualBox, and Docker to ensure automation and reproducibility of deployments.

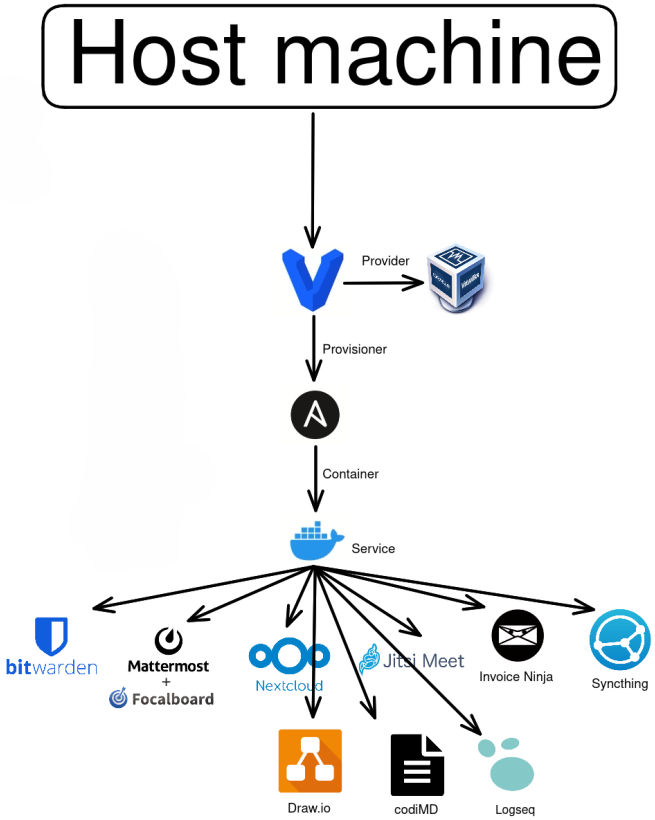

Figure 2.1: Infrastructure JANUS

Thanks to its modular structure, each component can be activated, modified, or replaced independently, making safe experimentation easy. In the context of this thesis, the JANUS project has been adapted and extended to meet the real needs of SMEs, with the goal of aligning the infrastructure with the CyFun framework.

Instead of keeping a simulation-only logic, the goal was to transform JANUS into an operational and secure infrastructure, directly usable in real environments. This is a strong point for the project because JANUS already has a high level of modularity, with the possibility to activate only the services that are needed. Deployment is already automated, and the entire environment can be launched on a local machine with a single command. This allows SMEs to benefit from a complete environment without needing to mobilize an IT team.

JANUS therefore becomes a multi-purpose project where the infrastructure can be used as a production platform, a testing environment, or a training lab. The integration of the CyFun framework focuses on the full implementation of the Identify function, but the structure is designed to allow easy future integration of the Protect, Detect, Respond, and Recover functions.

2.8 Cyfun

The Cyber Fundamentals cybersecurity framework developed by the Centre for Cybersecurity Belgium is a framework designed to provide Belgian SMEs with a simple, progressive, and operational reference to strengthen their cybersecurity posture [7]. Its main objective is to be usable by any kind of entity, from small businesses to large organizations.

It has different assurance levels: Basic, Important, and Essential, with an additional Small level that does not provide an assurance grade but allows an organization to perform an initial self-assessment.

The CyFun framework is structured around 5 core functions:

- **Identify** (identify assets, risks, and responsibilities)
- **Protect** (implement technical and organizational protections)
- **Detect** (detect incidents)
- **Respond** (respond to incidents)
- **Recover** (restore activities)

Each function contains sub-functions and comes with checklists to verify whether the measures are present in the organization. Once all the sub-functions of a function are validated, the whole function is considered compliant.

In the context of this thesis, the Basic level was selected as the target, as it reflects the reality of most accounting firms interviewed. This level helps structure a minimal but realistic security baseline. CyFun is easy to understand, which makes it suitable for SMEs that do not have cybersecurity experts.

2.9 Secure by Design

The *Secure by Design* approach is one of the foundational pillars of the JANUS project. It consists of proactively integrating security from the very first stages of design, architecture, and development of a product or service, rather than adding it later as an extra layer. This approach implies a holistic vision of cybersecurity at the organizational level, with the primary objective of reducing the number of vulnerabilities from the design phase and throughout the system's entire lifecycle [8].

In the context of the JANUS project, the Secure by Design approach is reflected in several technical and organizational choices:

- **Isolation of critical components:** achieved through virtualization and containerization;
- **Standardization and automation of configurations:** made possible by tools such as Ansible, enabling reproducible, traceable, and compliant environments;
- **Controlled modularity:** allowing only the necessary services to be activated, thereby reducing complexity and associated risks.

This security-by-design logic is particularly relevant in the context of SMEs, where human and technical resources are often limited. By integrating security from the start, it not only helps avoid future remediation costs but also provides a stronger foundation for compliance with cybersecurity requirements.

Chapter 3
Literature review, state of the art (SotA), definitions and notations

3.1 Introduction

This chapter reviews literature and practices related to designing secure IT infrastructures for small and medium-sized enterprises (SMEs), with Belgian accounting firms as a use case. It focuses on adapting modular, secure-by-design systems, like the JANUS cyber-range, to meet operational and compliance needs (e.g., CyFun, NIS2) through automation and configuration tools.

3.2 Divide and conquer: From title to sub-questions

To address this problem, we first break down the main problem into subquestions that guide the literature review, then define the objectives of this state-of-the-art to align with these subquestions.

Sub-Questions

The following sub-questions decompose the main problem into key areas of investigation:

1. **What are the cybersecurity challenges and infrastructure needs specific to Belgian accounting firms compared to general SMEs?** This sub-question identifies risks (e.g., financial data breaches), operational constraints, and regulatory requirements (e.g., NIS2, GDPR) in the accounting sector.

2. **What cybersecurity practices, tools, and organizational strategies are currently recommended for securing SME IT infrastructures in a modular and cost-effective way?** This sub-question reviews traditional and modern approaches (e.g., antivirus, IaC, DevSecOps) and their applicability to accounting firms.

3. **How can modular, secure-by-design infrastructures leverage automation and configuration tools to enhance cybersecurity for SMEs, and what are the associated challenges?** This sub-question explores the use of tools like Infrastructure as Code (IaC), Configuration as Code (CaC), and CI/CD in modular architectures, drawing from projects like JANUS, a cyber-range designed for training and simulation.

4. **How can the CyFun Framework be applied to ensure compliance and security for Belgian accounting firms, and how does it compare to other frameworks in supporting infrastructure adaptation?** This sub-question examines CyFun's practical implementation, its alignment with Bel-

gian regulations, and its role in adapting existing infrastructures like JANUS for operational use.

These sub-questions are interconnected: understanding the specific challenges and needs (sub-question 1) informs the evaluation of recommended practices (sub-question 2), which supports the design of modular, automated infrastructures (sub-question 3) and their alignment with regulatory frameworks like CyFun (sub-question 4).

3.3 Definitions

In the state of the art, several core concepts are frequently used in the cybersecurity scientific literature. To ensure a clear and shared understanding, this section presents the definitions used in this thesis, specifying their origin and relevance in this context.

3.3.1 List of Definitions

Configuration as Code (CaC)

The concept of Configuration as Code refers to the automated management of internal settings of systems, services, and applications using declarative scripts. CaC focuses on internal states, often expressed in declarative formats like YAML or JSON. This approach is one of the practical foundations of modern DevOps, where configuration changes are treated as full code [58].

Infrastructure as Code (IaC)

Infrastructure as Code allows describing, deploying, and managing IT infrastructure using declarative configuration files, often versioned in source code control systems. This approach promotes reproducibility, auditability, and large-scale automation. IaC uses a high-level descriptive coding language to automate the provisioning of IT infrastructure [31].

Cyber-range

A cyber-range is a simulated, controlled, and isolated environment used to train, test, and evaluate cybersecurity skills or tools. These platforms allow realistic scenarios to be executed, with complete infrastructures [36]. It is used both for training and to validate solutions or simulate incidents. This concept is particularly relevant to the JANUS project, which alternates between production use and training use.

3.4 Cybersecurity Challenges for SMEs and Accounting Firms

Small and Medium-sized Enterprises (SMEs) are a major part of the global economy. In Europe, they alone represent 99.8% of all businesses, making them essential to the global economy [43]. However, they are poorly prepared for cyberattacks and therefore represent a critical vulnerability point for our economy. Following our research in the literature, we identified a list of different challenges related to SMEs:

3.4.1 Lack of awareness

One of the most frequently mentioned issues is the lack of cybersecurity awareness. SMEs often believe they are too small to be targeted [1], or not interesting enough for cybercriminals. They either ignore their exposure to threats or underestimate it. This leads to a low perception of risk among both employees and executives, which makes them an easy target for attackers.

Sometimes, they are aware of the risks, but the company lacks the technical knowledge to properly implement solutions and make the right decisions. As a result, companies allocate their budgets to the wrong resources and don't follow good cybersecurity practices [47].

3.4.2 Human factors

The human factor is often recognized as the weakest link in the cybersecurity chain. Over 85% of cybersecurity incidents are caused, either unintentionally or intentionally, by humans [16]. This highlights the importance of training and raising employee awareness. In the literature, many suggest awareness training as a solution to this issue.

However, the problem is often not negligence, but rather that awareness trainings are inadequate, too theoretical and therefore ineffective. As Abdulmajeed Alahmari et al. explain [1], 85% of employees are aware of best cybersecurity practices, but only 54% follow them properly. This supports the findings of Friday Ugbebor et al. [12], where traditional awareness training approaches (like PowerPoints, abstract theory, etc.) are seen as too generic and ineffective. They emphasize the need for alignment with company culture. Therefore, it's crucial to adapt the training based on employee roles, as well as the company's environment and profile.

3.4.3 Limited resources

Financial and human limitations are among the most frequently cited problems in the literature. Companies lack the budget and adequate staff to have a dedicated cybersecurity team. This makes it difficult for them to remain compliant. In the study by Empirical Analysis of NIS2 Adoption in EU SMEs [54], 78% of SMEs surveyed said that budget constraints are the main barrier to NIS2 compliance. This lack of resources compromises the basic protection of companies. Even when a company invests in cybersecurity, the investments are often misdirected. Some use tools without configuring them properly, or rely on free solutions designed for personal use. There's a mismatch between the real needs and the internal capabilities of a company, which creates a false sense of security and leaves systems exposed [53]. Many SMEs also rely on outdated IT infrastructures that were deployed in a trial-and-error way, without a comprehensive security architecture. This can result in infrastructures that are not, or only rarely, updated, making it difficult to implement effective protection measures. On top of that, the rapid digitalization of SMEs has popularized the use of cloud, SaaS, IaaS, FaaS, and connected devices (IoT), which has increased their attack surface. These technologies are rarely supported by a clear framework or adequate governance [1, 6].

3.5 Complex Standards and Frameworks

One of the main obstacles SMEs face in terms of cybersecurity is the unsuitability of standard frameworks. Even though frameworks like ISO/IEC 27001 or the NIST CSF are industry-recognized, they're often too complex, expensive, and ill-suited for the reality of small businesses. SMEs usually lack the maturity and resources to implement them [6, 45, 47]. The scientific community tends to agree on one point: it's necessary to develop lightweight, customizable, and progressive cybersecurity frameworks that still comply with European regulations (GDPR, NIS2). We notice that current standards are generally designed for organizations with a dedicated cybersecurity team, an internal IT department, and legal support. In the case of SMEs, these roles are often outsourced or don't exist at all. This makes the implementation of such frameworks more complicated and sometimes unfeasible without external help. Several authors also highlight the lack of SME involvement in the development of standards. SMEs are rarely represented in standardization committees or European research projects, which leads to frameworks that don't reflect their real needs [43].

Need for Tailored Tools

Thomas Joswig et al. [54] emphasize the need for pre-configured, easy-to-deploy tools that are aligned with European standards.

3.5.1 Specific Risks for Accounting Firms

Among SMEs, accounting firms are in a particularly vulnerable position when it comes to cyber threats. This is due to the sensitive data they handle, their access to financial records, and client information. These firms heavily depend on digital tools to manage financial and tax-related data, which makes them especially exposed.

They deal with both their own and their clients' sensitive data (balance sheets, bank info, payroll, tax IDs, etc.). A data loss would threaten not only their own operations, but also those of all their clients. This double exposure makes accounting firms a strategic target for cybercriminals [26].

Employees in accounting firms are often non-technical, which increases the risk of phishing and social engineering attacks. There's a general lack of awareness, and a tendency to underestimate their exposure, thinking mistakenly that only large companies are targeted. As a result, investment in cybersecurity tends to be low, with few formal security policies in place. Leadership is sometimes unaware that they can be held legally responsible in the event of a data breach [48].

The most common attacks observed in this sector include:

- **Phishing attacks**: accounting firms receive daily emails from clients or institutions, which makes them highly vulnerable to phishing-type attacks. Attackers impersonate a client or institution to obtain information such as passwords, usernames, financial data, or to encourage the opening of malicious links or attachments.
- **Ransomware:** By encrypting client data, accounting files, and sensitive documents, attackers can completely block a firm's operations and prevent them from working. This can lead to delays in tax filings, financial losses, or even risks of regulatory sanctions.

- **Insider threats:** Sometimes, the threat can come from within. If a disgruntled or careless employee accesses confidential data, they could leak it.

These frequent threats can have huge consequences for accounting firms. They risk financial losses, losing client trust, legal action, and rising cyber insurance costs [26] [37].

Just like SMEs, accounting firms often lack the necessary resources and expertise to implement proper security standards. They don't have the budget to hire experts or run regular audits. Despite the risks, small accounting firms remain largely unprepared. As they increasingly adopt digital systems, they expand their attack surface. Even in large firms, best practices aren't always applied. The Deloitte cyberattack in 2016 is a good example [25]: just not having multi-factor authentication (MFA) in place left a door open for attackers. This shows that cybersecurity is not only a matter of budget, but also of properly implementing recommended measures.

3.6 Regulatory and Security Frameworks

The European economy is facing a growing wave of cyber threats due to increased digitalization. Institutions have responded with several initiatives to regulate cybersecurity. However, while these regulatory frameworks are necessary, they pose huge challenges for SMEs trying to comply. This section explores the evolution of cybersecurity governance frameworks.

Until recently, most SMEs were not subject to regulatory cybersecurity requirements. But with the enforcement of the European Union's NIS2 Directive (Network and Information Security Directive 2), small entities operating in critical sectors (finance, energy, health, etc.) now face new obligations.

NIS2 sets out several requirements in terms of:

- Risk governance
- Incident reporting within 24 hours
- Employee training
- Supply chain security
- Business continuity and incident response plans

However, a 2025 empirical study [54] shows that nearly one in two SMEs still does not know whether NIS2 applies to them. Even when they are aware, the main barriers to compliance are budget constraints (78%), lack of expertise, and regulatory complexity.

Given the complexity of standards like ISO/IEC 27001, local entities have introduced more accessible alternatives. In Belgium, the Centre for Cybersecurity Belgium created the Cyber Fundamentals framework, a lightweight certification scheme designed to be progressive and SME-friendly [9, 45].

CyFun aims to serve as a bridge between basic best practices and a gradual path toward more complex standards. It is structured in three steps:

- **Self-assessment:** using a questionnaire based on the NIST CSF functions
- **Commitment declaration:** with remediation of the identified gaps
- **Progressive certification:** by an accredited body

The main issue identified in the literature is not the absence of standards, but rather their lack of suitability. These frameworks are designed for large enterprises and are too rigid [13, 43]. This makes CyFun a good candidate to help SMEs become compliant.

Review of Governance Frameworks for SMEs in the EU

To better adapt to SMEs, several alternative frameworks have emerged in recent years. Each offers a different approach to help SMEs meet cybersecurity challenges while minimizing complexity.

KIS (Keep It Secure) In 2019, the Keep It Secure (KIS) initiative was launched as part of the Cyberwal strategy by Digital Wallonia, supported by the Agence du Numérique, INFOPOLE Cluster TIC, and the research centers CETIC and Multitel [32]. KIS addresses a clear issue: most Walloon SMEs in Belgium have no formal cybersecurity governance or policies, due to a lack of information, human and financial resources, and the topic's complexity. KIS proposes a two-part approach:

- **Trust, Keep It Secure label:** cybersecurity experts are selected based on technical interviews and criteria to assist SMEs, ensuring their competence.
- **Financial support (cybersecurity vouchers):** funding is available for SMEs to undergo cybersecurity audits by certified providers, followed by optional coaching.

KIS uses the NIST Cybersecurity Framework (Identify, Protect, Detect, Respond, Recover) and CIS Controls as references to provide a simple and structured support process. This framework does not replace standards like CyFun or ISO27001 but offers a human and financial entry point into cybersecurity [45].

PUZZLE Framework In 2020, the PUZZLE Framework was developed as part of European research projects (H2020), targeting cybersecurity for SMEs. The framework takes a hybrid approach, combining regulatory compliance and adaptability to small structures' realities by helping them monitor their risk levels. Recent literature (Bountouni et al. [6]) suggests PUZZLE as a suitable solution for SMEs, highlighting its modular design, adaptable to the organization's maturity level. It also integrates automation tools and dashboards to make security easier without requiring deep technical skills. The project is still under development.

CyFun In 2020, the Centre for Cybersecurity Belgium developed the CyFun framework, a national scheme aimed at boosting SME resilience. CyFun is based on four key frameworks (NIST CSF, ISO 27001/27002, IEC 62443, and CIS), ensuring alignment with European standards. These four were chosen because they are the most widely used in Belgium.

The goal is for the framework to be accessible to all, from micro-businesses to large enterprises. CyFun is built on four assurance levels: Small, Basic, Important, and Essential. Entities begin with a free self-assessment using the CyFun selection tool. Validation is then done via a declaration of commitment, followed by certification from an accredited organization [9].

ASMAS (Adaptable Security Maturity Assessment and Standardization) In 2022, Ozkan et al. [42] introduced ASMAS, a maturity framework designed specifically for digital SMEs. It analyzes an organization's profile based on different criteria (sector, size, threat exposure) to assess its risk posture. ASMAS creates implementation groups and enables contextual self-assessments. It supports gradual adoption with qualitative measures and helps raise cybersecurity awareness. While promising, it remains mostly theoretical and requires significant customization to be ready for environments like accounting firms.

3.7 Infrastructure need for SME

This section explores the actual technical needs of SMEs in cybersecurity. These needs have been analyzed considering their limited resources, lack of expertise, and regulatory requirements with a focus on SMEs in the accounting sector. They handle sensitive data, yet their IT infrastructure is often inadequate to face rising cyber threats.

Infrastructure constraints of SMEs

SMEs often have empirically deployed IT infrastructures with no global security architecture. As highlighted by Bountouni et al. [6], many use outdated systems, rarely updated, which makes implementing modern protection measures complicated. The fast digitalization has generalized the use of cloud services (SaaS, IaaS, FaaS), but these technologies are rarely accompanied by proper governance or clear security policies. For instance, accounting firms rely on digital tools (ERP, tax management software), but the absence of secure configurations exposes client data to significant risks [37].

Technical needs

SMEs require a technical infrastructure that balances accessibility, efficiency, and compliance. The needs identified in the literature include:

- **Secure backups:** Backups are essential to ensure business continuity against various attacks, particularly ransomware, which is devastating for accounting firms [48]. Regular, offline-stored backups are necessary to comply with CyFun. But as noted by [1], testing restore procedures is crucial, often neglected, making backups unusable during a crisis. An automated solution with regular testing and encrypted storage is required.

- **Network segmentation:** Network segmentation is another critical need identified in several studies. It limits the spread of an attack by isolating parts of the network based on risk levels. Sensitive resources like client data must be separated from administrative workstations, adding a layer of protection. Joswig and Kurz [54] show that unsegmented servers and VPNs pose high risks, with 38% of attacks exploiting this vulnerability.

- **Secure access and VPN:** One of the first technical needs for SMEs is setting up secure connections for remote employees. This involves using VPNs to create encrypted tunnels over insecure networks and applying multi-factor authentication (MFA) to protect access to company resources. Papathanasiou et al. [2] emphasize VPN with MFA as a baseline measure. The case study by Kafi [37] confirms the need for secure access against cyber threats.

- **Monitoring and detection tools:** Active network monitoring using tools like SIEM (Security Information and Event Management) or IDS (Intrusion Detection Systems) enables real-time threat detection such as suspicious access. Chidukwani et al. [12] show that less than 50% of SMEs regularly monitor their logs. For accounting firms, a simplified interface and alerts for non-experts are needed. The PUZZLE framework addresses this partly through AI-based network analysis modules adapted for SMEs.

- **Integrated training:** Integrating training modules into infrastructure is crucial. For accounting firms, phishing simulations and contextualized training (e.g., fake client emails) improve resilience [26]. An infrastructure like JANUS could include automated training modules.

- **Updates and patch management:** Another critical need concerns regular software updates and patch management. An infrastructure using outdated software is vulnerable to known exploits. The Equifax security breach, caused by a missed patch, highlights the importance of a rigorous patch management process for all businesses, including SMEs [37].

- **Data encryption:** Encryption protects sensitive data, a critical need for accounting firms handling fiscal and banking information. GDPR and CyFun require encryption at rest and in transit. SMEs must implement end-to-end encryption for communications and data storage. Kafi [37] notes that small businesses often skip encryption by default due to its complexity. Solutions must offer simplified, ready-to-use configurations.

key element to remember from this section

Accounting firms need infrastructure that enables encrypted communication and data storage, secure access, and network segmentation. They also require regular backups and the use of monitoring tools and update management.

3.8 Automation & IaC for SMEs

Automation and Infrastructure as Code (IaC) are approaches designed to meet the cybersecurity needs of small and medium-sized enterprises (SMEs). These technologies allow for the secure deployment and management of IT infrastructures, addressing common limitations in resources and technical expertise [47]. This section explores automation and IaC strategies, their advantages for SMEs, and their alignment with frameworks like Cyfun.

Automation and IaC Approaches

Cybersecurity automation uses tools and scripts to perform repetitive tasks such as vulnerability scanning, patch management, and compliance validation, minimizing manual interventions [2]. IaC enables infrastructure definition and provisioning through code (e.g., Ansible), ensuring logical and consistent configurations [34]. These approaches are essential components of DevSecOps pipelines that integrate security, development, and operations for secure software management [11].

Secure CI/CD Pipeline Automation

Automated CI/CD pipelines have security tools such as SAST, DAST, and SCA. While 68% of SMEs have adopted DevSecOps, only 12% perform security scans at each commit [11], highlighting the need for full automation. Tools like SonarQube, OWASP ZAP, and Snyk can identify vulnerabilities early in the development process, reducing exposure in production environments. For accounting firms, this proactive detection is vital against common software threats [48]. Open-source solutions such as Checkov and tfsec, embedded within pipelines, offer cost-effective vulnerability detection [35].

IaC for Reproducible Configuration

IaC tools like Terraform and Ansible allow cloud or on-prem infrastructure to be codified and version-controlled. Security practices such as encryption at rest or IAM restrictions are commonly applied [56], although gaps remain, especially in data encryption and log monitoring. A CI/CD pipeline proposed by Mangla integrates Terraform with tfsec and Semgrep to automatically identify over 155 misconfigurations [35]. This demonstrates that SMEs can achieve compliance with frameworks like Cyfun without deep expertise.

Security as Code

The Security as Code paradigm [29] uses open models (e.g., JSON, AsciiDoc) to represent and validate compliance requirements (e.g., GDPR, NIS2) within DevSecOps pipelines. Tools like Cyberismo enable the creation of "cards" for each security control or regulation, linking and automatically validating them to replace manual audits. For SMEs and especially accounting firms, this approach helps to reduce complexity. complexity and simplifies the tracking of obligations such as client data encryption mandated by GDPR [25].

Integrated Training

Automation also extends to workforce training. In accounting firms, where personnel are often non-technical, embedding educational modules in automated infrastructures can reduce human error [32].

Value for SMEs

Automation reduces the need for manual intervention, enabling effective cybersecurity management without dedicated teams and lowering costs. Hybrid PaaS solutions with auto-scaling and IaC tools like Terraform can reduce infrastructure costs by up to 20% [46]. Given SMEs' vulnerability to ransomware and phishing, automated DAST tools help detect runtime threats like XSS or SQL injections in real time [55]. IaC ensures consistent configurations, mitigating attack risks. It also enables modular infrastructure, where components can be activated based on need. GitOps platforms built on Kubernetes and ArgoCD are particularly adaptable and cost-effective for SMEs [52].

Reminder

Automation via IaC and CI/CD pipelines provides good solutions to major SME challenges, especially in accounting. These approaches lower costs, simplify compliance, and improve resilience to threats, allowing SMEs to deploy secure infrastructures

. without requiring deep expertise.

3.9 Modular Secure Infrastructure

3.9.1 Need for Modularity

The modularity of infrastructures allows for the separation of critical functions into independent components, enabling evolution, easier maintenance, and adaptability. Despite this principle being well-known in software engineering and complex systems, it is rarely used in smaller infrastructures. As described earlier, the PUZZLE framework [6] demonstrates the benefits of a modular approach, with security components that can be activated based on the company's maturity level. This separation of components allows SMEs to deploy solutions step by step, gradually integrating features instead of implementing individual, more costly solutions. Modularity is therefore not just a best practice but also enhances infrastructure resilience and security.

3.9.2 A Recurring Gap in the Literature

We noticed that although there is an abundance of work on SME cybersecurity, few studies propose modular, secure, and ready-to-deploy architectures. Furthermore, the constraints and regulations specific to small and medium accounting firms are often not taken into account. The literature mainly focuses on compliance frameworks (e.g., NIS2, GDPR) as well as maturity assessments or general best practices. This creates a gap, as these works do not provide ready-to-use, reproducible technical models aligned with European standards.

The study by Chidukwani et al. [13] also highlights this gap, stating that most SMEs don't know where to begin when it comes to securing their systems, and there is a lack of concrete examples or preconfigured solutions. Joswig et al. [54] also indicate that plug-and-play tools, compliant with regulations and easy to implement without expertise, are needed.

This absence of modular, secure, ready-to-use architectures is what hinders the implementation of the Security by Design principle in SMEs. Del-Real et al. [17] explain that even though the concept of *Security by Design* is widely referenced, it remains poorly defined, non-operational, and rarely applied in real infrastructures. Moreover, to meet the requirements of European regulations, it is essential to integrate security from the design phase of the infrastructure, including centralized identity management, access policies, rapid service recovery in case of incidents, etc. This can be achieved through IaC modules.

3.10 Literature Review

From what we've seen, several trends have emerged over the past decade. We present these in the *Trends* section, but despite this progress, the literature still shows limitations and opportunities which we summarize in *Limitations in Existing Work* and *Opportunities*.

3.10.1 Trends

We identified a few trends in SME cybersecurity:

- **Progressive and tailored cybersecurity frameworks:** Frameworks like CyFun or PUZZLE address the need for human support, providing more flexible and contextualized solutions as alternatives to rigid and complex standards like ISO/IEC 27001.
- **Increasing automation via DevSecOps:** The development of new security tools in CI/CD pipelines is becoming widespread. The Security as Code paradigm is gaining popularity, with solutions like Cyberismo [30] combining modularity and auditability, allowing compliance to be modeled as code.
- **Rise of IaC and modular infrastructure:** Tools like Terraform and Ansible now allow for the deployment of secure infrastructures at lower cost. They address SMEs' constant need for modularity and scalability.
- **Recognition of the human factor:** Recent studies emphasize the importance of employee awareness, often highlighting the need for contextualized training and cultural alignment to make cybersecurity a daily practice and increase cyber maturity [1, 13, 24, 51].

3.10.2 Limitations

Many of the solutions presented here unfortunately have drawbacks. To our knowledge, we identified several major limitations:

- **Lack of ready-to-use solutions:** Most of the analyzed papers describe theoretical models or infrastructures requiring complex configuration, making them unsuitable for SMEs without technical expertise [30].
- **Generic vs sector-specific:** Solutions like PUZZLE or Cyberismo are too generic and do not meet the needs of accounting firms, which handle sensitive data and lack the technical staff to manage, adapt, and configure such solutions.
- **Training not integrated:** Training or awareness programs are not integrated into infrastructure solutions and are often not automated, which is a real need for SMEs, especially accounting firms with non-technical staff [26].
- **Infrastructure with the CyFun framework:** To our knowledge, no study has shown the development of a modular infrastructure aligned with the CyFun framework, although the framework itself is recent.

3.10.3 Opportunities

Based on the limitations we presented, there are several opportunities for improvement in the field of modular infrastructure for small and medium accounting firms. The different opportunities and contributions are:

- Proposing a modular, secure, and automated infrastructure tailored to the specific constraints of SMEs in the accounting sector and aligned with CyFun requirements [54].
- Demonstrating the technical feasibility of a "Security by Design" approach based on IaC, using open-source tools and reproducible deployment.

- Adding ready-to-use and automated training modules into modular infrastructures to improve human resilience [24].
- Contributing to the improvement of DevSecOps in small structures by integrating best practices (automated testing, IaC, etc.).
- Documenting a reusable model that can serve as a base for Belgian SMEs like accounting firms.
- Involving SMEs in the design of frameworks and tools as recommended by [43], to ensure solutions are aligned with their real needs.

Summary

Small and medium-sized enterprises are a critical point in the cybersecurity world. They are highly exposed due to limited resources and a lack of awareness following rapid digitalization. This makes it difficult for them to adopt proper security practices despite their importance in the economy. In particular, accounting firms handle sensitive data and are subject to regulations, making them strategic targets for malicious actors since they manage many clients' data.

Current literature mainly focuses on risk identification using maturity models or awareness campaigns. Several frameworks (CyFun, PUZZLE, ASMAS) have been designed to support SMEs progressively. However, these frameworks do not directly offer concrete solutions like ready-to-deploy, secure infrastructures. This makes it difficult for companies to properly apply these frameworks due to the technical complexity, lack of automation, and lack of modularity in solution design. While traditional standards like ISO27001 and NIST are too complex and unsuitable, CyFun is perfectly suited for SMEs as a lighter and more adaptable framework.

DevSecOps and Infrastructure as Code approaches are becoming increasingly popular and promising. Tools like Ansible and Terraform automate infrastructure deployment and support compliance. However, adoption is still challenging for SMEs, again due to the lack of technical skills or security support budgets. Even when companies implement CI/CD pipelines, the critical security steps are rarely put in place. It is therefore important to understand that not only should these solutions be automated, but the essential security steps should be made concrete to provide a ready-to-use infrastructure.

The accounting sector is even more lacking in solutions. There are few, if any, architecture models that combine compliance and automation. The initiative that comes closest, Cyberismo, proposes "Security as Code" approaches to model compliance, but the tool requires advanced technical expertise. In addition, the human factor remains overlooked in secure infrastructure implementations and there is a lack of modular, adaptable training. Knowing that human error remains the main cause of incidents, there is a real need for less generic training and automated learning modules.

This observation reveals several opportunities for improvement, such as designing and building a modular infrastructure aligned with the CyFun framework and adapted to Belgian SMEs, particularly accounting firms.

Review methodology

This review aimed to gather relevant work on SME cybersecurity, modern development practices, compliance frameworks like CyFun, and tools to adapt the Janus infrastructure for Belgian accounting firms. Given the lack of studies directly addressing this specific combination, a broad yet systematic approach was used to ensure valid and comprehensive findings.

The search focused on peer-reviewed papers from 2015 onward to capture recent trends, as older sources risked outdated practices in fast-evolving fields like DevSecOps and compliance. Primary databases included ACM Digital Library, IEEE Xplore , and ProQuest [a], chosen for their high-quality, peer-reviewed content in computer science and cybersecurity. Google Scholar was used sparingly to cross-check gaps, prioritizing verified academic sources. Keywords were crafted to cover three areas: SME cybersecurity needs, modern practices (EaC, TDD, CI/CD), and compliance frameworks (CyFun, GDPR). Example queries included:

```
("SME cybersecurity" OR "small business security") AND ("compliance" OR "GDPR")
    AND ("Belgium" OR "European")
"DevSecOps" AND ("CI/CD" OR "test-driven development" OR "infrastructure as code
    ") AND ("security automation" OR "compliance")
("CyberFundamentals" OR "CyFun" OR "cybersecurity framework") AND ("SME" OR "
    accounting")
```

Listing 3.1: Search queries used for the literature review

These queries balanced specificity (e.g., CyFun, Belgium) with broader terms (e.g., SME, compliance) to catch related work, yielding around 80 initial papers. Zotero [b] was used to store, tag, and sort references by relevance, such as publication year or focus (SME, DevSecOps, compliance). Connected Papers [c] helped map relationships between papers, highlighting influential works and clusters, like SME security adoption or DevSecOps automation, ensuring key studies weren't missed. Non-peer-reviewed sources, like master's theses, were included only when they offered unique insights, such as practical tool implementations.

To ensure exhaustivity, the search iterated over refined queries when gaps appeared, such as few papers on CyFun for finance. Broader terms like "lightweight frameworks" or "financial data security" were then used. Relevance was judged by alignment with Janus's goals: modularity, compliance, and SME usability. Selected papers were analyzed for their methodology (e.g., case studies, experiments), findings (e.g., tool performance, compliance rates), and applicability to accounting firms.

For comparing approaches, metrics included:

- **Effectiveness**: How well tools or frameworks reduced risks (e.g., breach rates in SMEs)
- **Usability**: Ease of adoption for non-experts, critical for SMEs with limited staff.
- **Cost**: Affordability, as budget constraints are a key SME concern.
- **Compliance**: Alignment with GDPR or CyFun, vital for Belgian firms.

No experimental environment was built for this review, as it synthesizes literature rather than testing tools directly. However, Janus's setup (Vagrant, Ansible, Terraform,

Docker) was used as a reference to assess tools' fit. For example, papers testing Terraform with Checkov were prioritized for their relevance to Janus's IaC base.

[a] **ProQuest**: Advanced Technologies & Aerospace Database — `https://www.proquest.com`
[b] **Zotero**: free and open-source reference manager — `https://www.zotero.org`
[c] **Connected Papers**: graph-based tool to explore papers — `https://www.connectedpapers.com`

Chapter 4
Project's mission, objectives and requirements

The main objective of this master thesis is to adapt the JANUS project into a modular infrastructure that will be aligned with the level Basic of the CyFun cybersecurity framework, and specifically tailored to the needs of small and medium-sized enterprises in Belgium, more particularly for accounting firms. This would allow guaranteeing the basic level for NIS2 compliance. For this project, we will therefore establish a profile of the target company using scientific methodology. We will provide a ready-to-use, secure-by-design, and automated solution thanks to the IaC and CaC paradigms, allowing SMEs to apply this solution.

The project focuses on one CyFun function, the **Identify** function (Understand the main cyber threats to your most valuable assets) to allow a complete implementation rather than partially covering the different functions. This will ensure a strong implementation, which will be accompanied by documentation of the technical and organizational aspects.

4.1 Project Objectives

The main objectives of the project are:

- **Study and describe the needs:** small accounting firms have typical technical and organizational needs in Belgium, so we need to formalize them to understand the needs of the IT infrastructure.An in-depth study will define the infrastructure requirements.
- **Adapt and extend:** we will need to adapt the JANUS infrastructure and extend it for the needs of the accounting firms we studied.We will adapt to suit the production needs of SMEs particularly regarding persistence, documentation, automation, and compliance.
- **Fully cover "Identify" from the CyFun framework:** This includes inventorying assets, assigning roles and responsibilities, identifying risks, and generating audit-ready documentation. The process is entirely automated using IaC and Ansible-based playbooks
- **Ensure flexibility and reusability:** we must make sure that the infrastructure is not static, so that other SMEs can also use this infrastructure depending on their needs. And that the process to extend or reuse it is easy to access even with low technical knowledge.

4.2 Project Requirements

For the technical requirements we will need:

- **Ansible:** used for configuration management and application deployment. Ansible is used in this project to deploy the base configuration of machines, automate asset inventory, and generate technical documentation files.

- **Vagrant:** used to orchestrate the creation of reproducible virtualized environments. In the project, Vagrant allows quick deployment of several virtual machines representing different roles of the infrastructure.

- **Docker:** used to deploy containerized services, in addition to virtual machines. Containers make it easier to isolate specific components.

- **Alignment with CyFun Framework Level 1:** all technical actions in our project aim to meet the basic requirements of the national CyFun framework, especially for the Identify function. The infrastructure is designed to automatically generate deliverables for each sub-function of the framework.

- **VirtualBox:** our virtualization engine, compatible with Vagrant. It allows each component of the infrastructure to be isolated and emulates a production environment without depending on external resources.

- **Create organizational procedures:** in addition to the technical aspect, the project includes the automated creation of organizational documents needed to demonstrate compliance during an audit.

4.3 Documentation Requirements

One of the main goals of this project is to produce clear and complete documentation for SMEs. The implementation must be usable, especially for those that do not have advanced expertise in cybersecurity or system administration.

The documentation also supports compliance with the CyFun framework, especially for governance-related sub-functions.

We provide technical documentation useful for companies that want to deploy the infrastructure and become compliant, with step-by-step deployment instructions, without requiring advanced technical knowledge.

We also include organizational documentation, with internal procedure templates meant to be adapted by each company, as well as simplified governance diagrams aligned with CyFun recommendations.

Finally, we provide templates for security guidelines and policies, along with compliance checklists based on the Identify function requirements.

Technical Note

All documentation is produced in Markdown and YAML formats. This choice was made to make integration with automation tools easier and to allow future use of playbooks for automatically editing documents if needed.

4.4 Project Out-of-scope

This section defines the project boundaries to keep it realistic and achievable within the scope of a Master's thesis. The focus is on delivering a complete and well-documented implementation of the Identify function. The following elements are explicitly excluded from this project:

- **Full coverage of the CyFun framework:** The project does not cover all CyFun functions. Only the Identify function is fully implemented, with a clear structure, documentation, and deliverables. The other functions are mentioned as future work.
- **Large-scale testing in real SME environments:** Although the solution is designed for operational deployment, it has not been rolled out in multiple real-world SMEs. Testing is limited to a proof of concept, validated through simulations and feedback from interviews with accounting firms.
- **Obtaining official certification:** The project aims for alignment with Level 1 of the CyFun framework but does not aim to obtain an official certification from an external authority. However, the deliverables such as policies and procedures are designed to help prepare for pre-certification.

Chapter 5
Experimentation & data collection

5.1 Methodology

In this thesis, we chose a qualitative approach to validate the real needs of Belgian accounting firms. It was necessary to understand their infrastructure and cybersecurity needs. The main objective was to compare the assumptions derived from literature and document analysis with the reality in the field. This was done to identify gaps in maturity, awareness, and available resources that these firms face. This step is essential to ensure the proposed solution is relevant and useful.

To do this, we conducted a survey with five accounting firms. The choice of this method is based on several factors:

- **The will to model a representative profile based on real cases:** This profile is meant to be a reasonable representation of concrete cases. It is not fictional, but directly derived from real observations and then generalized.
- **The small sample size:** The sample was too small to produce statistical results, but it allowed us to deeply explore real-life situations. The feedback we received was rich in information.
- **The need to gather contextual information:** Unlike quantitative surveys, we could ask open-ended questions to better understand how these small businesses work, how they perceive cybersecurity, and the frameworks that surround them. The goal was to explore their actual practices.

The protocol we used aimed to validate the target profile, which serves as the foundation for designing a tailored infrastructure. This ensures that the proposed solution is not purely theoretical.

It is important to note that this methodology does not cover technical implementation aspects, which will be detailed in the next chapter (Chapter 6). This section is limited to validating the usage context and business needs. The objective is to establish a representative generic profile, which will guide the entire logic of the secure modular infrastructure design.

5.1.1 Validation of the Target Profile

To validate the profile and meet the real needs of small accounting businesses in Belgium, we carried out a qualitative survey on 5 accounting firms that fit the definition of small and medium-sized enterprises defined earlier 2.2. The goal was to validate what had been studied and to take into account the initial hypothesis regarding their infrastructure, cybersecurity measures, and compliance based on the real answers provided by the companies. It is important to note that the survey was only conducted on companies in Wallonia, and that the questionnaire included nine questions covering the following topics:

- **Management of their IT infrastructure and cybersecurity**: These questions were useful to understand if they have a person in charge of everything IT-related or if they call on external companies.
- **Software used for accounting and business operations**: This topic helps to understand what kind of software and tools they use, in order to better understand their sector and way of working.
- **Their essential tools to continue their activities without disruption**: This helped us understand their critical tools but also their dependencies.
- **Compliance with regulations**: This topic helps to understand which standards and regulations they must comply with, and also if they are aware of the standards and frameworks that exist to help them.
- **Their main cybersecurity concerns**: This last topic is important to understand their important "assets" and to see if they are aware of the risks and threats they could face.

All the answers were collected via written exchanges by email or phone. The data was anonymized to preserve the security and confidentiality of the companies.

5.2 Synthesis of the Results

Once the companies answered the questions, we observed the following trends:

- **IT Outsourcing**: 4 out of 5 firms fully outsource the management of their IT infrastructure to an external company. The only firm that doesn't is an exception in our case because it is not a dedicated IT or cybersecurity person, but an accountant who enjoys handling IT matters. This shows a strong dependency on outsourcing because companies do not have the budget to hire an IT team.
- **Software Used**: The firms mainly use the same software, starting with Sage BOB 50[a] and Sage BOB. Some companies also use Horus, another accounting software. They also use specific SaaS solutions for tax declarations like Intervat or Biztax.
- **Regulatory Compliance**: Accounting firms are often aware of GDPR and know they need to comply with it. However, even though they are aware, only 2 out of the 5 firms were able to describe the rules and processes they have put in place to comply. As for the knowledge of NIS2 or the Cyfun regulatory framework, none of the companies had heard of them.
- **Information Security**: Security practices vary greatly from one firm to another. Some companies mention applying two-factor authentication, regular data backups, or a strong password policy. Others have no specific protection and no data protection or network segmentation. This shows the need for a secure infrastructure by design to enforce good practices and comply with standards.
- **Cybersecurity Audits / Risk Assessment**: No company had performed a security audit for their organization. This evaluation shows that small accounting firms still lack the maturity or vision necessary to understand their vulnerabilities and to prioritize cybersecurity tasks after risk assessments.

- **Perceived Risks and Threats**: The firms interviewed often share the same cybersecurity fears. The number one risk (mentioned by 100% of firms) is the loss of data or the temporary or permanent blocking of data. This risk is often linked to ransomware attacks but also to potential technical failures or human errors. The second most cited risk is phishing: only 2 companies mentioned being afraid that one of their employees might click on a malicious link. This shows that the managers believe the phishing risk mainly concerns employees and not themselves. These risks match what has been seen in the literature, where companies are often not fully aware of the real cybersecurity risks they face. This observation aligns with the findings of Alahmari et al. [1] who emphasize that SMEs often underestimate the risks associated with cyber threats such as ransomware and phishing. Their study highlights a recurring lack of awareness and preparation among small businesses, despite the increasing prevalence and sophistication of these attacks.

- **Incident Response Plan and Employee Awareness**: No firm had implemented an incident response plan. Regarding awareness training, only one company had done an informal awareness session, explaining to employees the risks in cybersecurity and warning them to be careful. This situation exposes the firms to greater risks in case of a cyberattack.

All the raw data collected during the survey can be found in Appendix A.1 - Summary of responses.

[a] **Sage BOB**: Accounting software — `https://www.sage.com/fr-be/produits/sage-bob/`

Conclusion

The analysis of the answers provided by the Belgian accounting firms shows a strong dependency on IT outsourcing. They are aware of GDPR, but the concrete implementation of it is still incomplete. The interviewed firms have no knowledge of the NIS2 directive and are unaware of the Cyfun framework. The lack of cybersecurity awareness and incident response planning confirms that companies are vulnerable to cyber threats.

These results fully justify the design of a modular secure infrastructure aligned with the Cyfun framework.

5.3 Generic Profile

Following the results of the survey conducted with five accounting firms, we designed a representative profile of a small accounting SME in Belgium. This generic profile aims to help us later as a design base for the infrastructure and will guide us during the development.

In the typical profile (Figure 5.1), the company only has accountants and a secretary as employees. So there is no IT or cybersecurity manager. The computers are managed by an external company (which manages their cloud, Microsoft 365, etc.). They have a Wi-Fi connection used through Ethernet or wireless. The accounting software is available as regular software or as Software as a Service.

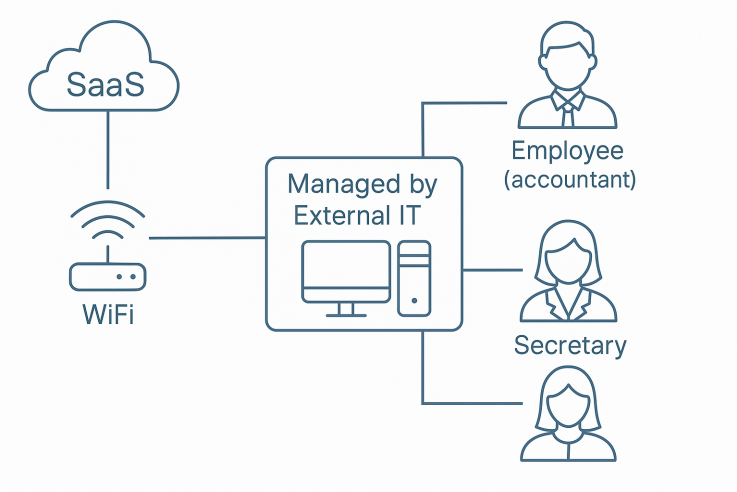

Figure 5.1: Schematic IT view of a typical Belgian accounting firm

This shows us that this profile has no internal IT department, with a strong dependency on external companies for all the technical aspects of the business. But these companies don't handle compliance or cybersecurity risks. So no real controls are in place to protect against cyber threats.

5.4 Setup, Requirements, Environment, Tools & Materials

In this research, the objective was to design a representative profile of Belgian accounting firms, taking into account their real constraints regarding cybersecurity and IT infrastructure. To define and validate this generic profile, it was essential to implement a clear

protocol to ensure the validity and traceability of qualitative data collection. This section aims to describe the resources used, the tools employed, and the conditions in which the questionnaire was distributed.

5.4.1 Questionnaire Preparation

To determine which questions were important, we based them on the document analysis presented in the previous chapters. This gave us a theoretical profile that we could then try to validate. The questionnaire included 9 open-ended questions and was written in French, the native language, to make it easier for the targeted respondents and to allow them to express their answers freely. The questions were phrased in a non-technical way to suit a non-expert audience. The tone was designed to create a trusting environment and to encourage honest, judgment-free responses, especially so participants would not fear revealing non-compliance with regulations that may apply.

The questionnaires were completed exclusively by email and phone, to allow personalized follow-up and clarification when needed.

5.4.2 Company Selection

Five accounting firms were selected in the Walloon region. The choice was guided by ease of contact and the small size of the firms. These companies do not have internal IT departments. The selection followed a purposive sampling strategy. The goal was not statistical representativeness but the collection of rich and contextual qualitative information.

The companies were contacted through professional networks or recommendations. All of them agreed to answer the questionnaire voluntarily, with the assurance that the collected data would be anonymized and used only for academic purposes.

5.4.3 Response Modalities

Responses were collected over a two-week period, from March 18 to March 29, 2025. Two firms responded by email, sending back the completed questionnaire in text format. Three others were interviewed by phone, with immediate note-taking and later written confirmation of the collected statements. No audio recordings were made. The average duration of a call was about 30 minutes.

5.4.4 Analysis Tools and Environment

The analysis of the results was done using a table that allowed cross-referencing the information. A matrix was built for each theme of the questionnaire, with one row per company and one column per analysis criterion. Categories were created to make it easier to code the responses.

Chapter 6
Experiment's output/Data Analysis

6.1 Implementation of CyFun Identify Function

In this section, we introduce the core of the thesis, which is the implementation of the Identify function from the CyFun framework level basic, inside a modular infrastructure by automating as much as possible to help small Belgian companies, especially accounting firms, to be compliant with CyFun.

The decision to focus on the Identify function aims to build a structured base for a cybersecurity approach. Before trying to defend a company, it is essential to know its assets and risks. That's why this function is the most important one to reach the Basic level of compliance in the CyFun framework.

This work is based on the business needs identified in Chapter 4, which show a lack of visibility on assets, legal rules they must apply, and their critical dependencies. The goal is to automate as much as possible using Infrastructure as Code and Configuration as Code, while also defining organizational processes to be compliant with different regulations (CyFun, GDPR, and NIS2).

Approach

In Chapter 4, we highlighted the typical IT and business needs of small accounting firms in Belgium. These findings helped us improve our implementation. Each sub-function of the CyFun Identify category is addressed with maximum automation while keeping everything as modular as possible (with Ansible playbooks) and supported by procedures.
This chapter explains the implementation of each sub-function with a structured and reproducible approach.

The CyFun framework defines three maturity levels for cybersecurity practices: *Basic*, *Important*, and *Essential*. This thesis focuses specifically on achieving the **Basic** level for the **Identify** function, which serves as the foundation for any cybersecurity posture. Identifying assets, responsibilities, risks, and applicable rules is a prerequisite for managing and mitigating threats effectively.

To meet the **Basic** level of CyFun compliance, the Identify function is divided into three categories, each composed of specific sub-functions:

- **Asset Management (ID.AM)**
  - **ID.AM-1** : Inventory of physical devices and systems
  - **ID.AM-2** : Inventory of software platforms and applications
  - **ID.AM-3** : Identification of organizational information and communication flows
  - **ID.AM-4** : Cataloguing of external systems

  - **ID.AM-5** : Prioritization of resources based on classification, criticality, and business value

- **Governance (ID.GV)**
  - **ID.GV-1** : Establishment and communication of an organizational cybersecurity policy
  - **ID.GV-3** : Understanding and management of legal and regulatory cybersecurity requirements
  - **ID.GV-4** : Integration of cybersecurity risk management into governance processes

- **Risk Assessment (ID.RA)**
  - **ID.RA-1** : Identification and documentation of asset vulnerabilities
  - **ID.RA-5** : Risk assessment based on threats, vulnerabilities, likelihood, and impact

Each of the following sub-sections deals with a specific sub-function. For each, we specify the objective as defined in the CyFun framework, then describe the technical means implemented to meet it, mainly based on automation via Ansible playbooks and deployment scripts. We then detail the elements generated by this implementation, and their role in demonstrating compliance. Finally, we present the organisational aspects required to support the application of the measure over the long term, including the roles involved, the internal procedures and the documents produced.

This structured approach ensures traceability, reproducibility and objective assessment of compliance with the Basic level requirements for the Identify function.

ID.AM-1 – Inventory of Physical Assets

The ID.AM-1 sub-function requires the organization to maintain an up-to-date inventory of its physical assets. This inventory must include all assets, whether they are connected to the organization's network or not. That means assets like servers, workstations, printers, etc. This inventory is really important because it forms the foundation of a good cybersecurity strategy. It helps to identify the resources the company needs to protect. Without knowing its assets, it's hard for a company to protect itself properly. In our case, SMEs, especially small accounting firms, must ensure they know their assets and are able to identify them in order to meet the basic level of CyFun compliance.

The main objective is to automate this process as much as possible to reduce human intervention. As we saw during the interviews with companies, they don't have the resources to hire IT staff to handle compliance aspects. That's why we implemented different scripts to reduce the technical burden on SMEs, who often don't have technical skills. At the same time, we ensure that the inventory is reproducible and modular.

Implementation

The JANUS infrastructure was extended to automate the collection of physical assets. We created an Ansible playbook called `collect_physical.yml` to follow an Infrastructure as Code approach.

```yaml
---
- name: Collect system facts and write to asset inventory
  hosts: user_domain
  gather_facts: yes
  remote_user: "{{ user_var }}"
  connection: local

  vars_files:
    - ../../../user/local_or_remote.yml

  tasks:
    - name: Set asset facts
      set_fact:
        asset_info:
          asset_type: infrastructure
          description: "Auto-collected infrastructure asset for CyFun ID.AM-1 compliance"
          collected_on: "{{ ansible_date_time.date }}"
          assets:
            - id: "{{ ansible_hostname }}"
              name: "{{ ansible_hostname }}"
              type: "vm"
              os: "{{ ansible_distribution }} {{ ansible_distribution_version }}"
              ip: "{{ ansible_default_ipv4.address }}"
              cpu: "{{ ansible_processor_cores }} cores"
              ram: "{{ ansible_memtotal_mb }} MB"

    - name: Write infrastructure asset inventory to file
      copy:
        content: "{{ asset_info | to_nice_yaml(indent=2) }}"
        dest: "/janus/domain/cyfun/identify/output/assets.yml"

    - name: Print confirmation message
      debug:
        msg: " collect-assets.yml executed successfully for CyFun ID.AM-1"
```

It allows automatic collection of physical asset data from the JANUS infrastructure (using Ansible's `gather_facts` module), such as hostname, operating system, IP address, etc., without any manual intervention. The playbook is automatically executed by Vagrant during each deployment or rebuild of a virtual machine. This ensures continuous compliance, even if we change the VM's allocated size, after deployment, it will be automatically registered with updated information.

Once collected, the inventory is generated as a YAML file named `assets.yml`, stored at the following location:

`/janus/domain/cyfun/identify/output/assets.yml`

Even though the goal is to automate as much as possible, some assets in a company cannot be detected automatically. The goal is also to keep everything as modular as possible. So we adapted the solution by adding a manual file called `manual-assets.yml` (Appendix B.1) , located in the same folder as `assets.yml`. It allows us to manually

add the company's physical equipment that cannot be collected automatically (printers, routers, etc.). This file includes mandatory fields to comply with the framework (ID, name, type, owner, location, last review date) and optional fields like operating system or serial number, to provide complete documentation.

```yaml
asset_type: infrastructure
assets:
  - cpu: 4 cores
    id: vagrant
    ip: 10.0.2.15
    name: vagrant
    os: Debian 12.6
    ram: 3914 MB
    type: vm
collected_on: '2025-05-18'
description: Auto-collected infrastructure asset for CyFun ID.AM-1 compliance
```

Compliance Demonstration – CyFun ID.AM-1 Basic Level

The Basic level of the ID.AM-1 sub-function requires the organization to maintain an up-to-date inventory of its physical devices, whether they are connected to the network or not. This inventory must cover all equipment used to process or store information, including virtual machines, workstations, printers, or network components.

In our case, this objective is achieved in a pragmatic way, adapted to the limited resources of small businesses. The collection of deployed machines in the JANUS infrastructure is automated using an Ansible playbook. This automation allows us to systematically identify virtual machines and extract key technical details at each deployment. The result is stored in an automatically generated inventory file, which ensures minimum traceability without human intervention.

For equipment that cannot be detected automatically, a manual template file is provided. This file follows the structure required by the CyFun framework and allows organizations to complete the necessary information themselves in order to remain compliant, even without advanced monitoring tools. It serves as a basic and adaptable documentation method, not covering the full asset lifecycle but sufficient to meet the expectations of the Basic level.

Although this implementation is not a complete asset management system, it allows us to meet the requirements of ID.AM-1. It offers an automated starting point, complemented by manual input, suitable for SME constraints and sufficient to demonstrate compliance with the required level.

ID.AM-2 – Inventory of Software Platforms and Applications

The ID.AM-2 sub-function requires a complete inventory of the software platforms and applications used within the organization. This includes local applications but also remote services like SaaS. This sub-function complements the asset inventory to help identify potential vulnerabilities and ensure compliance. For Belgian accounting firms, this task is particularly important because they use specialized software like Sage BOB, Horus, Intervat, etc., as SaaS services, and these tools handle sensitive data.

Implementation

The implementation we aimed for to be compliant with ID.AM-2, while automating the information collection, is to automate the collection of installed software and active Docker services. This allows us to only collect services that are actually running and used by the company. With this method, we give the company flexibility while staying compliant with CyFun.

The script `collect_software.yml` is used to collect installed and active software. (Appendix B.2)

It runs Ansible via Vagrant during the infrastructure startup. It collects running services via `docker ps` and retrieves installed packages via `dpkg` on Debian systems.

Once the software and packages are collected from the infrastructure, all results are merged into a single structured YAML file:

`/janus/domain/cyfun/identify/software/output/assets_software.yml`

```yaml
- collected_on: '2025-05-24'
  host: vagrant
  inventory_type: software
  os: Debian 12.6
  software:
    docker_services:
    - name: logseq
      versions:
      - ghcr.io/logseq/logseq-webapp:latest
    - name: drawio
      versions:
      - jgraph/drawio
    - name: codimd-codimd
      versions:
      - hackmdio/hackmd:2.4.2
    - name: codimd
      versions:
      - postgres:11.6-alpine
    - name: syncthing
      versions:
      - syncthing/syncthing
    - name: nextcloud
      versions:
      - nextcloud
      - mariadb:10.6
    - name: jitsi
      versions:
      - jitsi/jicofo:stable-9220-1
      - jitsi/jvb:stable-9220-1
      - jitsi/prosody:stable-9220-1
```

A complementary manual file, `assets_software_manual.yml`, is also planned to document platforms that cannot be detected automatically, such as SaaS or external services. (Appendix B.3)

Compliance Demonstration – CyFun ID.AM-2 Basic Level

The sub-function ID.AM-2 requires the organisation to maintain an inventory of the software platforms and applications it uses. This inventory should cover both locally installed software and remote services, including Saas solutions. The objective is to provide enough visibility to identify critical dependencies and reduce risks related to outdated, unauthorized, or unsupported software. In our case, the JANUS infrastructure allows the automation of the collection of installed software as well as running services through Docker. This approach, even if incomplete compared to a full configuration management system, provides a consistent foundation to meet the expectations of the Basic level. It ensures that all services actually used on the infrastructure are listed in a central file, updated at each redeployment. In fact, if a service is not enabled or removed from the Vagrant configuration, it will no longer be activated and therefore will not be detected. Software that cannot be detected automatically, especially external services or SaaS tools used in accounting firms like SAGE, is not collected automatically. However, we created a manual file template that allows the organization to document them. This file completes the automatically collected data and follows the structure expected by the CyFun framework.

With this combined approach, using automation for local components and a manual model for external platforms, we meet the essential requirements of the ID.AM-2 sub-function at the Basic level. The organization has a documented and updateable inventory that covers the critical elements of its software environment.

ID.AM-3 – Mapping Organizational Communications and Information Flows

The ID.AM-3 sub-function asks the organization to map the types of information it uses, and to list all the types of information the organization uses or stores. For an accounting firm, this helps better understand the sensitive data they handle and also ensure the confidentiality of this data. This will then allow them to classify information and be compliant with GDPR.

The implementation aims to identify and categorize the critical types of information processed by accounting firms. It also aims to document data flows, storage locations, transmission channels, and levels of criticality. The goal is to generate a structured artifact in YAML format, to make it easier to use in the next tasks and sub-functions of the CyFun framework.

Implementation

For the implementation of ID.AM-3, unlike sub-functions ID.AM-1 and ID.AM-2 where automation with Ansible was used, the implementation here is based on a manual approach, due to the difficulties we encountered while trying to automate this type of data collection.

Several limitations caused problems. First, the data comes from various sources (physical meetings, phone calls, etc.), often non-digital, unstructured, and not traceable by automated tools. Then, the exchange channels, as explained before, are not easily detectable by scripts. Gathering information from a physical meeting while respecting different regulations takes a lot of time and resources to find a suitable solution. Finally, the complexity of exchanges like verbal communication cannot be handled with automated analysis.

Faced with these constraints, a manual YAML template called `assets_information.yml` was created. It is based on the generic profile we established for Belgian accounting firms in Section 5.1. This profile was validated during a qualitative survey with five accounting

firms. The YAML template was designed to be intuitive and adaptable, allowing non-technical users to properly document data flows without specific cybersecurity knowledge. `TO_BE_FILLED` markers are included in the template to indicate where the company should edit or add information to be compliant with this sub-function.

```yaml
# CyFun ID.AM-3 compliance - Information types used and simplified data flow mapping

collected_on: TO_BE_FILLED
collected_by: TO_BE_FILLED

information_types:

  - id: INF-001
    name: Client Data
    description: Client identification information (name, address, VAT number, email, phone)
    category: personal / accounting
    source: TO_BE_FILLED (e.g. web form, client email, in-person meeting)
    stored_in:
      - Nextcloud (if active)
      - Bitwarden (if client access credentials stored)
      - TO_BE_FILLED (e.g. BOB SaaS)
    accessed_by:
      - accountant
      - secretary
    exchanged_via:
      - email
      - HTTPS
      - physical meeting
    criticality: high

  - id: INF-002
    name: Accounting Documents
    description: Invoices, bank statements, expense reports, VAT declarations
    category: financial documents
    source: TO_BE_FILLED (e.g. client upload, bank export)
    stored_in:
      - Nextcloud (if active)
      - TO_BE_FILLED (e.g. accounting SaaS)
    accessed_by:
      - accountant
    exchanged_via:
      - HTTPS
      - secure email
    criticality: high
```

Important remark

ID.AM-4 – Catalogue of External Information Systems

This sub-function recommends listing the external information systems the organization relies on. Even though no requirement is imposed at the Basic assurance level, we followed this recommendation in the sub-functions ID.AM-1 and ID.AM-2.

Compliance Demonstration – CyFun ID.AM-3 Basic Level

The sub-function of the ID.AM-3 at the basic level requires that the organization have to list the types of information it processes or stores, and to document the main associated information flows. This includes identifying the sources, storage locations, transmission channels, and users involved. The objective is to give a clear view of the information handled, its sensitivity, and its potential exposure.

In our case, due to the unstructured nature of information flows in small companies, especially in accounting firms, full automation was not possible. The solution adopted is therefore based on a manual model, designed to be used without technical skills. This model is based on the generic profile we created using the feedback we collected, and it allows organizations to map critical information types such as client data or accounting documents. The `assets_information.yml` file we provide allows a company to fill in the required fields according to its own structure. It offers a standardized format, while keeping the flexibility needed for local adaptation. Pre-filled fields and `TO_BE_FILLED` markers guide the user in the documentation process.

Even if this approach remains simply a declaration of information, it still meets the Basic level expectations by providing formal, reusable, structured documentation that fits the reality of SMEs. It is a concrete first step toward more mature information flow management.

ID.AM-5 – Resource Prioritization Based on Classification, Criticality, and Operational Value

This sub-function aims to organize the organization's resources, such as physical assets and software, by order of priority. Based on their classification, but also on their criticality and operational value (importance level for business processes). This prioritization helps focus cybersecurity efforts on the most sensitive resources, such as access credentials, client data, or accounting software for small accounting firms. By following this sub-function, companies will be able to optimize their resources and prioritize their efforts to comply with regulations. It also helps strengthen their resilience.

To meet this requirement, the first step was to centralize the inventories we had already collected earlier (physical assets, software, types of information) into a single file called `inventory.yml`. This file will be used to prioritize the inventory and can also help the company understand its full set of assets.

To centralize the inventories, we created the playbook `collect_all_inventory.yml` (Appendix B.4) , which merges data from the following files:

- `assets.yml` and `manual-assets.yml`: inventory of all physical assets in the company
- `assets_software.yml` and `manual_software.yml`: inventory of all software in the company
- `assets_information.yml`: types of information used and data flows

This playbook uses the Ansible module `slurp` [3] to read these files and parse their content. Note that we filtered out invalid entries, such as data still containing `TO_BE_FILLED`, to exclude manual inventories that haven't been completed yet. The cleaned data is then aggregated into the file `inventory.yml`

```yaml
assets_manual: []
collected_on: '2025-05-24'
host: vagrant
information_types:
- accessed_by:
  - accountant
  - secretary
  category: personal / accounting
  criticality: high
  description: Client identification information (name, address, VAT number, email,
    phone)
  exchanged_via:
  - email
  - HTTPS
  - physical meeting
  id: INF-001
  name: Client Data
  source: TO_BE_FILLED (e.g. web form, client email, in-person meeting)
  stored_in:
  - Nextcloud (if active)
  - Bitwarden (if client access credentials stored)
  - TO_BE_FILLED (e.g. BOB SaaS)
- accessed_by:
  - accountant
  category: financial documents
  criticality: high
  description: Invoices, bank statements, expense reports, VAT declarations
  exchanged_via:
  - HTTPS
  - secure email
  id: INF-002
  name: Accounting Documents
  source: TO_BE_FILLED (e.g. client upload, bank export)
  stored_in:
  - Nextcloud (if active)
  - TO_BE_FILLED (e.g. accounting SaaS)
- accessed_by:
  - accountant
  - secretary
  category: credentials
  criticality: critical
  description: SaaS logins, email access
  exchanged_via:
  - HTTPS
```

Once the inventory file is centralized, we need to prioritize the resources. To help automate this, we created a playbook `prioritize_inventory.yml` (Appendix B.6), which

applies a prioritization matrix defined in `prioritization_matrix.yml` (Appendix B.5).

About the prioritization matrix

The matrix is based on the generic profile of accounting firms, identifying critical resources and assigning priorities to them. However, this matrix can be adapted to the specific needs of any company.

This matrix includes predefined rules to assign each resource a classification, criticality level, and operational value. When the playbooks are run, they generate the final file `prioritized_inventory.yml`.

The following output shows the result of the prioritization matrix when only `jitsi` services are running:

```yaml
- classification: critical
  criticality: critical
  name: Credentials and Access Information
  value: very_high

- classification: confidential
  criticality: high
  name: Client Data
  value: very_high

- classification: confidential
  criticality: high
  name: Accounting Documents
  value: high

- classification: confidential
  criticality: medium
  name: Client Communications
  value: moderate

- classification: confidential
  criticality: medium
  name: Internal HR Data
  value: moderate

- classification: internal
  criticality: low
  name: jitsi
  value: low

- classification: internal
  criticality: low
  name: System Logs
  value: low
```

which is ordered logically by sensitivity. That means the most sensitive resources, such as Bitwarden or access credentials, appear at the top of the list, while public or support tools appear at the bottom. This file can help companies prioritize security updates and

patches, and identify the resources to include in a business continuity and organizational plan.

This approach is designed to be as reproducible, coherent, and modular as possible. It also allows accounting firms to adapt the matrix to their specific needs by modifying the file `prioritization_matrix.yml`.

Compliance Demonstration – CyFun ID.AM-5 Basic Level

The Basic level of the ID.AM-5 sub-function requires the organization to prioritize its resources according to their classification, criticality, and operational value. This prioritization helps to guide protection efforts on the most sensitive or essential elements of the company's information system.

In our case, the solution focuses on centralizing previously collected inventories into a single file, then applying a prioritization logic to that file. The result is not a dynamic prioritization tool, but a simplified and reusable method adapted to small organizations. It helps them identify their most important assets and document them in a structured and consistent way.

The file `prioritized_inventory.yml` is generated after merging existing inventories and applying a matrix that assigns priority levels. This matrix is based on a generic profile built for accounting firms, but it can be adapted by editing a simple YAML file. The output is ordered by sensitivity and operational value, so that the most critical elements appear at the top of the list.

Even if the method does not rely on real-time analysis or advanced algorithms, it provides a clear and usable view of the organization's key resources. It is aligned with the expectations of the CyFun framework at the Basic level, by offering a simple way to document and prioritize resources. This approach can serve as a first step toward more advanced risk and asset management processes.

Compliance Support: Checklists for ID.AM-* Sub-Functions

To ensure that the company meets the requirements of ID.AM-1 to ID.AM-5 sub-functions of the CyFun framework at the Basic level and to help them verify their compliance, we created a set of operational checklists in the directory

`/janus/domain/cyfun/identify/checklist/`

Each checklist is named according to the sub-function, for example `checklist-ID.AM-1.md`, `checklist-ID.AM-2.md`, etc.

These checklists provide a structured list of control points to check and mark. These documents are not mandatory to be aligned with the CyFun framework, but are meant to serve as practical guides for the company to make sure nothing was forgotten during asset management implementation.

The checklists were designed using references from Cyfun for asset and information classification management and `CIS Controls v8` [14] for hardware and software inventory.

Each checklist is structured to include:

- The minimum requirements to reach basic compliance (in our case, the basic level).
- Optional recommendations to improve the company's maturity.

- An audit field to indicate the status (`[x]` for validated, `[ ]` for not validated), which helps both the person implementing and the person verifying to clearly see whether all checkpoints have been validated.

This checklist-based approach allows the company to:

- **Verify compliance**: Each control point is directly linked to CyFun requirements.
- **Simplify adoption**: The checklists are actionable, with clear and easy-to-understand instructions, even for people outside of IT.
- **Anticipate gaps**: Unchecked points are clearly visible since we can see the status of the checkpoints.

Important

Why use checklists? Because they offer a practical and structured way to validate company compliance with a simplicity that is especially useful when small businesses have limited resources. This reduces the risk of forgetting steps and simplifies documentation in case of an audit.

Cybersecurity Governance: Implementation of ID.GV-1, ID.GV-3, and ID.GV-4 Sub-Functions

Cybersecurity governance is a strategic point that allows an organization to structure its efforts to protect its assets. As described in sub-functions ID.GV-1, ID.GV-3, and ID.GV-4, the goal is to establish and monitor the strategy, expectations, and risk management policy. This section shows the analysis and implementation of these sub-functions through policy and procedure documents stored in `/janus/domain/cyfun/identify/policies`, evaluating their compliance with CyFun requirements and their limitations in our environment.

It is important to understand that cyber risks have become very strategic and companies can no longer rely only on operational responses. They must also implement and adopt information security policies to be able to respond to legal and financial challenges.

ID.GV-1: Establishment and Communication of Cybersecurity Policies

For this first sub-function, we had to establish a cybersecurity policy defining the rules regarding information security. It takes into account several aspects, including the use of resources, access control, data classification, etc.

The document `cybersecurity-policy.md`(Appendix C.1) contains the different policies we considered essential to comply with regulations.

In addition, `cybersecurity-procedures.md` (Appendix C.2) details the processes for creation, approval, communication, and update. The full document lifecycle is respected.

These documents are aligned with recognized standards such as ISO/IEC 27002:2022 and CIS Controls v8, which require documented policies. The procedures include how policies are communicated, initial training, and annual review.

The policy is adapted for small accounting firms with clear processes for communication and training, aligned with CyFun requirements. However, the lack of automation to track acknowledgments exposes the company to risks of non-compliance.

Compliance Demonstration – CyFun ID.GV-1 Basic Level

The Basic level of the ID.GV-1 sub-function requires the organization to establish a formal, documented cybersecurity policy adapted to its needs. This policy must define the essential rules related to information security and be clearly communicated to the relevant parties.

In our case, a cybersecurity policy was written as a structured document. It covers the expected elements such as access management, resource usage, data classification, and user responsibilities. It is accompanied by a procedure document that explains how the policy is approved, shared, updated, and reviewed over time.

This approach allows us to meet the core expectations of the Basic level. Even if the process is not automated, the documents are complete and accessible enough to initiate governance suitable for a small structure. Initial training and yearly review processes are also described. In this case, we did not find it necessary to automate anything.

The main limitation identified is the absence of an automatic mechanism to track who has read or accepted the rules. Such a mechanism would have allowed us to verify who still needs to read the policies and who already did. However, this does not prevent compliance with the Basic level criteria, which do not require this level of formalization. The company now has a clear policy to improve its security posture and to formalize its governance practices.

ID.GV-3: Management of Legal and Regulatory Requirements

To comply with ID.GV-3, we created the document `legal_compliance.md` (Appendix C.3), which addresses legal and regulatory obligations, focusing on GDPR, Belgian accounting laws, confidentiality rules, and the NIS2 directive.

This document helps better understand the needs and regulations for small accounting firms. To respond to the various regulations, we implemented the following requirements:

- **GDPR:** Data protection through strong passwords, encrypted backups, and a reporting process within 72 hours.
- **Accounting law:** Accounting firms must keep client documents for 7 years.
- **NIS2:** Small accounting firms are not subject to NIS2, but if a firm expands its services and becomes a supplier to a NIS2 entity or handles data that presents a risk to the client in case of an incident, it will have to comply contractually with NIS2.

This document is provided to all staff during onboarding and explains legal requirements in the form of operational and measurable actions, with SMART objectives.

Compliance Demonstration – CyFun ID.GV-3 Basic Level

The ID.GV-3 sub-function expects the organization to identify, understand, and manage its legal and regulatory obligations related to cybersecurity. This includes data protection requirements, sector-specific regulations, and any contractual or subcontracting-related obligations.

In our case, these requirements are covered in a dedicated document called

`legal_compliance.md`. This document lists the main rules to follow for Belgian accounting firms, especially regarding GDPR, local accounting laws, and the NIS2 directive (which is not applicable to most SMEs). It is not a complete legal audit, but a practical

document designed to help understand the applicable obligations and propose concrete and realistic actions. It also helps companies better understand what they are expected to follow. As shown in the feedback from accounting firms, some were not always aware of the regulations they need to comply with.

The actions described are simple and directly applicable. For example, regarding GDPR, the policy includes the use of strong passwords, encrypted backups, and a 72-hour incident notification process. Concerning accounting law, it reminds that client documents must be stored for at least seven years. For NIS2, the document explains that small accounting firms are not directly affected, but may have to follow some contractual obligations if they work as subcontractors for entities that are in scope.

The document is given to each employee during onboarding and presents the obligations using concrete actions with SMART objectives. Even if it is not automated, this simple and understandable format helps small structures meet the expectations of the CyFun Basic level regarding legal and regulatory compliance.

ID.GV-4: Management of Cybersecurity Risks

For this last governance sub-function, the document `risk_strategy.md` (Appendix C.4) defines our high-level cybersecurity risk management strategy. It integrates risk management into company governance. The complementary file`risk_management_plan.md` (Appendix C.5) defines a structured process to identify, assess, and treat cybersecurity risks. We made it as accessible as possible.

It includes a risk methodology using a matrix based on impact and probability (scale 1 to 3), producing a score from 1 to 9, calculated as impact $\times$ probability. We chose this format because it is simple to understand and easy to use, even for people without technical skills.

Then, we provided a table template listing different risks and their controls, along with possible treatment plans. The review must be done annually.

Each role in the organization is involved. Roles are defined based on the generic profile (manager, accountant, secretary, IT provider), which strengthens the governance approach.

The goal here is to promote a risk culture, help the company better understand the risks it may face, and think about how to manage the most dangerous ones.

It is by formalizing responsibilities and making decisions visible that this risk management approach best supports low-maturity companies.

Compliance Demonstration – CyFun ID.GV-4 Basic Level

The ID.GV-4 sub-function expects the organization to integrate cybersecurity risk management strategy into its governance processes. The aim is to explain identify threats, assess risks, and define actions to treat them, while considering the context and available resources.

In our case, a risk management plan was written as a simple and structured document. It includes a method based on a matrix that combines impact and probability, to help prioritize risks in a simple way for companies with low maturity levels.

A table template is provided, allowing the organization to list its risks, existing controls, and planned actions to reduce them. The plan recommends an annual review, which matches the expected practices for a small structure. We believe that adding automation such as notifications or emails would add value, by helping companies make sure this review

actually happens every year. This is not implemented yet, but we suggest it as a future improvement.

The roles involved are based on the generic profile studied in accounting firms. This role distribution strengthens governance and prevents risk management from depending on a single person. It also helps companies better understand their risk exposure and see their limitations. We believe that if the risk process were fully automated, companies might feel less concerned and involved than when they do it manually.

Even if this approach does not include advanced tools or automated tracking, it provides a clear, understandable, and adapted base to meet the expectations of the Basic level of ID.GV-4. It encourages better understanding of risks and makes responsibilities visible.

ID.RA-1: Asset Vulnerabilities Are Identified and Documented

This Identify sub-function is one of the most difficult to implement automatically. However, it is a critical step in risk management for a company in terms of cybersecurity. Small accounting firms must understand the vulnerabilities in their assets to better understand the threats they face.

This sub-function aims to identify and document the vulnerabilities of assets, considering potential threats and associated risks. Before describing the identification and documentation process, we first define the core concepts of vulnerabilities, threats, and risks.

Core Concepts: Vulnerabilities, Threats, and Risks

- **Vulnerability:** A vulnerability is a weakness or flaw in an organization's hardware, software, network configuration, or organizational procedures [38]. It represents a potential entry point through which a malicious actor can compromise the organization's assets, such as the client databases of an accounting firm or its tax management software.

- **Threat:** A threat is a malicious or undesirable event that exploits a vulnerability to cause harm. Threats can be intentional (e.g., ransomware attacks, phishing targeting employees) or accidental (e.g., human errors, hardware failures) [40]. In our context, common threats include ransomware.

- **Risk:** Risk is the potential for loss or damage when a threat exploits a vulnerability. It combines the likelihood that a threat will occur and the potential impact on the organization [41]. For example, a vulnerability in unpatched accounting software, combined with the threat of ransomware, results in a risk of financial loss, GDPR violations, or damage to the reputation of an accounting firm.

Vulnerability Identification and Documentation Process

The management of sub-function `ID.RA-1` is based on a multi-step process. First, we must use the inventory to identify vulnerabilities, then assess the related threats, and finally document the results. To help SMEs with this process, we need to automate and simplify the approach as much as possible.

Once the assets are identified, we implemented a fully automated playbook to collect known vulnerabilities: `collect_vulnerabilities.yml` (Appendix C.6). It uses an open-source vulnerability scanner called **Trivy**, developed by Aqua Security. Trivy is our main

component to support this sub-function, which requires the identification and documentation of vulnerabilities on digital assets.

Trivy fits our needs well. Its main features include:

- **Docker image scanning:** Trivy can analyze local or remote images and extract all included libraries and components.
- **Known CVE detection:** It includes a constantly updated vulnerability database to detect public security issues.
- **Severity filtering:** We can target only high and critical vulnerabilities.

Another useful feature is that Trivy outputs results in `JSON`, which is directly usable in our reporting automation.

In our context, Trivy runs inside a container and is executed automatically via an Ansible playbook. It scans all active Docker images on our machines and therefore all running software like Mattermost, Bitwarden, etc. The scanner then lists all known vulnerabilities, including their IDs, severity levels, and related CWEs.

The `collect_vulnerabilities.yml` playbook automates the identification and documentation of vulnerabilities in Docker containers. These containers are a key part of the JANUS infrastructure, since all software runs inside them. The playbook starts by deleting the old `vulnerabilities.yml` file and previous JSON reports. It then creates a `trivy-reports` folder inside `/risk`.

```yaml
collected_on: "2025-05-24"
scanner: trivy_container
images_scanned: 30
unique_cve_count: 96
vulnerabilities:
  - VulnerabilityID: CVE-2023-2953
    cwe:
    - CWE-476
    images:
    - ghcr.io/bitwarden/admin:2025.5.1
    - ghcr.io/bitwarden/identity:2025.5.1
    - invoiceninja/invoiceninja-debian:latest
    - nextcloud:latest
    - ghcr.io/bitwarden/nginx:2025.5.1
    severity: HIGH
    title: 'openldap: null pointer dereference in  ber_memalloc_x  function'
  - VulnerabilityID: CVE-2023-31484
    cwe:
    - CWE-295
    images:
    - ghcr.io/bitwarden/admin:2025.5.1
    - invoiceninja/invoiceninja-debian:latest
    - nextcloud:latest
    - ghcr.io/bitwarden/nginx:2025.5.1
    severity: HIGH
    title: 'perl: CPAN.pm does not verify TLS certificates when downloading
    ↪  distributions
      over HTTPS'
```

```
28   - VulnerabilityID: CVE-2023-45853
29     cwe:
30     - CWE-190
31     images:
32     - ghcr.io/bitwarden/admin:2025.5.1
33     - invoiceninja/invoiceninja-debian:latest
34     - nextcloud:latest
35     - ghcr.io/bitwarden/web:2025.5.0
36     severity: CRITICAL
37     title: 'zlib: integer overflow and resultant heap-based buffer overflow in
       ↪  zipOpenNewFileInZip4_6'
38   - VulnerabilityID: CVE-2023-24998
```

Next, it extracts all active images for a full audit. Before scanning, Trivy initializes a local CVE cache to avoid downloading the vulnerability database repeatedly for each image. It then filters only high and critical vulnerabilities. Once the scan is done for each image, Trivy generates `JSON` files inside `trivy-reports`, providing valid artifacts.

To make Trivy's output more readable, we aggregate the results into a single `vulnerabilities.yml` file, which lists all CVEs by image. This file includes the scan date, list of images, number of unique CVEs, and details for each vulnerability.

This approach allows us to scan only the active images, making our automation modular. If a company does not use a certain service, it is not activated and therefore not included in the scan. Also, Trivy's database is always up to date, which helps us stay current and improves our automation. Each result is timestamped and versioned, with vulnerabilities listed by unique ID.

Compliance Demonstration – CyFun ID.RA-1 Basic Level

The Basic level of the ID.RA-1 sub-function requires the organization to identify and document vulnerabilities present on its digital assets. The goal is to provide an initial understanding of the risks related to the exposure of certain services or software.

In our case, this requirement is addressed through the automation of scans on active containers deployed in the JANUS infrastructure. Thanks to the Trivy tool, Docker images that are currently running are analyzed during each redeployment. The file `vulnerabilities.yml` then groups the results in a centralized and structured way, including CVE identifiers, severity levels, and affected components.

This approach meets the expectations of the Basic level, which requires regular identification of known vulnerabilities. The scope is limited to active containers, which is a realistic and suitable practice for SMEs. It avoids analyzing inactive or unused services, which makes the analysis simpler and still relevant.

Even if this approach does not yet include a deep risk evaluation or automatic correlation with current threats, it provides a concrete and usable base to start managing vulnerabilities. The organization has a clear inventory of detected flaws, with timestamped, reproducible artifacts aligned with recommended practices.

In summary, this automation allows small organizations to reach the expected level without needing advanced technical expertise, while laying the foundation for a more mature vulnerability management process.

ID.RA-5: Risk Determination and Prioritization

The ID.RA-5 sub-function of the CyFun framework is the logical conclusion of the Identify function. After listing assets, threats, and vulnerabilities, it is now necessary to combine these elements to model concrete risk scenarios. These risks need to be prioritized to guide treatment and governance measures.

In the context of our JANUS infrastructure, which is adapted for accounting firms, this prioritization helps focus security efforts on resources that are truly critical for business continuity. Accounting firms must ensure the protection of personal data as well as regulatory compliance.

For this implementation, we chose a hybrid approach that combines automation for collecting technical vulnerabilities (already implemented for ID.RA-1) and manual modeling of relevant threats and vulnerabilities in a file named `manual-threats.yml`(Appendix C.8), followed by risk scenario construction in `manual-risks.yml`(Appendix C.7).

This approach is based on the principle that risk is not just about a single technical vulnerability, but always results from a specific chain of events:

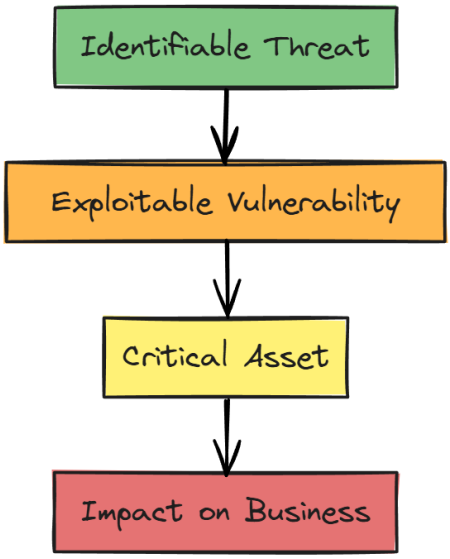

Figure 6.1: Example of risk chain

The file `manual-risks.yml` acts as a structured risk register, where each entry includes:

Field	Description
Targeted asset	The specific system, data, or component under risk
Modeled threat	The described threat that could exploit the vulnerability
Vulnerability	The flaw or weakness that enables the threat
Impact analysis	The impact on Confidentiality, Integrity, and Availability
Probability	Likelihood that the scenario occurs
Risk level	Overall score or level of risk assigned
Mitigation measures	Controls already in place or planned actions

Table 6.1: Structure of a risk entry in `manual-risks.yml`

This allows for precise and differentiated analysis. For example, a vulnerability affecting Bitwarden carries significantly more weight than a flaw in Draw.io, since Bitwarden is central to protecting account secrets, compared to plans or diagrams.

Justification for Manual Evaluation

Despite technological advances in automated vulnerability detection, the idea of a fully automated risk analysis is still not suitable today.

First, the business impact of a given vulnerability cannot be reliably determined by a technical scanner. A CVE identifier provides information about a known flaw, its exploitation conditions, and its standardized technical severity (often based on CVSS). However, it does not predict its real impact on the organization, which depends on many factors such as the criticality of the targeted asset, its position in the architecture, or how important it is for business operations.

For example, a critical vulnerability affecting an internal web application not exposed to the internet will not carry the same level of risk as a medium vulnerability affecting a public-facing service handling sensitive data. This exposure context analysis cannot be done without a human understanding of the environment.

Second, threat scenarios are highly contextual, and their likelihood depends on the organization's activity, attack surface, and profile. An accounting firm, handling tax and banking data, is more likely to be targeted by phishing or ransomware campaigns than by large-scale DDoS attacks. Detection tools do not have the capacity to integrate this sector-specific knowledge or to evaluate an attacker's economic motivations. Manual threat modeling allows the inclusion of credible hypotheses based on past events or known organizational weaknesses.

Moreover, risk acceptability varies greatly from one company to another. It is influenced by factors that are often non-technical, such as risk tolerance, cybersecurity budget, or the organization's maturity level. A small business with low risk appetite will adopt stricter thresholds than a more resilient company or one with insurance mechanisms in place.

Finally, it should be noted that manual analysis brings organizational value that goes beyond technical accuracy. It supports better shared understanding between stakeholders. This helps a company demonstrate during an audit that it was capable of identifying and analyzing its risks and made a conscious decision to accept or treat them.

This is not about rejecting automation, but rather about using it to our advantage and assigning it a clear role: provide technical data, while manual evaluation enhances

understanding and helps the company assess and prioritize risks by actively taking part in the decision-making process.

Compliance Demonstration – CyFun ID.RA-5 Basic Level

The Basic level of the ID.RA-5 sub-function requires the organization to identify its risk scenarios based on the previously collected information (assets, vulnerabilities, threats) and to prioritize them according to their potential impact and likelihood. This step is the logical conclusion of the Identify function in the CyFun framework, as it transforms observations into actionable decisions.

In our case, the approach we used combines technical automation, already implemented for vulnerability detection, with manual modeling of threats and risk scenarios. This combination allows us to reflect the real context of small companies, especially accounting firms, and to include aspects that automated tools cannot handle alone.

The file `manual-risks.yml` serves as a structured risk register, where each risk is described using a set of clear fields: targeted asset, identified threat, exploited vulnerability, impact on confidentiality, integrity, and availability, likelihood, risk level, and mitigation measures.

Beyond the analysis itself, this method also helps strengthen governance by encouraging a collective reflection on security priorities. It offers a solid base to demonstrate, during an audit, that the organization has identified its risks, understands them, and has made rational decisions about how to treat them.

Even though this approach is not fully automated, it still meets the expectations of the Basic level for the ID.RA-5 sub-function. It enables a small company to maintain a clear and usable risk register, aligned with its actual capabilities, and lays the groundwork for a more mature security management process.

6.2 File and Inventory Structure in JANUS

To ensure clarity within the JANUS infrastructure, special attention was given to how files and inventories are organized. The figure below 6.2 illustrates the tree-based architecture we implemented, following a modular and function-oriented vision.

The root folder `Janus/` is the main entry point of the project. It contains both automation components and the various domains that can be launched. Inside the `domain/` directory, and more specifically in `Cyfun/identify/`, we concentrated the implementation of the Identify function.

We made this choice to isolate compliance logic from user environments and other modules, while also leaving room for future expansion of the other CyFun functions.

The `identify/` folder itself is divided into subfolders that correspond to the sub-functions defined in the CyFun framework. In addition to the sub-functions, we created a `checklist/` folder to host control lists designed for CyFun implementers or internal audit purposes.

To support better understanding, even though it is already mentioned inside the checklists, all documents that need to be filled manually are consistently named with the prefix `manual-*`, which helps simplify governance and documentation tracking.

Moreover, this architecture could be transferable. A company that wants to replicate this approach could simply clone the JANUS environment and adjust the YAML files to

match its own perimeter. This would provide a ready-to-use structure and documentation baseline for compliance purposes.

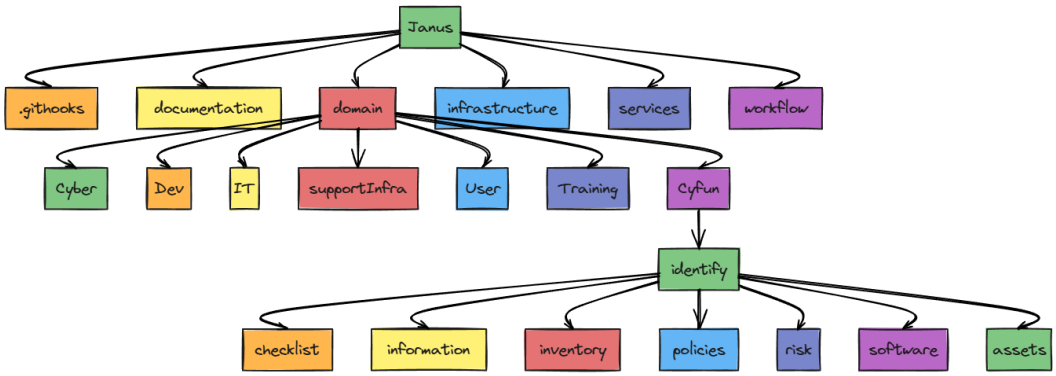

Figure 6.2: Infrastructure Janus with cyfun

6.3 Tests and Validation of the Infrastructure

This section aims to show that the JANUS infrastructure works as expected, that it meets the Basic level compliance requirements of the CyFun framework, and that the produced artifacts (inventories, procedures, checklists) are usable, understandable, and traceable.

The goal here is not to test technical performance or do a full security audit, but to verify that the sub-functions of the Identify category of the CyFun framework have been correctly implemented, with a sufficient level of documentation for them to be used by a small business without a dedicated IT team.

6.3.1 Objectives of the Validation

The main objective of this validation phase is to confirm that all mechanisms deployed in the JANUS infrastructure do meet the Basic level requirements of the CyFun framework for the Identify function.

More specifically, the validation checks that each implemented sub-function produces the expected artifacts, and that these artifacts can be accessed, understood, and used by a non-technical user. These artifacts must also be usable as evidence in case of a compliance audit, which means they need to follow certain formal rules in their format, location, and content.

This is not a performance or load test. The validation focuses on alignment with the framework objectives. That means we want to verify that:

- the automated playbooks produce the required files
- the files are complete, readable, and stored in the correct locations
- the infrastructure design allows for repeatability of actions

This validation shows the educational and operational value of the project. Even a company with no technical knowledge can, using this structure, identify its assets, organize its internal processes, and provide documentation that meets the Basic level requirements of the CyFun framework.

6.3.2 Validation Workflow

The validation process simulates the behavior of a typical small business by checking at each step that the JANUS infrastructure components correctly perform their compliance role according to the Identify function requirements of the CyFun framework.

First, a complete environment is deployed using Vagrant, which orchestrates the creation of virtual machines via VirtualBox. This deployment is fully automated and generates an infrastructure identical to what an SME would use in real conditions.

For validation purposes, a specific configuration activates CyFun compliance functions. The process only requires Vagrant and VirtualBox to be installed and can be executed locally without complex setup.

With a single command: `set "USE_CYFUN=true" && vagrant destroy -force && vagrant box update && vagrant up` we initialize the environment variable to `true`, which triggers the compliance playbooks for the Identify function inside the Vagrantfile. This same Vagrantfile can later be used to launch the other CyFun functions.

Each playbook corresponds to a CyFun sub-function. For example, the `collect_physical.yml` playbook is used to meet the requirements of item ID.AM-1 (physical asset inventory). After the playbooks are executed, the generated files are manually reviewed to check their integrity, content, and compliance with framework expectations.

For example, in the `assets.yml` file, fields like virtual machine name, operating system, RAM, and IP address must be correctly filled in.

6.3.3 Test Results Summary

The table below 6.2 summarizes the test results for each implemented sub-function of the *Identify* category from the CyFun framework (Basic level). For each sub-function, we indicate the Ansible playbook used, the generated output files, and the validation status based on the corresponding checklist.

6.3.4 Limitations of the Validation

Although the different playbooks and documentation used during the validation of the JANUS infrastructure have shown alignment with the Basic level requirements of the Identify function of the CyFun framework, some limitations must be acknowledged.

First, certain elements still need to be added manually, which limits the full automation of the process. For example, equipment that cannot be detected automatically, or software such as SaaS, is not part of the JANUS infrastructure. Similarly, organizational communication flows, which are more related to human mapping than technical detection, require manual modeling.

Second, no external audit has been conducted to validate the robustness or completeness of the generated files. Compliance with the CyFun framework requirements is based here on internal validation using the developed checklists and a strict interpretation of the Basic level criteria.

Finally, the results obtained are specific to the JANUS environment, as deployed in a controlled test setting. Although this infrastructure represents a typical accounting firm, an adaptation would be necessary before any real-world deployment. The specificities of each company (architecture, internal policies, digital culture) can influence the relevance or content of the default artifacts produced.

Sub-function	**Playbook used**	**Output generated**	**Checklist status**
ID.AM-1	`collect-assets.yml`	`assets.yml`, `manual-assets.yml`	✓ Partial
ID.AM-2	`collect-software.yml`	`assets_software.yml`, `manual_software.yml`	✓ Partial
ID.AM-3	`collect_all_inventory.yml`	`assets_information.yml`, `inventory.yml`	✓ Compliant
ID.AM-4	(in ID.AM-2)	/	✓ Partial
ID.AM-5	`prioritization_matrix.yml`, `prioritize_inventory.yml`	`prioritized_inventory.yml`	✓ Compliant
ID.GV-1	manual doc	`cybersecurity-policy.md`, `cybersecurity-procedures.md`	✓ Partial
ID.GV-3	manual doc	`legal_compliance.md`	✓ Compliant
ID.GV-4	manual doc	`risk_management_plan.md`, `risk_strategy.md`	✓ Partial
ID.RA-1	`collect_vulnerabilities.yml`	`vulnerabilities.yml`, JSON reports	✓ Compliant
ID.RA-5	manual model	`manual-threats.yml`, `manual-risks.yml`	✓ Partial

Table 6.2: Summary of test results for CyFun Identify sub-functions

6.3.5 Conclusion of the Validation

The tests carried out show that the implementation of the Identify function in the JANUS infrastructure is functional, reproducible, and compliant with the Basic level requirements of the CyFun framework. Each sub-function has been translated into technical or organizational mechanisms, producing the expected artifacts, and these elements are traceable thanks to the associated checklist system.

This approach shows that it is possible to implement a solid infrastructure and a foundation for cybersecurity compliance, even in organizations with limited resources. By automating what can be automated and supporting manual tasks with clear and structured materials, we provide a concrete starting point to help small businesses structure their governance and move closer to full compliance.

Chapter 7
CyberSecurity analysis of your project implementation

In this chapter, we discuss the intrinsic security of the JANUS project. Even though the main objective of this work was to implement the Identify function of the CyFun framework, we must ensure that the JANUS platform, as a whole and for each of its components, is designed and operated using a security-first approach. In other words, it is important to verify the robustness of our project.

From the beginning of our implementation, one of the core principles of JANUS was to never trust by default. This means that every change pushed to the Git repository must be automatically checked and manually reviewed before being accepted. The main branch of the JANUS codebase is configured as a protected branch in GitLab. No contributor, even with write permissions, can push directly to it. This forced us to create a feature branch for every change, then open a merge request.

For the merge request to be accepted, the CI/CD pipeline must succeed, and another team member must approve the merge. This means the code is reviewed both automatically and by a human before being included in the main branch. This double validation helps guarantee quality and security in the project. It prevents simple commits from introducing misconfigurations or, worse, malicious scripts. In the project, we also implemented Git hooks at the local level to block as early as possible any commit that does not follow our good practices. For example, we check that the branch name follows a strict format, and that any modified `Vagrantfile` is validated using the `vagrant validate` command.

We also enforce commit message formats. For each commit message, we must define its type to bring more clarity to every commit. This helps prevent the main repository from being polluted by syntax errors or non-compliant practices, and fits perfectly into our logic of never trusting by default.

During the design phase, we also reconsidered how JANUS uses VirtualBox and Vagrant to deploy virtual machines. Originally, JANUS needed static IPs for each service, which meant VirtualBox had to create bridged network interfaces or modify the host routing table. These actions required admin privileges on the host machine, which is incompatible with a least privilege posture.

To fix this issue, we replaced the use of static IPs with NAT + port forwarding. Each service now runs in NAT mode, and necessary ports are forwarded from the host to the VM. Thanks to this change, no task requires elevated privileges on the host to start or interact with a VM. Then, the project was structured to prioritize isolation and traceability. For example, Ansible roles are organized into clearly named folders. We also use Ansible Vault systematically to encrypt all secrets. Sensitive files can only be decrypted with the Vault key.

To detect configuration issues or vulnerabilities, we have several automatic checks in the CI/CD pipeline:

- YAML linting to make sure all config files are well-formed.

- Molecule tests to simulate Ansible role execution and check that outputs match expected schemas without errors.
- Automatic VM scanning with Trivy to detect known vulnerabilities in the images.
- Static code analysis with SonarQube.
- DAST scanning using OWASP ZAP against deployed web services, to check for common flaws like XSS or SQL injection.

These automated controls, combined with manual review before merging to `main`, help create a solid foundation for a secure infrastructure. We aim to ensure that playbook structure is always compliant, and dependencies do not contain known vulnerabilities. This helps defend against both internal and external threats.

Among the risks we identified, we could face a malicious code injection by someone in the project or a compromised account. This attacker could modify an Ansible playbook to introduce a malicious script. If this playbook passes the automatic controls, it could silently execute malicious code to open a backdoor in the generated VMs. To counter this threat, we validate automatically and require a human review during the merge to the main branch by someone else than the person who proposed the changes.

Another example of risk could be an attacker exploiting a vulnerability in a service. Thanks to the segmentation of our infrastructure, the impact of this compromise is strongly limited. Each service runs in an isolated virtual machine in NAT mode with specific port forwarding, which prevents the attacker from directly moving to other internal services or to the host.

Despite these precautions, our risk analysis reveals some security limitations that must be mentioned. First, we do not currently apply cryptographic signing to playbooks or Packer artifacts. In the future, it would be useful to sign each playbook with a key and verify this signature before execution.

Second, regarding automatic asset inventory detection: our script runs only at provisioning time (when a VM is started or rebuilt via `vagrant up`). If a VM's configuration is modified between runs, either by an admin or an attacker, the change will not be detected until the next rebuild. An improvement would be to run inventory jobs periodically using scheduled tasks like `cron`.

To conclude this chapter, we can say that the JANUS platform, as we have deployed and adapted it, is built on a solid foundation that respects the security-first requirement. This also shows that JANUS is not just a tool to help others comply with standards. It applies the necessary cybersecurity principles to itself. Its future development can build on what is already in place to further strengthen resilience, while keeping the simplicity and reproducibility needed to serve as a reference for small organizations.

Chapter 8
Discussion

8.1 Comparison with state of the art/related works

In the existing literature, several frameworks and solutions are proposed to try to improve cybersecurity for SMEs, but none of them fully meet the specific needs of small and medium Belgian accounting firms, while also offering a ready-to-use and modular solution.

The main objective of this work was not to directly compete with existing solutions in SME cybersecurity, but to directly propose a complementary solution, which can take into account the technical reality of small accounting firms. And also to provide the beginning of a "ready to use" solution compared to the others. To explain our approach, it is useful to position our contribution in relation to existing initiatives such as KIS, PUZZLE, and Cyberismo.

First, the nature of the approaches is very different. Where the initiatives studied in the state of the art often aim to provide a framework, a platform, or support, our project takes a different direction. It is a prototype of a secure, reusable, and documented infrastructure that can be adapted to other contexts with limited effort. That means that the JANUS project, in its current modification, does not stop at recommendations or tools to assemble, but directly offers a coherent, automated environment structured by the basic CyFun compliance framework. With our main goal being to make deployment as simple as possible.

Unlike **KIS**, which mainly relies on manual audits and recommendations implemented by approved providers, our solution aims for automation from the initialization phase. KIS is very useful to raise awareness in small companies and to start a reflection on cybersecurity, but it does not provide a concrete technical solution that can be deployed independently. Janus offers an infrastructure with open-source tools that does not directly depend on third-party people.

As for **PUZZLE**, the project explores advanced technologies like artificial intelligence and blockchain to improve SME cybersecurity, but it remains mainly a research framework and is not yet available. Even if it might be promising in the long term, integrating it into a small Belgian accounting firm remains complex. Also, PUZZLE does not deliver a deployable pipeline or usable tools. Our project fills this gap by providing a set of files and playbooks that can be used immediately to create a secure environment.

Finally, **Cyberismo** adopts a modern and DevSecOps-oriented vision, with open-source tools and a will to simplify compliance. It is an interesting approach for technical teams already organized around software development. However, for companies without internal developers, like most accounting firms, entering a DevSecOps logic can be too technical. This is where JANUS takes another direction, simplifying everything and avoiding advanced practices so that people without deep technical knowledge can still use it.

To sum up this comparison, the different initiatives bring important bricks to different aspects of SME cybersecurity. Whether it is awareness, integration into development processes, or even innovation. But there is still no solution or approach that provides a coherent, modular infrastructure, compliant with CyFun basic level, and accessible for deployment in small accounting firms. This is exactly the gap we tried to fill, by showing

that it is possible to have a complete infrastructure, largely automated, and aligned with a national framework.

8.2 Difficulties and lessons learned

8.2.1 Difficulties encountered

One of the first obstacles was the automatic identification of network assets. At the beginning, the goal was to use tools like Nmap to scan the network and automatically list all devices. However, this approach quickly became too complex to integrate within the limited time, especially because of the project constraints. In addition, the JANUS infrastructure was based on static IP addresses, which was not compatible with the NAT and port forwarding approach required by the new project vision. So, it was necessary to deeply change the network configuration of the services to match these constraints, which required a lot of effort.

Another major technical difficulty was the creation of the vulnerability collection playbook with Trivy. The first tests did not collect vulnerabilities reliably, and the lack of a cache mechanism made the analysis very slow. So, I had to adapt the process to introduce a local cache, and to structure the playbook to make it reusable, understandable, and aligned with CyFun.

Finally, at the strategic level, the decision to focus only on the **Identify** pillar of the CyFun framework became obvious. Trying to cover all functions of the framework in a superficial way would have diluted the effort and would not have guaranteed a usable result. The goal was to produce a complete, structured, and documented implementation of one pillar, so that it could be used as a base for future work. Thanks to this approach, it is now easy to add a new pillar without changing the general structure of the project.

8.2.2 Lessons learned

This project showed the importance of setting realistic and well-defined goals. Trying to do everything can lead to something unmaintainable. On the other hand, focusing on one function at 100% makes it possible to deliver a clean, usable, and understandable result for future contributors.

Another important lesson is related to time and priority management. In retrospect, I would have spent less time on the theoretical part (state of the art) at the beginning, and more time on the initial structure of the implementation. Having a clear vision from the start would have helped to better anticipate the problems related to network configuration, playbook organization, and file reusability.

At the personal level, this project allowed me to discover **Ansible** and **Vagrant**, two powerful tools I did not know before, which are very useful for reproducible environments. It also taught me the importance of documentation, file structure, and clarity in comments and variable names. A well-organized documentation makes the project much more accessible to others, and helps for future improvements.

Finally, this work showed how important communication and coordination are, even in an academic context. This kind of collaborative project could benefit from more regular interactions with other students working on JANUS, to better align contributions and quickly identify common difficulties.

8.3 Limitations of validity

This work made it possible to propose a first solution for an operational and automated infrastructure aligned with the CyFun framework. Despite this, it presents several limitations that are important to acknowledge in order to better understand the scope and validity of the results obtained.

First, we decided to focus only on the Identify function of the CyFun framework. This choice was motivated by a desire for depth rather than coverage, meaning to produce a complete, consistent, documented, and reusable implementation instead of providing a partial and incomplete version of the entire framework. While this strategy allowed a complete execution and a better quality result, it does not allow us to evaluate the feasibility of covering the full framework with the same automation approach.

Another limitation lies in the absence of formal evaluation by end users or business experts. The results produced were not tested in a real accounting firm or reviewed in depth by professionals outside the academic context. Therefore, even if the system works technically, its real use remains theoretical. Its acceptability by users still needs to be confirmed.

In addition, the implementation and tests were carried out in a controlled virtual environment, with a simulated network using Vagrant and VirtualBox. This setup, while suitable for technical validation, does not fully reflect the complexity of real SME environments.

Finally, even if the solution was designed to be reusable in other SME contexts, it is still optimized for a specific target profile: small Belgian accounting firms. Generalizing the solution to other sectors would require specific adjustments.

8.4 Future work

This work represents a first step towards the implementation of a secure, automated, and cybersecurity-compliant infrastructure for Belgian SMEs. Several improvement and extension ideas can be considered to further develop the results obtained.

8.4.1 Extension to the other CyFun functions level basic

The logical continuation of this project would be to progressively integrate the other functions of the CyFun framework (Protect, Detect, Respond, Recover). The modular structure implemented in JANUS was specifically designed to allow this extensibility. Each function could be developed separately, with its own playbooks, analysis tools, and documentation files, while respecting the conventions already defined in the project.

8.4.2 Deployment and validation in real companies

Another important step would be to test the infrastructure in a real accounting firm or in a partner SME. This would make it possible to get concrete feedback on the understanding of the reports, the use of the infrastructure, but also the documentation needs. It would help us to understand the different adoption obstacles, and we could consider offering adapted training for onboarding the infrastructure. These field tests are essential to truly validate the usability and operational relevance of the solution, beyond a simple academic framework.

8.4.3 Adding a user interface

To improve accessibility for non-technical users, a future development could include a simple graphical interface to run playbooks, view results, or check compliance levels. We believe this is one of the most important elements for adoption in companies where technical knowledge is limited. Such an interface would allow SME managers, without advanced system administration skills, to deploy and manage their infrastructure independently. An infrastructure with a graphical interface would make it accessible to all types of companies.

8.4.4 Improvement of detection and analysis capabilities

Some features, such as automatic network asset detection, could not be integrated in this version due to technical complexity and time constraints. It would be relevant to reintroduce these ideas in a future version, by exploring, for example, the integration of passive scanners. We must also take into account that accounting firms have clients who connect to the network, and therefore it is important to distinguish between a client asset and an employee asset. One possibility could be to use two separate networks. This would allow an automated inventory of all assets, not just the infrastructure, and make it possible to complete the asset inventory in a continuous and discreet way.

Chapter 9
Conclusions

Reminder

In this thesis, we explored the feasibility of implementing and developing a modular, secure and automated infrastructure, adapted to small accounting firms in Belgium, and aligned with the CyFun framework. The goal was to address the lack of practical and accessible solutions for SMEs that must comply with growing cybersecurity requirements.

Small accounting firms in Belgium face increasing pressure to comply with cybersecurity regulations. New frameworks like CyFun are there to help, but they do not take into account the lack of human and financial resources to implement standard solutions. The real issue is that implementing a framework like CyFun, even at its basic level, requires technical expertise and a lot of time. While some solutions exist like Cyberismo, KIS or PUZZLE, they are still too complex, inaccessible or too generic for structures like accounting firms. This thesis proposes a ready-to-use, fully automated infrastructure, based on the JANUS project. The infrastructure was modified and structured to align with the Identify function of the CyFun framework, with a focus on modularity, documentation and automation. The implementation integrates tools such as Vagrant, Ansible and Trivy, enabling asset identification (with some limitations), vulnerability scanning and risk documentation.

Our contribution is both technical and methodological. The choice to focus on one single function of the framework allows a clean and structured implementation, reusable as a base for future work. Instead of partially covering all the pillars, the thesis offers a complete and usable demonstration of one function in depth. The result is a deployable proof of concept that shows how small companies can achieve a significant level of cybersecurity compliance with minimal manual intervention. This project does not replace expert knowledge or advanced infrastructure, but it lowers the technical barrier and proposes an evolutive and reusable template.

Research questions

Question 1: Is it possible to design a secure, modular infrastructure that complies with the CyFun framework, while remaining adapted to the technical and organizational constraints specific to small accounting firms in Belgium?

Yes, this thesis shows that such an infrastructure is achievable. Using simple open-source tools, we built a clear, versioned and documented architecture, deployable on a local machine or a small server. The use of Vagrant and Ansible helped reduce the complexity of deployment while ensuring security by design. The infrastructure is adapted to small structures without specialized personnel.

Question 2: To what extent can the automation of cybersecurity functions, particularly the Identify function, help compensate for the lack of specialized personnel within SMEs?

Automation plays an essential role. Tasks such as asset inventory, system information collection, and vulnerability analysis are executed automatically using Ansible and Trivy. Reports are generated without manual action, which greatly reduces dependency on internal expertise. Minimal supervision is still required, but efficiency and regularity of analysis are ensured.

Question 3: Can the JANUS project be adapted to serve as a reusable infrastructure model for other SME sectors, while ensuring scalability, understandable documentation, and security by design?

Yes, even if this thesis focused mainly on small Belgian accounting firms, the modular organization of the files and the versioned documentation make the project easily adaptable. Each CyFun function can be integrated separately without changing the overall architecture. SMEs in other sectors could reuse the structure by adapting only the critical services or target assets. The project remains simple, readable, and follows a minimal DevSecOps logic.

Conclusion

This project shows that it is possible to make CyFun framework compliance and security accessible while considering the limited resources of SMEs. By addressing the real constraints of Belgian accounting firms, this thesis lays a first foundation for a broader approach: making cybersecurity affordable, understandable, and applicable to SMEs without sacrificing quality or methodology.

Bibliography

[1] Abdulmajeed Alahmari, B.D.: Cybersecurity Risk Management in SMEs: A Systematic Review of Recent Evidence. 2020 IEEE International Conference on Cyber Security and Resilience (CSR) (2020), accessed on March 25, 2025 [Cited on pages 13, 17, 21, and 30.]

[2] Anastasios Papathanasiou, George Liontos, A.K.V.L.E.G.: Cybersecurity Guide for SMEs: Protecting Small and Medium-Sized Enterprises in the Digital Era. Journal of Information Security pp. 1–43 (2025), accessed on March 25, 2025 [Cited on pages 17 and 18.]

[3] Ansible: ansible.builtin.slurp module (2025), `https://docs.ansible.com/ansible/latest/collections/ansible/builtin/slurp_module.html` [Cited on page 41.]

[4] Ansible: Creating a playbook (2025), `https://docs.ansible.com/ansible/latest/getting_started/get_started_playbook.html`, accessed on May 15, 2025 [Cited on page 8.]

[5] Ansible: Introduction to Ansible (2025), `https://docs.ansible.com/ansible/latest/getting_started/introduction.html`, accessed on May 15, 2025 [Cited on page 8.]

[6] Bountouni, N., Koussouris, S., Vasileiou, A., Kazazis, S.A.: A holistic framework for safeguarding of smes: A case study. In: 2023 19th International Conference on the Design of Reliable Communication Networks (DRCN). pp. 1–8. IEEE (Apr 2023) [Cited on pages 2, 13, 14, 16, 17, and 20.]

[7] CCB: CyberFundamentals Framework (2025), `https://atwork.safeonweb.be/fr/tools-resources/cyberfundamentals-framework`, accessed on May 15, 2025 [Cited on page 9.]

[8] Centre, A.C.S.: Secure by design (2025), `https://www.cyber.gov.au/resources-business-and-government/governance-and-user-education/secure-by-design` [Cited on page 9.]

[9] Centre pour la Cybersécurité Belgique (CCB): Cybersécurité – Guide pour les PME. Tech. rep., CCB (2021), accessed on March 25, 2025 [Cited on pages 15 and 16.]

[10] Centre pour la Cybersécurité Belgique (CCB): Cadre des CyberFundamentals (CyFun) (2024), `https://atwork.safeonweb.be/fr/tools-resources/cyberfundamentals-framework`, accessed on May 15, 2025 [Cited on page 2.]

[11] Cheenepalli, J., Hastings, J.D., Ahmed, K.M., Fenner, C.: Advancing DevSecOps in SMEs. Dakota State University Research Report (2024), accessed on March 30, 2025 [Cited on pages 2, 18, and 19.]

[12] Chidukwani, A., Zander, S., Koutsakis, P.: Cybersecurity Preparedness of SMBs. Computers Security (2024), accessed on March 30, 2025 [Cited on pages 13 and 18.]

[13] Chidukwani, Sebastian Zander, P.K.: A Survey on the Cyber Security of Small-to-Medium Businesses: Challenges, Research Focus and Recommendations. IEEE Access 10, 85701–85719 (2022), accessed on March 25, 2025 [Cited on pages 2, 16, 20, and 21.]

[14] fo internet security (CIS), C.: Critical security controls (2016), https://www.cisecurity.org/controls/v8, cIS Critical Security Controls Version 8 [Cited on page 43.]

[15] cisa: Secure by Design (2024), https://www.cisa.gov/securebydesign, accessed on May 15, 2025 [Cited on page 7.]

[16] Cook, K.D.: Effective Cyber Security Strategies for Small Businesses. Walden University Doctoral Dissertation (2017), accessed on March 30, 2025 [Cited on page 13.]

[17] Del-Real, C., Busser, E.D., van den Berg, B.: Shielding Software Systems: A Comparison of Security by Design and Privacy by Design Based on a Systematic Literature Review. Computer Law Security Review (2024), accessed on March 30, 2025 [Cited on page 20.]

[18] docker: What is a container? (2024), https://docs.docker.com/get-started/docker-concepts/the-basics/what-is-a-container/, accessed on May 15, 2025 [Cited on page 7.]

[19] docker: What is Docker? (2024), https://docs.docker.com/get-started/docker-overview/, accessed on May 15, 2025 [Cited on page 7.]

[20] economie fgov: SME definition (2023), https://economie.fgov.be/fr/themes/entreprises/pme-et-independants-en/definitions-et-sources, accessed on May 15, 2025 [Cited on page 6.]

[21] European Commission: SME definition (2020), https://single-market-economy.ec.europa.eu/smes/sme-fundamentals/sme-definition_en, accessed on May 15, 2025 [Cited on page 6.]

[22] European Union: Directive (EU) 2022/2555 of the European Parliament and of the Council on measures for a high common level of cybersecurity across the Union (NIS2 Directive) (2022), https://eur-lex.europa.eu/legal-content/FR/LSU/?uri=oj:JOL_2022_333_R_0002, accessed on March 25, 2025 [Cited on page 1.]

[23] European Union Agency for Cybersecurity (ENISA): Cybersecurity for SMEs: Challenges and Recommendations (2021), https://www.enisa.europa.eu/publications/enisa-report-cybersecurity-for-smes, accessed on March 25, 2025 [Cited on page 1.]

[24] Gasiba, T., Iosif, A.C., Lechner, U., Pinto-Albuquerque, M.: Raising Security Awareness of Cloud Deployments using Infrastructure as Code through CyberSecurity Challenges. Proceedings of the 16th International Conference on Availability, Reliability and Security (ARES) (2021), accessed on March 30, 2025 [Cited on pages 21 and 22.]

[25] Halbach, H.T.: How the Growth of Technology has Forced Accounting Firms to put an Emphasis on Cybersecurity. Bachelor of science in business administration in finance and accounting, University of Arkansas, Fayetteville (May 2021), accessed on March 25, 2025 [Cited on pages 15 and 19.]

[26] Hasan, L., Hossain, M.Z., Johora, F.T., Hasan, M.H.: Cybersecurity in Accounting: Protecting Financial Data in the Digital Age. European Journal of Applied Science, Engineering and Technology 2(6), 64–80 (2024), accessed on March 30, 2025 [Cited on pages 2, 14, 15, 18, and 21.]

[27] hashicorp: Vagrantfile (2024), https://developer.hashicorp.com/vagrant/docs/vagrantfile, accessed on May 15, 2025 [Cited on page 7.]

[28] hashicorp: What is Vagrant? (2024), https://developer.hashicorp.com/vagrant, accessed on May 15, 2025 [Cited on page 7.]

[29] Haverinen, H., Janhunen, T., Päivärinta, T., Kaartinen, S., Lempinen, S., Merilä, S.: Automating Cybersecurity Compliance in DevSecOps with Open Information Model for Security as Code. Proceedings of the 4th European Symposium on Software Engineering and Applications (eSAAM) (2024), accessed on March 30, 2025 [Cited on page 19.]

[30] Henry Haverinen, Tomi Janhunen, T.P.S.L.S.K.S.M.: Automating Cybersecurity Compliance in DevSecOps with Open Information Model for Security as Code. ACM digital library (2024), accessed on March 30, 2025 [Cited on page 21.]

[31] IBM: What is IaC? (2025), https://www.ibm.com/think/topics/infrastructure-as-code, accessed on May 15, 2025 [Cited on page 12.]

[32] Jeremy Grandclaudon: Cyberwal by Digital Wallonia. Dispositif Keep It Secure (2022), https://www.digitalwallonia.be/fr/publications/keepitsecure/, accessed on March 25, 2025 [Cited on pages 16 and 19.]

[33] L'echo: Les PME sont les moteurs puissants de l'économie belge (2010), https://www.lecho.be/dossier/entreprendreenbelgique/les-pme-sont-les-moteurs-puissants-de-l-economie-belge/8986183.html, accessed on May 15, 2025 [Cited on page 6.]

[34] Lepiller, J., Piskac, R., Schäf, M., Santolucito, M.: Analyzing Infrastructure as Code to Prevent Intra-update Sniping Vulnerabilities. Tools and Algorithms for the Construction and Analysis of Systems (TACAS) (2021), accessed on March 30, 2025 [Cited on page 18.]

[35] Mangla, M.: Securing CI/CD Pipeline: Automating the Detection of Misconfigurations and Integrating Security Tools. Master's thesis (msc cybersecurity), National College of Ireland (2023), accessed on March 30, 2025 [Cited on page 19.]

[36] Matthew Finio, Amanda Downie: What is a cyber range (2024), https://www.ibm.com/think/topics/cyber-range, accessed on May 15, 2025 [Cited on page 12.]

[37] Md Abdullahel Kafi, N.A.: Securing financial information in the digital realm: Case studies in cybersecurity for accounting data protection. In: American Journal of Trade and Policy (2023) [Cited on pages 15, 17, and 18.]

[38] National cyber security centre: Understanding vulnerabilities (2024), https://www.ncsc.gov.uk/collection/vulnerability-management/understanding-vulnerabilities, accessed on May 15, 2025 [Cited on page 47.]

[39] National Institute of Standards and Technology: The NIST Cybersecurity Framework (CSF) 2.0 (2024), https://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.29.pd, accessed on May 15, 2025 [Cited on page 3.]

[40] NIST: Cyber Threat (2025), https://csrc.nist.gov/glossary/term/cyber_threat, accessed on May 15, 2025 [Cited on page 47.]

[41] NIST: Cybersecurity Risk (2025), https://csrc.nist.gov/glossary/term/cybersecurity_risk, accessed on May 15, 2025 [Cited on page 47.]

[42] Ozkan, B.Y., Spruit, M.: Adaptable Security Maturity Assessment and Standardization for Digital SMEs. Journal of Computer Information Systems (2022), accessed on March 30, 2025 [Cited on page 17.]

[43] Ozkan, Y., Spruit, M.: Cybersecurity Standardisation for SMEs: The Stakeholders' Perspectives and a Research Agenda. International Journal of Standardization Research (IJSR) pp. 41–72 (2019), accessed on March 25, 2025 [Cited on pages 2, 12, 14, 16, and 22.]

[44] Palma, S.D., Nucci, D.D., Palomba, F., Tamburri, D.A.: Toward a catalog of software quality metrics for infrastructure code. Journal of Systems and Software 170, 110726 (2020), accessed on March 30, 2025 [Cited on page 2.]

[45] Ponsard, C., Massonet, P., Grandclaudon, J., Point, N.: From Lightweight Cybersecurity Assessment to SME Certification Scheme in Belgium. In: 2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW). pp. 75–78 (2020), accessed on March 25, 2025 [Cited on pages 14, 15, and 16.]

[46] Rana, O., Al-Maatouk, Q., Ling, L.: Critical Review of Design Considerations in Forming a Cloud Infrastructure for SMEs. International Conference on Decision Aid Sciences and Applications (DASA) (2022), accessed on March 30, 2025 [Cited on page 19.]

[47] Rombaldo Junior, C., Becker, I., Johnson, S.: Unaware, Unfunded and Uneducated: A Systematic Review of SME Cybersecurity. arXiv preprint arXiv:2309.17186 (2024), https://arxiv.org/abs/2309.17186, accessed on March 25, 2025 [Cited on pages 2, 3, 13, 14, and 18.]

[48] Sadiq, S.: Analysis of Cybersecurity Threats to Financial and Accounting Data: Implications for Organizational Risk Management. Honors bachelor's thesis, Brac University (2023), accessed on March 25, 2025 [Cited on pages 14, 17, and 19.]

[49] Statbel: Structural Business Statistics 2022: 95.9% of Belgian Enterprises Are Micro-Enterprises (2022), https://statbel.fgov.be/en/news/structural-business-statistics-2022-959-belgian-enterprises-are-micro-enterprises, accessed on March 25, 2025 [Cited on page 1.]

[50] Stephanie Susnjara, Ian Smalley : What is virtualization? (2025), https://www.ibm.com/think/topics/virtualization, accessed on May 15, 2025 [Cited on page 6.]

[51] Sánchez-Gordón, M., Colomo-Palacios, R.: Security as Culture: A Systematic Literature Review of DevSecOps. Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops (ICSEW) (2020), accessed on March 30, 2025 [Cited on page 21.]

[52] Teppan, L.E., Flå, L.A., Jaatun, M.G.: A Survey on Infrastructure-as-Code Solutions for Cloud Development. CloudCom (2022), accessed on March 30, 2025 [Cited on page 19.]

[53] Tetteh, A.K.: Cybersecurity Needs for SMEs. Issues in Information Systems 25(1), 237–245 (2024), accessed on March 25, 2025 [Cited on pages 2 and 13.]

[54] Thomas Joswig, W.K.: Empirical analysis of nis2 adoption in eu smes: Challenges for critical infrastructure in germany. In: Journal of Next-Generation Research 5.0 (2025) [Cited on pages 3, 13, 14, 15, 17, 20, and 21.]

[55] Thool, A., Brown, C.: Integrating DAST in Kanban and CI/CD: A Real-World Security Case Study. arXiv preprint (2024), accessed on March 30, 2025 [Cited on page 19.]

[56] Verdet, A., Hamdaqa, M., Silva, L.D., Khomh, F.: Exploring Security Practices in Infrastructure as Code: An Empirical Study. arXiv preprint (2023), accessed on March 30, 2025 [Cited on page 19.]

[57] vmware: What is a bare metal hypervisor? (2025), `https://www.vmware.com/topics/bare-metal-hypervisor`, accessed on May 15, 2025 [Cited on page 6.]

[58] Xu, T., Legunsen, O.: Configuration testing: Testing configuration values as code and with code. arXiv preprint arXiv:1905.12195v2 (July 2019) [Cited on page 12.]

Appendix A

Survey Results: Target Profile Validation

A.1 Summary of responses

The responses collected from five Belgian accounting firms are presented as follows:

Question	**Company 1**	**Company 2**	**Company 3**	**Company 4**	**Company 5**
IT / cybersecurity team	Externalized + internal contact	Internal accountant manages IT	External provider	Freelance external	Dedicated IT company
Accounting software used	BOB 50, Sage Cloud Demat	Sage BOB 50, HORUS, SaaS for tax returns	Bob, Horus	Sage BOB	Sage BOB 50
Software for compliance/continuity	Client portfolio management, administrative tools	Sage BOB and Horus	Bob and Horus	Microsoft 365, BOB	Email, BOB 50
Regulations to follow	GDPR	GDPR	No idea	GDPR	GDPR
Information security (GDPR, NIS2, CyFun)	GDPR implemented, client protocol	GDPR yes, CyFun unknown	Basic knowledge of GDPR	GDPR	Basic knowledge of GDPR
Cybersecurity audit / risk assessment	Never done	No	No	No	No
Main risks and threats	Attacks preventing access to data	Malware, phishing	Data loss	Ransomware, data loss	Data loss
Sensitive data protection	None	Dropbox backup + strong passwords	2FA + daily backups on two sites	Restricted access	Weekly backup
Incident response plan / employee awareness	None	Informal awareness, no plan	None	No formal plan	No plan, no training

Appendix B

Source code

B.1 Manual Asset Inventory File (manual-assets.yml)

```yaml
asset_type: human-assets
description: >
  This file is intended to document all organizational assets that cannot be
  automatically collected via the system (e.g. user laptops, mobile phones, printers,
  routers, or any other physical or offline assets).

  Fields marked as REQUIRED must be filled to ensure CyFun ID.AM-1 compliance.
  Fields marked as OPTIONAL can be left blank or marked as TO_DO if unknown.

last_review: TO_BE_FILLED  # REQUIRED - Last time this file was reviewed (format: YYYY-MM-DD)

assets:
  - id: TO_BE_FILLED            # REQUIRED - Unique asset ID (e.g. laptop-jdoe-001)
    name: TO_BE_FILLED          # REQUIRED - Human-readable name (e.g. John's Laptop)
    type: TO_BE_FILLED          # REQUIRED - Device type (e.g. laptop, printer, router, phone, etc.)
    owner: TO_BE_FILLED         # REQUIRED - Assigned user or department (e.g. Alice, Accounting)
    location: TO_BE_FILLED      # REQUIRED - Physical location or office (e.g. Floor 2, Room 210)
    os: TO_DO                   # OPTIONAL - Operating system (e.g. Windows 11, iOS 16.1)
    ip: TO_DO                   # OPTIONAL - IP address (can be null if offline or unmanaged)
    network: TO_DO              # OPTIONAL - "connected", "not connected", "unknown"
    serial_number: TO_DO        # OPTIONAL - Device serial number or inventory label
    manufacturer: TO_DO         # OPTIONAL - Manufacturer or brand (e.g. Dell, HP)
    purchase_date: TO_DO        # OPTIONAL - Date of purchase (format: YYYY-MM-DD)
    last_review: TO_BE_FILLED   # REQUIRED - Date this asset entry was last checked or updated

  - id: TO_BE_FILLED
    name: TO_BE_FILLED
    type: TO_BE_FILLED
    owner: TO_BE_FILLED
    location: TO_BE_FILLED
```

```
    os: TO_DO
    ip: TO_DO
    network: TO_DO
    serial_number: TO_DO
    manufacturer: TO_DO
    purchase_date: TO_DO
    last_review: TO_BE_FILLED

# -> Add one YAML object (starting with "-") per asset.
# -> Keep this file tracked in Git to support versioning and auditability.
```

B.2 Software Collection Script for CyFun ID.AM-2

The following script is used to automate the collection of installed software and running services on Debian-based systems as part of the CyFun ID.AM-2 sub-function compliance.

```
---
- name: Collect software inventory for CyFun ID.AM-2
  hosts: user_domain
  connection: local
  gather_facts: yes
  remote_user: "{{ user_var }}"

  vars_files:
    - ../../../user/local_or_remote.yml

  tasks:
    - name: Get running Docker containers (services)
      shell: docker ps --format '{% raw %}{{.Names}}|{{.Image}}{% endraw %}'
      register: docker_services
      changed_when: false
      become: true

    - name: Initialize raw docker entries list
      set_fact:
        docker_entries_raw: []

    - name: Normalize Docker container names
      set_fact:
        docker_entries_raw: "{{ docker_entries_raw + [ {
          'clean_name': (
            item.split('|')[0]
            |
            ↪  regex_replace('^(docker-)?(jitsi-meet|docker-jitsi-meet|.*jitsi).*$',
            ↪  'jitsi')
            |
            ↪  regex_replace('^(.*?)(-app|-db|-web|-api|-sso|-mssql|-mysql|-nginx|-redis|-identi
            ↪  '\\1')
            | lower
          ),
          'version': item.split('|')[1]
```

```
      } ] }}"
    loop: "{{ docker_services.stdout_lines }}"

  - name: Extract list of unique clean names
    set_fact:
      unique_service_names: "{{ docker_entries_raw |
      ↪  map(attribute='clean_name') | list | unique }}"

  - name: Initialize grouped docker entries
    set_fact:
      docker_entries: []

  - name: Group versions per service name
    set_fact:
      docker_entries: "{{ docker_entries + [ {
        'name': item,
        'versions': (docker_entries_raw | selectattr('clean_name', 'equalto',
        ↪  item) | map(attribute='version') | list | unique)
      } ] }}"
    loop: "{{ unique_service_names }}"

  - name: Get installed software (Debian/Ubuntu)
    command: dpkg-query -W -f='${Package} ${Version}\n'
    register: raw_software
    changed_when: false

  - name: Initialize software entries list
    set_fact:
      software_entries: []

  - name: Build software entries from package list
    set_fact:
      software_entries: >-
        {{
          software_entries +
          [
            {
              'name': item.split()[0],
              'version': (item.split()[1] if (item.split() | length > 1) else
              ↪  'unknown')
            }
          ]
        }}
    loop: "{{ raw_software.stdout_lines }}"
    no_log: true

  - name: Define new inventory entry with separation
    set_fact:
      new_entry:
        inventory_type: software
        host: "{{ ansible_hostname }}"
        collected_on: "{{ ansible_date_time.date }}"
        os: "{{ ansible_distribution }} {{ ansible_distribution_version }}"
```

```
82          software:
83            docker_services: "{{ docker_entries }}"
84            packages: "{{ software_entries }}"
85
86      - name: Load existing software inventory (if any)
87        slurp:
88          src: /janus/domain/cyfun/identify/software/assets_software.yml
89        register: existing_software_inventory
90        ignore_errors: true
91
92      - name: Parse existing inventory YAML if present
93        set_fact:
94          software_inventory: "{{ existing_software_inventory.content | b64decode
            ↪  | from_yaml }}"
95        when:
96          - existing_software_inventory is defined
97          - existing_software_inventory.content is defined
98
99      - name: Normalize software_inventory into a list
100       set_fact:
101         software_inventory: >-
102           {{
103             software_inventory
104             if software_inventory is defined and software_inventory | type_debug
                ↪  == 'list'
105             else [ software_inventory ]
106           }}
107       when: software_inventory is defined
108
109     - name: Initialize software_inventory if undefined
110       set_fact:
111         software_inventory: []
112       when: software_inventory is not defined
113
114     - name: Replace software inventory with current scan only
115       set_fact:
116         updated_software_inventory: [ "{{ new_entry }}" ]
117
118     - name: Save updated software inventory to file
119       copy:
120         dest: /janus/domain/cyfun/identify/software/assets_software.yml
121         content: "{{ updated_software_inventory | to_nice_yaml(indent=2) }}"
122         mode: '0644'
123
124     - name: Confirm completion
125       debug:
126         msg: " collect-software.yml executed successfully for CyFun ID.AM-2"
```

B.3 Manual Software Inventory File (assets_software_manual.yml)

This file is used to manually document all software and platforms that cannot be automatically detected by the infrastructure. This includes SaaS, mobile apps, legacy tools, or external databases. It is part of the compliance process for CyFun ID.AM-2.

```yaml
inventory_type: manual-software
description: >
  This file is intended to manually document software and platforms that are not
  automatically detectable by the infrastructure inventory system.

  Examples include SaaS platforms, mobile applications, externally hosted databases,
  or any legacy system not provisioned through infrastructure-as-code.

  Fields marked as REQUIRED must be completed to ensure CyFun ID.AM-2 compliance.
  Fields marked as OPTIONAL can be marked as TO_DO or left blank if unknown.

last_review: TO_BE_FILLED  # REQUIRED - Last review date of this inventory (format: YYYY-MM-DD)

software:
  - id: TO_BE_FILLED            # REQUIRED - Unique ID (e.g. saas-01)
    name: TO_BE_FILLED          # REQUIRED - Name of the software or platform (e.g. Google Workspace)
    type: TO_BE_FILLED          # REQUIRED - SaaS, Desktop App, Web Platform, Database, etc.
    description: TO_DO          # OPTIONAL - Short description of what it does
    data_processed: TO_DO       # OPTIONAL - Description of handled data (e.g. payroll, invoices)
    number_of_users: TO_DO      # OPTIONAL - Approximate number of users
    version: TO_DO              # OPTIONAL - Version number or plan (e.g. Business Plus)
    contract: TO_DO             # OPTIONAL - Link or reference to provider contract
    location: TO_DO             # OPTIONAL - Hosting location (e.g. EU, US, cloud region)
    review_date: TO_BE_FILLED   # REQUIRED - Last date the software entry was verified

  - id: TO_BE_FILLED
    name: TO_BE_FILLED
    type: TO_BE_FILLED
    description: TO_DO
    data_processed: TO_DO
    number_of_users: TO_DO
    version: TO_DO
    contract: TO_DO
    location: TO_DO
    review_date: TO_BE_FILLED

# -> Add one entry per non-automated software or platform.
```

```
# -> Keep this file version-controlled to allow auditing and updates.
```

B.4 Collect all inventory for CyFun (collect_all_inventory.yml)

```yaml
---
- name: Collect all inventory for CyFun ID.AM (AM-1 to AM-3)
  hosts: user_domain
  connection: local
  gather_facts: yes
  remote_user: "{{ user_var }}"

  vars_files:
    - ../../../user/local_or_remote.yml

  tasks:
    # Read auto-collected infrastructure assets
    - name: Slurp infrastructure assets
      slurp:
        src: ../assets.yml
      register: raw_infra

    - name: Parse infrastructure assets YAML
      set_fact:
        infra: "{{ raw_infra.content | b64decode | from_yaml }}"

    # Read auto-collected software inventory
    - name: Slurp software inventory
      slurp:
        src: ../software/assets_software.yml
      register: raw_soft

    - name: Parse software inventory YAML
      set_fact:
        soft: "{{ raw_soft.content | b64decode | from_yaml }}"

    # Simplify docker service names and remove versions
    - name: Extract simplified docker service names (first part before dash)
      set_fact:
        simplified_docker_services: >-
          {{
            soft[0].software.docker_services
            | map(attribute='name')
            | map('regex_replace', '-.*$', '')
            | list
            | unique
          }}

    # Remove packages and keep simplified docker services only
    - name: Strip packages and apply simplified docker service names
      set_fact:
```

```
47        soft: >-
48          {{
49            {
50              'inventory_type': soft[0].inventory_type,
51              'host': soft[0].host,
52              'collected_on': soft[0].collected_on,
53              'os': soft[0].os,
54              'software': {
55                'docker_services': simplified_docker_services |
                  ↪ map('regex_replace', '(-.*$)', '') | list | unique
56              }
57            }
58          }}
59
60    # Read information types
61    - name: Slurp information types
62      slurp:
63        src: ../information/assets_information.yml
64      register: raw_info
65
66    - name: Parse information types YAML
67      set_fact:
68        info: "{{ raw_info.content | b64decode | from_yaml }}"
69
70    # Read manual assets
71    - name: Slurp manual assets
72      slurp:
73        src: ../manual-assets.yml
74      register: raw_manual_assets
75
76    - name: Parse manual assets YAML
77      set_fact:
78        manual_assets: "{{ raw_manual_assets.content | b64decode | from_yaml }}"
79
80    # Read manual software
81    - name: Slurp manual software
82      slurp:
83        src: ../software/manual_software.yml
84      register: raw_manual_soft
85
86    - name: Parse manual software YAML
87      set_fact:
88        manual_soft: "{{ raw_manual_soft.content | b64decode | from_yaml }}"
89
90    # Filter out entries not yet filled
91    - name: Filter manual assets (ignore TO_BE_FILLED)
92      set_fact:
93        valid_manual_assets: |
94          {{
95            manual_assets.assets
96            | rejectattr('id', 'search', 'TO_BE_FILLED')
97            | list
98          }}
```

```
99
100     - name: Filter manual software (ignore TO_BE_FILLED)
101       set_fact:
102         valid_manual_software: |
103           {{
104             manual_soft.software
105             | rejectattr('id', 'search', 'TO_BE_FILLED')
106             | list
107           }}
108
109     # Assemble full inventory
110     - name: Assemble full inventory
111       set_fact:
112         full_inventory:
113           collected_on: "{{ ansible_date_time.date }}"
114           host: "{{ ansible_hostname }}"
115           infrastructure: "{{ infra.assets }}"
116           software_auto: "{{ [soft] }}"
117           software_manual: "{{ valid_manual_software }}"
118           information_types: "{{ info.information_types }}"
119           assets_manual: "{{ valid_manual_assets }}"
120
121     # Write combined inventory to file
122     - name: Save full inventory to inventory.yml
123       copy:
124         dest: "{{ playbook_dir }}/inventory.yml"
125         content: "{{ full_inventory | to_nice_yaml(indent=2) }}"
126         mode: '0644'
127
128     - name: Confirm completion
129       debug:
130         msg: " collect_all_inventory.yml executed successfully; inventory saved
          ↪  to {{ playbook_dir }}/inventory.yml"
```

B.5 Prioritization Matrix YAML File (prioritization_matrix.yml)

This YAML file defines the classification, criticality, and operational value for different types of resources. It supports the automation of ID.AM-5 by assigning priorities to each element based on predefined rules.

```
1  default:
2    classification: internal
3    criticality: medium
4    value: moderate
5
6  rules:
7
8    # Logiciels auto-détectés
9    software_auto:
10     - match: bitwarden
11       classification: confidential
```

```yaml
      criticality: high
      value: very_high

    - match: invoiceninja
      classification: internal
      criticality: medium
      value: high

    - match: nextcloud
      classification: internal
      criticality: high
      value: high

    - match: mattermost
      classification: internal
      criticality: medium
      value: moderate

    - match: codimd
      classification: internal
      criticality: low
      value: moderate

    - match: syncthing
      classification: internal
      criticality: medium
      value: moderate

    - match: logseq
      classification: internal
      criticality: low
      value: low

    - match: drawio
      classification: public
      criticality: low
      value: low

    - match: jitsi
      classification: internal
      criticality: low
      value: low

    - match: yourls
      classification: public
      criticality: low
      value: low

  # Logiciels manuels
  software_manual:
    - match: BOB
      classification: confidential
      criticality: critical
```

```
      value: very_high

    - match: Sage
      classification: confidential
      criticality: critical
      value: very_high

    - match: Google Workspace
      classification: confidential
      criticality: high
      value: high

    - match: Office 365
      classification: internal
      criticality: medium
      value: high

  # Informations sensibles
  information_types:
    - match: Client Data
      classification: confidential
      criticality: high
      value: very_high

    - match: Accounting Documents
      classification: confidential
      criticality: high
      value: high

    - match: Credentials and Access Information
      classification: critical
      criticality: critical
      value: very_high

    - match: Client Communications
      classification: confidential
      criticality: medium
      value: moderate

    - match: Internal HR Data
      classification: confidential
      criticality: medium
      value: moderate

    - match: System Logs
      classification: internal
      criticality: low
      value: low

  # Actifs matériels manuels (optionnel si enrichis)
  assets_manual:
    - match: laptop
      classification: internal
```

```
      criticality: medium
      value: moderate

    - match: printer
      classification: internal
      criticality: low
      value: low

    - match: router
      classification: internal
      criticality: high
      value: high
```

B.6 Prioritization Playbook (prioritize_inventory.yml)

The following Ansible playbook applies a prioritization matrix to inventory resources in order to fulfill CyFun ID.AM-5 requirements. It generates a prioritized YAML file sorted by criticality.

```
---
- name: Prioritize inventory for CyFun ID.AM-5 compliance
  hosts: user_domain
  connection: local
  gather_facts: yes
  remote_user: "{{ user_var }}"

  vars_files:
    - ../../../user/local_or_remote.yml

  tasks:
    - name: Slurp collected inventory
      slurp:
        src: inventory.yml
      register: raw_inventory

    - name: Parse inventory YAML
      set_fact:
        inventory: "{{ raw_inventory.content | b64decode | from_yaml }}"

    - name: Slurp prioritization matrix
      slurp:
        src: prioritization_matrix.yml
      register: raw_matrix

    - name: Parse matrix YAML
      set_fact:
        matrix: "{{ raw_matrix.content | b64decode | from_yaml }}"

    - name: Initialize priority lists
      set_fact:
        prioritized_software_auto: []
```

```
      prioritized_software_manual: []
      prioritized_info: []
      prioritized_manual_assets: []

  - name: Prioritize auto software services
    set_fact:
      prioritized_software_auto: >-
        {{ prioritized_software_auto + [ {
            'name': svc,
            'classification': (matrix.rules.software_auto
              | selectattr('match','equalto', svc)
              | map(attribute='classification')
              | list | first
              | default(matrix.default.classification)),
            'criticality': (matrix.rules.software_auto
              | selectattr('match','equalto', svc)
              | map(attribute='criticality')
              | list | first
              | default(matrix.default.criticality)),
            'value': (matrix.rules.software_auto
              | selectattr('match','equalto', svc)
              | map(attribute='value')
              | list | first
              | default(matrix.default.value))
          } ] }}
    loop: "{{ inventory.software_auto[0].software.docker_services
      | map('regex_replace','-.*$','')
      | unique }}"
    loop_control:
      loop_var: svc

  - name: Prioritize manual software entries
    set_fact:
      prioritized_software_manual: >-
        {{ prioritized_software_manual + [ {
            'name': item.name,
            'classification': (matrix.rules.software_manual
              | selectattr('match','equalto', item.name)
              | map(attribute='classification')
              | list | first
              | default(matrix.default.classification)),
            'criticality': (matrix.rules.software_manual
              | selectattr('match','equalto', item.name)
              | map(attribute='criticality')
              | list | first
              | default(matrix.default.criticality)),
            'value': (matrix.rules.software_manual
              | selectattr('match','equalto', item.name)
              | map(attribute='value')
              | list | first
              | default(matrix.default.value))
          } ] }}
    loop: "{{ inventory.software_manual }}"
```

```
86
87    - name: Prioritize information types
88      set_fact:
89        prioritized_info: >-
90          {{ prioritized_info + [ {
91              'name': item.name,
92              'classification': (matrix.rules.information_types
93                | selectattr('match','equalto', item.name)
94                | map(attribute='classification')
95                | list | first
96                | default(matrix.default.classification)),
97              'criticality': (matrix.rules.information_types
98                | selectattr('match','equalto', item.name)
99                | map(attribute='criticality')
100               | list | first
101               | default(matrix.default.criticality)),
102             'value': (matrix.rules.information_types
103               | selectattr('match','equalto', item.name)
104               | map(attribute='value')
105               | list | first
106               | default(matrix.default.value))
107           } ] }}
108     loop: "{{ inventory.information_types }}"
109
110   - name: Prioritize manual assets
111     set_fact:
112       prioritized_manual_assets: >-
113         {{ prioritized_manual_assets + [ {
114             'id': item.id,
115             'name': item.name,
116             'type': item.type,
117             'classification': (matrix.rules.assets_manual
118               | selectattr('match','equalto', item.type)
119               | map(attribute='classification')
120               | list | first
121               | default(matrix.default.classification)),
122             'criticality': (matrix.rules.assets_manual
123               | selectattr('match','equalto', item.type)
124               | map(attribute='criticality')
125               | list | first
126               | default(matrix.default.criticality)),
127             'value': (matrix.rules.assets_manual
128               | selectattr('match','equalto', item.type)
129               | map(attribute='value')
130               | list | first
131               | default(matrix.default.value))
132           } ] }}
133     loop: "{{ inventory.assets_manual }}"
134
135   - name: Combine all prioritized assets
136     set_fact:
137       prioritized_inventory: >-
138         {{ prioritized_software_auto
```

```
            + prioritized_software_manual
            + prioritized_info
            + prioritized_manual_assets }}

  - name: Define criticality weight mapping
    set_fact:
      weight_map:
        critical: 4
        high:     3
        medium:   2
        low:      1

  - name: Annotate each resource with crit_weight
    set_fact:
      weighted_assets: >-
        {{ (weighted_assets | default([]))
           + [ item | combine({'crit_weight': weight_map[item.criticality]}) ]
           }}
    loop: "{{ prioritized_inventory }}"
    loop_control:
      loop_var: item

  - name: Sort assets by descending crit_weight
    set_fact:
      prioritized_inventory: >-
        {{ weighted_assets
           | sort(attribute='crit_weight', reverse=true) }}

  - name: Strip internal crit_weight attribute
    set_fact:
      prioritized_inventory: >-
        {{ prioritized_inventory
           | map('dict2items')
           | map('rejectattr', 'key', 'equalto', 'crit_weight')
           | map('items2dict')
           | list }}

  - name: Save prioritized inventory to file
    copy:
      dest: "{{ playbook_dir }}/prioritized_inventory.yml"
      content: "{{ prioritized_inventory | to_nice_yaml(indent=2) }}"
      mode: '0644'

  - name: Confirm completion
    debug:
      msg: " prioritize_inventory.yml executed successfully;
       prioritized_inventory.yml generated."
```

Appendix C

Governance

C.1 Cybersecurity Policy

The following document defines the cybersecurity policy for the firm. It is stored in the project as a Markdown file named `cybersecurity-policy.md` and supports compliance with sub-function `ID.GV-1`.

```
# Cybersecurity Policy

## 1. Purpose and Scope

This policy defines how the accounting firm protects its information, systems, and data. It applies to all employees, contractors, and third-party users who access company resources.

## 2. Policy Statements

### 2.1 Acceptable Use

* All users must follow approved methods when using computers, networks, and software.
* Passwords must meet minimum complexity requirements.

### 2.2 Access Control

* Access to sensitive systems and data is granted only when needed.
* User accounts must be reviewed and disabled promptly when no longer needed.

### 2.3 Data Classification and Handling

* Information is classified as Public, Internal, Confidential, or Critical.
* Confidential and Critical data must be encrypted in transit and at rest.
* Physical and digital copies of sensitive information must be stored securely.

### 2.4 Incident Reporting

* All security incidents or suspected breaches must be reported immediately to the manager or designated security officer.
* An incident response process must be followed. (TODO: Cyfun : Respond-RS.RP-1 )

### 2.5 Training and Awareness

* All new employees must complete cybersecurity training during onboarding.
* Annual refresher training is mandatory for all staff.

## 3. Roles and Responsibilities
```

```

**Directive Manager**: Approves this policy and makes sure it is applied.

**Accountant(s)**: Follow the rules and help protect client data.

**Secretary**: Follows rules and good security practices.

## 4. Review and Update

* This policy is reviewed at least once a year and after major changes in technology or business processes.
* Approval of changes must come from Directive management.
* Employees will be informed of updates by email and team meetings.

## 5. References

* ISO/IEC 27001:2022, Clauses 4, 5, 7.5
* ISO/IEC 27002:2022, Control 5.1
* CIS Controls v8, Control 14

---

**Last Review:** TO_BE_FILLED
```

C.2 Cybersecurity Procedures

The following document defines the internal procedures that support the cybersecurity policy. It is stored in the project as a Markdown file named `cybersecurity-procedures.md` and contributes to CyFun sub-function ID.GV-1 compliance.

```
# Cybersecurity Procedures

These procedures support the **Cybersecurity Policy** (see `cybersecurity-policy.md`) and ensure the firm stays compliant with **ID.GV-1**. They apply to the directive manager, accountants, secretary, and any external IT provider.

---

## 1. Policy Creation & Approval

1. **Drafting**
   - Directive manager writes or updates `cybersecurity-policy.md`.
   - Use existing checklists (e.g. `checklist-ID.AM-1.md`, `checklist-ID.AM-2.md`) as inputs for scope and controls.

2. **Review by External IT**
   - Send draft to external IT partner for technical feedback (network, encryption, backups).
   - Incorporate their comments into the policy document.

```

```
3. **Final Approval**
   - Directive manager approves the final draft.

## 2. Communication & Distribution

1. **Internal Posting**
   - Copy `cybersecurity-policy.md` and `cybersecurity-procedures.md` to shared folder or intranet.
   - Place a PDF version in the office common area (print or digital display).

2. **Email Notification**
   - Directive manager emails all staff whenever policy or procedures change.
   - Email must include:
     - Summary of changes
     - Link to updated files
     - Deadline to acknowledge receipt (e.g. "Please confirm by reply before YYYY-MM-DD").

3. **Acknowledgement Tracking**
   - Use a simple Google Form or spreadsheet.
   - Each employee (accountant, secretary) signs off with name, date, and policy version.

## 3. Onboarding & Training

1. **New Hires**
   - During onboarding, the secretary shares:
     - `cybersecurity-policy.md`
     - `cybersecurity-procedures.md`
     - `legal_compliance.md`

2. **Training Session**
   - External IT delivers a short (30-minute) walkthrough of key points:
     - Password rules
     - Data classification
     - Incident reporting steps

3. **Annual Refresh**
   - Once per year, schedule a team meeting:
     - Review main policy points
     - Highlight any changes
     - Update acknowledgement list

## 4. Review & Update Cycle

1. **Scheduled Review**
   - Every 12 months, directive manager runs a reminder (e.g., calendar invite or automated task).
   - Review includes:
     - Policy text (`cybersecurity-policy.md`)
     - Procedures (`cybersecurity-procedures.md`)
     - All checklists (ID.AM-1 through ID.RA)

```

```
2. **Triggering Events**
   - Any major change in software (new SaaS platform)
   - Change of external IT provider
   - A security incident or audit finding

3. **Revision Process**
   - Directive manager drafts updates.
   - External IT reviews technical sections.
   - Approval as per Step 1.

4. **Version Tracking**
   - Maintain a simple table at the end of each document:

   | Date       | Author            | Change Summary                |
   | ---------- | ----------------- | ----------------------------- |
   | YYYY-MM-DD | Directive Manager | First release                 |

## 5. Incident Response Alignment

- In case of a reported incident (see Policy 2.4), ((TODO)follow the firm's
↪  **Incident Response Guide**).
- Update both policy and procedures afterward to reflect lessons learned.

---

## References

- **Cybersecurity Policy**:
↪  `domain/cyfun/identify/policies/cybersecurity-policy.md`
- **Inventory Policy**: `inventory-policy.md`
- **Checklists**:
  - Hardware: `checklist-ID.AM-1.md`
  - Software: `checklist-ID.AM-2.md`
  - Information: `checklist-ID.AM-3.md`
  - Prioritization: `checklist-ID.AM-5.md`
  - Risk: `checklist-ID.RA.md`

**Last Review:** TO_BE_FILLED
```

C.3 Legal and Regulatory Compliance

The following Markdown file is part of the documentation used to comply with CyFun subfunction `ID.GV-3`. It describes how the firm addresses key legal obligations in Belgium, including GDPR, accounting law, and potential NIS2 exposure. The file is stored as `legal_compliance.md`.

```
# Legal and Regulatory Compliance

This document explains the main legal and cybersecurity rules for our accounting
↪  firm in Belgium. It helps meet **CyFun ID.GV-3** (governance) and follows
↪  **NIS2** rules, active since October 2024.
```

```
4
5  All team members (managers, accountants, secretary) must know these rules. The
   ↪  **Directive Manager** ensures they are followed.
6
7  ---
8
9  ## 1. Main Legal Rules
10
11 ### 1.1 General Data Protection Regulation (GDPR)
12 - We handle clients personal and financial data.
13 - Data must be safe (strong passwords, encryption, secure backups).
14 - Clients can ask to see, change, or delete their data.
15 - If there's a data breach, we must tell the Belgian Data Protection Authority
   ↪  (APD) within **72 hours**.
16
17 ### 1.2 Belgian Accounting Law
18 - Accounting documents must be kept for **at least 7 years**.
19 - Documents must be ready for tax authorities if requested.
20 - Digital storage is allowed but must be secure and organized.
21
22 ### 1.3 Confidentiality Rules
23 - As accountants, we must keep client data private.
24 - Only authorized staff can access sensitive files.
25 - Personal USB drives or private email accounts are not allowed.
26
27 ### 1.4 NIS2 Directive
28
29 - **Our firm may be an ''important entity'' or a supplier to critical sectors
   ↪  (e.g., healthcare, finance).**
30 - **CyFun level 1 meets basic NIS2 needs, but extra steps (e.g., audits,
   ↪  documentation) may be needed for full compliance.**
31
32 ---
33
34 ## 2. Our Actions
35
36 | Rule                              | How We Follow It
   ↪  | Who's Responsible | SMART Goal
   ↪  |
37 |-----------------------------------|-----------------------------------------------------------------
38 | GDPR - Protect Client Data        | Strong passwords, limited access, encrypted
   ↪  backups via **JANUS**              | Directive Manager | **Test 100% of
   ↪  backups monthly, checked with JANUS reports.**          |
39 | GDPR - Client Data Requests       | Clients can email to update or delete data
   ↪  | Secretary         | **Handle 95% of requests in 5 working days, checked
   ↪  every 3 months.**  |
40 | GDPR - Report Data Breach         | Process in place, reviewed yearly
   ↪  | Directive Manager | **Report 100% of breaches to APD in 72 hours, checked
   ↪  yearly.**        |
41 | Accounting - 7-Year Storage       | Store documents on TO_BE_FILLED (Nextcloud),
   ↪  **set up by JANUS**              | Accountant        | **Test 95% of
   ↪  documents for access every 6 months.**                  |
42 | Confidentiality                   | Work accounts only, monitored via **JANUS
   ↪  SIEM**                                | All Employees     | **No cases of
   ↪  unauthorized devices, checked every 3 months.**         |
```

```
| **NIS2 - Supplier Security**   | **Check suppliers yearly (e.g., Nextcloud GDPR/NIS2 compliance)**              | Directive Manager | **Evaluate 100% of suppliers by December each year.**                  |

---

## 3. Review and Updates

This document must be checked every year or when new rules appear. The **Directive Manager** handles updates. Changes are recorded below.

---

## Review History

| Date             | Reviewed By       | Comment                          |
|------------------|-------------------|----------------------------------|
| TO_BE_FILLED     | Directive Manager | Initial version for CyFun and NIS2 |

---

**Last Review**: TO_BE_FILLED
```

C.4 Cybersecurity Risk strategy

The following document supports CyFun sub-function `ID.GV-4`. It describe the risk strategy for the company.

```
# Cybersecurity Risk Management Strategy

This document defines our organization's strategy to manage cybersecurity risks. It aligns with CyFun ID.GV-4 requirements and supports continuous improvement.

## 1. Objectives

- Protect our critical assets (client data, credentials, accounting systems).
- Integrate risk management into company governance.
- Make decisions based on real risks and business impact.
- Allocate the right resources (time, budget, people) to protect what matters.

## 2. Governance Integration

Cybersecurity risk management is part of our overall risk management process.

- The **Directive Manager** is responsible for approving the strategy.
- All staff (accountants, secretary, external IT) participate in risk identification.
- Risks are reviewed during management meetings and documented in the Risk Register.

```

```
## 3. Risk Identification and Evaluation

We use the CyFun framework to identify risks across these areas:

- Infrastructure and physical assets (ID.AM-1)
- Software and platforms (ID.AM-2)
- Information and data flows (ID.AM-3)
- Vulnerabilities (ID.RA-1)
- Threats and risk scenarios (ID.RA-5)

Risks are evaluated using a simple matrix (impact × probability). High risks are treated first.

## 4. Risk Treatment and Action Plans

For each identified risk:

- We check what protections are already in place.
- We define new actions when needed (e.g. training, backup, patching).
- We assign owners and deadlines.

All actions are recorded in our **Risk Management Plan** (see `risk_management_plan.md`).

## 5. Resource Planning

The Directive Manager ensures that we have the necessary resources to apply protections:

- Human resources: clear roles and responsibilities
- Financial resources: small yearly budget for training, backup, and support
- Technical tools: use of free/open-source tools where possible (e.g. Trivy scanner)

## 6. Review Process

We review this strategy:

- Once per year
- After any serious incident or major change (new client, new software, etc.)

Each review is documented with date and summary of changes.

## 7. Improvement Path

This strategy will evolve. We aim to:

- Add automation where possible
- Include employees more in the risk process
- Align progressively with ISO 27001 and NIS2 expectations

---

```

```
**Approved by:** Directive Manager
**Last update:** TO_BE_FILLED

```

C.5 Cybersecurity Risk Management Plan

The following document supports CyFun sub-function `ID.GV-4` and clause 6 of ISO/IEC 27001:2022. It helps the firm assess and manage cybersecurity risks using a simple template stored as `risk_management_plan.md`.

```
# Cybersecurity Risk Management Plan

This document helps our accounting firm identify, evaluate, and treat cybersecurity risks, fulfilling **CyFun ID.GV-4** and ISO27001 clause6 requirements. It is intended as a living template: update the sections marked **TO_BE_FILLED** with details that match your business.

---

## 1. Purpose

Provide a simple, repeatable method to:

1. List and describe cybersecurity risks that could impact the firm or its clients.
2. Prioritise them by **Impact** and **Probability**.
3. Decide and track risk treatment actions.
4. Review the plan at least once per year or after major changes.

## 2. Scope

* All IT systems and cloud services used for accounting work.
* All data processed for and by clients (financial, personal, HR).
* Staff: directive manager, accountants, secretary, and external IT provider.

## 3. Roles & Responsibilities

| Role                  | Responsibilities                                |
| --------------------- | ----------------------------------------------- |
| **Directive Manager** | Owns this plan, approves risk treatments.       |
| **Accountant(s)**     | Report new risks, apply daily controls.         |
| **Secretary**         | Support data handling and backups.              |
| **External IT**       | Advise on technical controls, implement fixes. |

## 4. Risk Assessment Method

We rate **Impact** and **Probability** on a 1-to-3 scale (Low=1, Medium=2, High=3).

| Impact × Probability | 1 (Low)  | 2 (Medium) | 3 (High) |
| -------------------- | -------- | ---------- | -------- |
```

```
| **1 (Low)**          | Accept   | Monitor    | Mitigate |
| **2 (Medium)**       | Monitor  | Mitigate   | Reduce   |
| **3 (High)**         | Escalate | Reduce     | Critical |

**Risk Score = Impact × Probability**  → ranges from **1** to **9**.

## 5. Risk Register (Template)

| # | Risk Description                    | Impact | Prob. | Score | Existing Controls                    | Treatment Plan (TO_BE_FILLED)   | Owner (TO_BE_FILLED)   | Deadline (TO_BE_FILLED)   |
| - | ----------------------------------- | ------ | ----- | ----- | ------------------------------------ | ------------------------------- | ---------------------- | ------------------------- |
| 1 | Ransomware encrypts accounting files | 3      | 2     | 6     | Daily backups;                       | e.g.     off-site backup        |                        |                           |
|   |                                     |        |       |       | Directive Manager      | TO_BE_FILLED                 |                        |                           |
| 2 | Phishing email steals credentials    | 3      | 2     | 6     | Bitwarden; yearly phishing drill     | Accountant                      | TO_BE_FILLED           |                           |
| 3 | SaaS outage (BOB, Sage)              | 2      | 2     | 4     | **TO_BE_FILLED**                     | create local fallback process   |                        |                           |
|   |                                     |        |       |       | Secretary              | TO_BE_FILL                   |                        |                           |
| 4 | **TO_BE_FILLED**                     | 1-3    | 1-3   | calc  | **TO_BE_FILLED**                     | **TO_BE_FILLED**                |                        |                           |
|   |                                     |        |       |       | **TO_BE_FILLED**       | **TO_BE_FILLED**             |                        |                           |
| ... | **TO_BE_FILLED**                     |        |       |       |                                      |                                 |                        |                           |

Add one row per identified risk. Use the score to sort the table from High to Low.

## 6. Resource Allocation

After prioritising, the directive manager decides which controls need budget or external help. Example:

* Off-site backup subscription → € 200 / year.
* Phishing awareness session → 2 hours of external IT time.

## 7. Review & Update

* **Frequency:** once per year **and** after any major change (new software, incident, regulation).
* Update the **Risk Register**, **Treatment Plan**, and **Resource Allocation**.
* Record changes in the history table below.

| Date           | Reviewer          | Summary of Changes       |
| -------------- | ----------------- | ------------------------ |
| TO_BE_FILLED   | Directive Manager | Initial template created |

---
```

```

**Last Review:** TO_BE_FILLED
```

C.6 Vulnerability Collection Playbook (collect_vulnerabilities.yml)

The following Ansible playbook automates the detection and documentation of vulnerabilities using Trivy, supporting compliance with CyFun sub-function `ID.RA-1`. It scans all active Docker containers and consolidates vulnerability data into a structured YAML report.

```yaml
###############################################################################
# collect_vulnerabilities.yml - CyFun  ID.RA-1
###############################################################################
- name: Collect vulnerabilities for CyFun ID.RA-1
  hosts: user_domain                  #  ton inventaire
  connection: local
  gather_facts: no
  remote_user: "{{ user_var }}"

  vars_files:
    - ../../../user/local_or_remote.yml

  vars:
    trivy_image:       aquasec/trivy:0.49.1
    output_dir:        trivy-reports
    output_dir_full:  "{{ playbook_dir }}/{{ output_dir }}"
    severities:        HIGH,CRITICAL
    vuln_file:         vulnerabilities.yml
    vuln_file_full:   "{{ playbook_dir }}/{{ vuln_file }}"
    dl_timeout:        300          # s
    trivy_mem:         512m         # limite mémoire par scan

###############################################################################
# TASKS
###############################################################################
  tasks:

  #  0. Purge
  - name: Remove previous vulnerability file
    ansible.builtin.file:
      path: "{{ vuln_file_full }}"
      state: absent

  - name: Delete old Trivy reports directory
    ansible.builtin.file:
      path: "{{ output_dir_full }}"
      state: absent

  - name: Re-create empty report directory
    ansible.builtin.file:
      path: "{{ output_dir_full }}"
```

```
42      state: directory
43      mode: "0755"
44
45    #  1. Inventaire & pull des images
46    - name: Gather tagged docker images
47      community.docker.docker_host_info:
48        images: yes
49      become: true
50      register: docker_host
51      no_log: true
52
53    - name: Extract unique tagged images
54      ansible.builtin.set_fact:
55        images_to_scan: >-
56          {{
57            docker_host.images | default([])
58            | map(attribute='RepoTags') | flatten
59            | reject('equalto', None) | unique
60          }}
61
62    - name: Pull images locally
63      community.docker.docker_image:
64        name: "{{ item }}"
65        source: pull
66        timeout: "{{ dl_timeout }}"
67      loop: "{{ images_to_scan }}"
68      loop_control: { label: "{{ item }}" }
69      become: true
70      throttle: 1
71
72    #  2. Scan Trivy (conteneur)
73    - name: Ensure shared Trivy cache directory exists
74      ansible.builtin.file:
75        path: "{{ playbook_dir }}/trivy-cache"
76        state: directory
77        mode: "0755"
78
79    - name: Prime Trivy vulnerability database (one shot)
80      become: true
81      community.docker.docker_container:
82        name: trivy_update
83        image: "{{ trivy_image }}"
84        command: [ image, --download-db-only, --cache-dir=/cache, alpine:3.19 ]
85        volumes:
86          - "{{ playbook_dir }}/trivy-cache:/cache"
87        state: started
88        detach: false
89        auto_remove: true
90        timeout: 600
91      when: lookup('ansible.builtin.fileglob',
92                   playbook_dir + '/trivy-cache/db/*',
93                   errors='ignore') | length == 0
94
```

```
95  - name: Scan each image with Trivy
96    community.docker.docker_container:
97      name: "trivy_scan_{{ item | regex_replace('[^a-zA-Z0-9]', '_') }}"
98      image: "{{ trivy_image }}"
99      command:
100       - image
101       - --severity={{ severities }}
102       - --skip-update
103       - --cache-dir=/cache
104       - --timeout={{ dl_timeout }}s
105       - --format=json
106       - --scanners
107       - vuln
108       - --output=/reports/{{ item | basename | regex_replace('[/:]', '_')
          ↪  }}.json
109       - "{{ item }}"
110     volumes:
111       - "{{ output_dir_full }}:/reports"
112       - "{{ playbook_dir }}/trivy-cache:/cache"
113     state: started
114     detach: false
115     memory: "{{ trivy_mem }}"
116     auto_remove: true
117     timeout: "{{ dl_timeout | int + 30 }}"
118   loop: "{{ images_to_scan }}"
119   loop_control: { label: "{{ item }}" }
120   become: true
121   throttle: 1
122   ignore_errors: true
123   failed_when: false
124
125 #  3. Collecte des rapports
126 - name: Find Trivy JSON outputs
127   ansible.builtin.find:
128     paths: "{{ output_dir_full }}"
129     patterns: "*.json"
130   register: trivy_reports
131
132 - meta: end_play
133   when: trivy_reports.matched | int == 0
134
135 #  3-4. Consolidation & YAML (fait côté contrôleur)
136 - name: Consolidate Trivy reports and write YAML (memory-safe)
137   delegate_to: localhost
138   run_once: true
139   shell: |
140     python3 - <<'PY'
141     import json, yaml, glob, os, datetime
142     out_dir  = os.environ['OUT_DIR']
143     out_file = os.environ['OUT_FILE']
144
145     vulns = {}
146     for jf in glob.glob(os.path.join(out_dir, '*.json')):
```

```
        with open(jf) as f:
            data = json.load(f)
        img = data.get('ArtifactName')
        for res in data.get('Results', []):
            for v in (res.get('Vulnerabilities') or []):
                vid = v['VulnerabilityID']
                entry = vulns.setdefault(vid, {
                    'severity': v.get('Severity', 'UNKNOWN'),
                    'title':    v.get('Title', ''),
                    'cwe':      v.get('CweIDs', []),
                    'images':   []
                })
                if img not in entry['images']:
                    entry['images'].append(img)

    final = {
        'collected_on': datetime.date.today().isoformat(),
        'scanner':      'trivy_container',
        'unique_cve_count': len(vulns),
        'vulnerabilities': [
            {'VulnerabilityID': k, **v} for k, v in vulns.items()
        ]
    }
    yaml.safe_dump(final, open(out_file, 'w'), sort_keys=False)
    PY
  environment:
    OUT_DIR:  "{{ output_dir_full }}"
    OUT_FILE: "{{ vuln_file_full }}"
  no_log: true          # ne rien afficher
  changed_when: true

#  5. Message final
- name: Completion message (summary only)
  ansible.builtin.debug:
    msg: >-
       Scan terminé : {{ trivy_reports.matched }} images,
      {{ lookup('ansible.builtin.file', vuln_file_full)
         | from_yaml
         | json_query('unique_cve_count') }} CVE uniques.

```

C.7 Manual Risk Analysis File (manual-risks.yml)

The following file documents manually defined risk scenarios in accordance with CyFun sub-function `ID.RA-5`. Each entry combines an asset, a threat, a vulnerability, and a risk analysis that includes impacts and likelihood. The file is stored as `manual-risks.yml`.

```
# manual-risks.yml - Risk analysis for CyFun ID.RA-5
file_type: manual-risks
description: >
  This file documents risk scenarios for critical assets, derived from the
  ↪ combination
```

```
  of identified threats, known vulnerabilities, and their business impact.

  It ensures compliance with:
    - ID.RA-5 (Risk is based on threats, vulnerabilities, and impact)

generated_on: TO_BE_FILLED   # REQUIRED - Date of generation or last review

risks:

  - id: risk-001
    # REQUIRED - Unique ID
    asset: bitwarden
    # REQUIRED - Affected asset or service
    threat_id: threat-001
    # REQUIRED - Reference to threat ID (from manual-threats.yml)
    vulnerability_id: CVE-2024-2961
    # REQUIRED - Associated vulnerability (Trivy or manual)
    description: >
      This CVE could be exploited remotely. Combined with phishing,
      it could compromise stored credentials.
    confidentiality_impact: high
    # REQUIRED - Impact on confidentiality
    integrity_impact: medium
    # REQUIRED - Impact on integrity
    availability_impact: low
    # REQUIRED - Impact on availability
    likelihood: high
    # REQUIRED - Likelihood of exploitation
    risk_level: high
    # REQUIRED - Final assessed risk (low/medium/high)
    mitigation: Patch Bitwarden regularly and enforce 2FA
    # REQUIRED - Suggested or applied mitigation
    last_review: TO_BE_FILLED
    # REQUIRED - Last review date (format: YYYY-MM-DD)
```

C.8 Manual Threat Scenarios File (manual-threats.yml)

The following YAML file contains manually documented threat scenarios used to support CyFun sub-functions `ID.RA-1` and `ID.RA-5`. Each entry links to vulnerabilities and impacted assets. The file is stored as `manual-threats.yml`.

```
# manual-threats.yml - Minimal template for CyFun ID.RA-1 and ID.RA-5
file_type: manual-threats

description: >
  This file documents manually identified threat scenarios for compliance with:
    - ID.RA-1 (document vulnerabilities)
    - ID.RA-5 (derive risks from threats, vulnerabilities, and impacts)

last_review: TO_BE_FILLED   # REQUIRED - last review date (YYYY-MM-DD)

```

```yaml
threats:

  - id: TH-001
    # REQUIRED - unique identifier
    description: Remote attacker exploits a known vulnerability (CVE-2024-2961)
      in the Jitsi Web container.      # REQUIRED - description
    type: vulnerability_exploitation
      # REQUIRED - e.g., misconfig, phishing, ransomware
    source: external_attacker
      # OPTIONAL - internal, 3rd_party, etc.
    related_asset: jitsi/web:stable-9220-1
      # OPTIONAL - impacted container or host
    related_vulnerability: CVE-2024-2961
      # OPTIONAL - known CVE or weakness
    cwe: CWE-787
      # OPTIONAL - CWE ID if applicable
    likelihood: high
      # REQUIRED - low / medium / high
    impact: high
      # REQUIRED - effect on CIA triad
    severity: high
      # REQUIRED - overall risk level
    mitigation: Keep Jitsi images updated weekly and restrict network exposure.
      # REQUIRED - existing or planned control
    last_review: TO_BE_FILLED
      # REQUIRED - date this entry was last updated

```

Appendix D
Documentation

D.1 Digital format documentation

All digital resources related to this thesis are hosted on a dedicated GitHub repository: `https://github.com/aquerinj/Thesis-Querinjean/`

This repository contains:

- A `.zip` archive of the full JANUS infrastructure project code (exported from the original private GitLab repository).
- A video demonstration showing how to launch the infrastructure and collect security inventory data for CyFun compliance.
- Screenshots of key outputs and visual results of automated processes.

Recommended viewing order

1. Start by reading the `README.md` file in the GitHub repository to understand prerequisites and the structure.
2. Watch the demonstration video to visualize the automated deployment and inventory collection process.
3. Explore the folders to inspect:
    - Infrastructure code: `/janus/`
    - Ansible tasks: `/domain/cyfun/identify/`
    - Output files and inventory data: `/output/` folders

How to launch the project (Local test)

To run the project locally, install the following dependencies, virtualbox and vagrant (Example on a Debian-based system):

```
sudo apt install virtualbox vagrant
```

Then navigate to the user domain folder:

```
cd janus/domain/user
```

To launch the environment with CyFun compliance features:

- On Linux/macOS:

  ```
  USE_CYFUN=true vagrant destroy --force && vagrant box update && vagrant up
  ```

- On Windows (CMD):

  ```
  set "USE_CYFUN=true" && vagrant destroy --force && vagrant box update && vagrant up
  ```