NOVA SCHOOL OF SCIENCE & TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE

THIAGO ARAUJO MONTEIRO

BSc in Computer Science and Engineering

CREATING A THREAT MODELING PROTOCOL FOR NON-HIERARCHICAL ORGANIZATIONS

MASTER IN COMPUTER SCIENCE AND ENGINEERING

NOVA University Lisbon

Draft: *April 7, 2025*

NOVA SCHOOL OF
SCIENCE & TECHNOLOGY

DEPARTMENT OF
COMPUTER SCIENCE

CREATING A THREAT MODELING PROTOCOL FOR NON-HIERARCHICAL ORGANIZATIONS

THIAGO ARAUJO MONTEIRO

BSc in Computer Science and Engineering

Adviser: Kevin Gallagher
Associate Professor, NOVA University Lisbon

MASTER IN COMPUTER SCIENCE AND ENGINEERING

NOVA University Lisbon
Draft: *April 7, 2025*

Abstract

This research explores the development of a threat modeling protocol specifically designed for non-hierarchical organizations. Grounded in analyses of distributed governance frameworks and security methodologies, the study presents an innovative approach that treats horizontality as a strategic asset. Leveraging tools such as attack trees, STRIDE, PASTA, and COLBAC, the protocol aims to integrate cybersecurity with democratic participation, addressing the unique challenges of decentralized structures. This work fills a gap in the literature by providing practical guidelines to identify, mitigate, and prevent threats in contexts where distributed trust and collaboration are essential.

Keywords: threat modeling, horizontal organizations, distributed governance, collaborative security, decentralized trust

Resumo

A pesquisa explora a criação de um protocolo de modelagem de ameaças projetado especificamente para organizações não-hierárquicas. Com base em análises de frameworks de governança distribuída e metodologias de segurança, o estudo desenvolve uma abordagem inovadora que considera a horizontalidade como um ativo estratégico. Utilizando ferramentas como árvores de ataque, STRIDE, PASTA e COLBAC, o protocolo busca integrar segurança cibernética e participação democrática, abordando desafios únicos de estruturas descentralizadas. Este trabalho contribui para preencher lacunas na literatura ao oferecer diretrizes práticas para identificar, mitigar e prevenir ameaças em contextos onde a confiança distribuída e a colaboração são fundamentais.

Palavras-chave: modelagem de ameaças, organizações horizontais, governança distribuída, segurança colaborativa, confiança descentralizada

Contents

Acronyms

ABC Asset-Based Cryptocurrency *(pp. 18–21, 49)*

COLBAC Collective based access control system *(pp. 10, 17, 18, 20, 21, 49)*
CoReTM Collaborative and Remote Threat Modeling *(p. 17)*

DAC Discretionary Access Control *(p. 17)*
DFDs Diagramas de Fluxo de Dados *(p. 12)*
DREAD Damage, Reproducibility, Exploitability, Affected users, Discoverability *(p. 13)*

hTMM Hybrid Threat Modeling Method *(p. 17)*

MAC Mandatory Access Control *(p. 17)*

PASTA Process for Attack Simulation and Threat Analysis *(pp. 1, 13–15, 20)*
PGP Pretty Good Privacy *(pp. 19, 20)*
PnGs Personae Non Gratae *(pp. 16, 17)*
PTM Participatory Threat Modeling *(pp. 17, 21)*

RBAC Role-Based Access Control *(p. 17)*

STRIDE Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, Elevation of privilege *(pp. 1, 2, 7, 8, 12–14, 17, 18, 20, 21, 47, 49)*

WoT Web of Trust *(pp. 19, 20)*

1

Introduction

1.1 Threat Modeling: Relevance and Challenges

Threat modeling is an essential discipline in information security, whose main function is to identify, classify and mitigate vulnerabilities in technological systems before they can be exploited by adversaries [38, 51]. In a context where systems become increasingly complex and integrated, threat modeling stands out as a critical tool for anticipating risks and establishing effective security measures [45, 36].

Traditional models such as Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, Elevation of privilege (STRIDE), attack trees, and iterative methodologies such as Process for Attack Simulation and Threat Analysis (PASTA) have been widely applied in hierarchical contexts [34, 35, 47]. These approaches focus on linear, hierarchical data flows, but face significant challenges when applied to horizontal organizations, where the distribution of power and responsibilities fundamentally alters risk dynamics [40, 9].

Horizontal organizational structures, characterized by the absence of a formal hierarchy, face particular challenges in threat modeling [9]. The lack of centralization can make it difficult to implement role-based access controls (RBAC) and other systems that rely on hierarchical structures [9]. Furthermore, the temporary centralization of organizational secrets, such as passwords or encryption keys, often leads to conflicts known as "password wars" during leadership transitions [19].

Additionally, digital tools often promote implicit centralization of power, creating the phenomenon of the "digital vanguard," where individuals control critical resources such as communication platforms [10]. This is exacerbated by attacks specific to horizontal systems, such as identity spoofing (Sybil attacks) and quorum manipulation, which exploit the dependency on participatory processes [46, 7].

The challenges highlighted indicate the need for adaptations in threat modeling methods for horizontal contexts [9]. In addition to technical limitations, such as the difficulty of integrating collaborative cryptography [1], the need for participatory tools that respect democratic dynamics and promote resilience is also highlighted [6].

1.2 Horizontal Security in Times of Interconnection

In today's interconnected world, horizontal organizations challenge the assumption that security depends on a clear chain of command [11, 42]. The absence of a formal hierarchy can become a strategic asset by hindering centralized attacks and enabling a reconfiguration of trust management, promoting organizational resilience [42, 9]. In distributed trust systems, such as those used in blockchain-based organizations, security is promoted by collaborative mechanisms that replace formal leaders with participatory processes and solutions oriented towards transparency and consensus [33, 1].

Traditional threat analysis methodologies such as STRIDE and attack trees provide valuable insights but face limitations in decentralized environments, highlighting the need for approaches better suited to the specificities of horizontal structures [14, 36]. Less hierarchical contexts require adapted approaches that understand the complexity of horizontal trust and the potential associated risks [9].

In this sense, technologies such as collaborative cryptography [9, 1] and threat modeling approaches that adopt the global perspective of the organization can promote a more realistic understanding of security in decentralized structures. Horizontality, often seen as a challenge, should be explored as a strategic asset capable of diluting single points of failure and strengthening organizational resilience [42].

1.3 Organizational Governance: A Historical Perspective

Organizational governance reflects the social, economic, and technological structures of each era. From the earliest human groups to the complex organizations of contemporary times, the ways of organizing power and decision-making have been shaped to respond to specific contexts [11]. The hierarchical model, widely adopted, emerged as a solution to demands for control and efficiency. However, history also records experiments that challenged this logic, suggesting the possibility of new approaches to the management and coordination of activities [15, 50].

Even in systems considered pioneers in horizontality, such as Athenian democracy, governance faced significant limitations related to inclusion and practical applicability, highlighting weaknesses in the operationalization of equal participation [2]. As the Industrial Revolution progressed, hierarchical centralization intensified to cope with organizational growth and complexity [50]. Additionally, experiences such as cooperatives and the 19th century labor movement outlined alternatives to absolute centralization, while modern technologies offer decentralized structures that challenge traditional paradigms of control [15, 42].

Innovations such as blockchain open up new possibilities for decentralization, yet face challenges in equitable distribution of power and resources, as evidenced by the concentration of miners on public networks [17].

These historical and technological evolutions not only shape governance structures, but also introduce unique challenges in threat modeling [49, 41]. Critical analysis of these attempts allows us to identify vulnerabilities and strengths that underpin the construction of security protocols in horizontal organizations [9].

1.4 Security Protocol for Non-Hierarchical Organizations

This research proposes a security protocol that integrates horizontality as a strategic element, going beyond the simple adaptation of traditional methodologies [9]. The main objective is to demonstrate how decentralization, when structured in a way that is coherent with organizational principles, can strengthen resilience against complex threats, mitigating single points of failure and distributing responsibilities equitably [38]. The protocol aims to balance operational efficiency and democratic participation, ensuring that security measures do not compromise decision-making agility or the inclusion of members in critical processes [33]. To this end, it relies on collaborative approaches, such as cryptography adapted to horizontal contexts [9], and on threat modeling methodologies that consider participatory dynamics [43].

The integration between security and governance is addressed through guidelines that harmonize technical requirements with organizational principles [43]. The protocol foresees the application of transparent and auditable consensus mechanisms, inspired by distributed reputation models [33], to validate access policies and mitigate risks such as Sybil attacks [46]. In addition, it incorporates modular structures that allow adaptation to different levels of horizontality, from fully decentralized networks to organizations with more hierarchical structures [9]. This flexibility is essential to respond to dynamic threats without compromising the autonomy of members and to cover the largest number of organizations.

To strengthen resilience, the protocol combines technical and social layers: cryptographic techniques protect against external threats, while radical transparency structures and periodic reviews by rotating committees prevent internal fraud [42]. The traceability of decisions via immutable records ensures that vulnerabilities are identified and corrected collaboratively, in line with studies on failures in distributed systems [33]. By incorporating lessons from historical cases and methodological innovations, the protocol offers a practical framework for organizations that seek security without giving up their horizontal identity, paving the way for detailed analyses in subsequent chapters.

1.5 Defining the Scope of Research

The diversity of horizontal organizations ranges from informal collectives to complex digital networks, each with its own dynamics [42]. To ensure analytical focus, this study is limited to structures that operate under strict principles of horizontality, characterized

by: (1) absence of formal hierarchies or permanent centralization of power; (2) decision-making processes based on consensus or broad participation; and (3) explicit mechanisms for distributing responsibilities and resources [9]. This delimitation excludes hybrid or partially decentralized models, where the coexistence of hierarchical and horizontal structures introduces additional variables that make it difficult to evaluate the proposed protocol in isolation [11].

The choice to analyze organizations such as worker cooperatives and community networks is justified by their empirical relevance: these models have robust documentation on operational challenges [15], in addition to explicitly adopting principles of self-management and radical transparency [42]. These characteristics allow testing the protocol in contexts where security depends directly on collective coordination, without intermediaries or central authorities [38].

Although digital platforms and decentralized social networks [18] represent equally relevant cases, their dynamic nature and dependence on heterogeneous technical infrastructures would require methodological adaptations beyond the current scope. Future studies could explore these variations, using the protocol developed here as a basis for comparative analyses in less controlled environments.

1.6 Expected Contributions

The research will deliver a threat modeling protocol specific to horizontal organizations, integrating the principles of distributed governance, collective participation, and transparency [9]. This protocol will be designed to identify, assess, and mitigate threats in decentralized structures, providing practical guidelines adapted to the particularities of these organizations. In addition, it will be accompanied by an evaluation method to validate its effectiveness and its application in real cases.

The protocol is expected to demonstrate how horizontality can be a strategic asset, reinforcing organizational security and resilience in the face of complex threats. In addition to contributing to theoretical advancement on security in decentralized structures, the research seeks to offer a practical solution that strengthens autonomy and promotes the integration between security and democratic governance.

1.7 Structure of the Thesis

After the introduction, the Background chapter presents the fundamentals of threat modeling and security in horizontal structures. The Related Work chapter analyzes previous studies, identifying relevant gaps and opportunities.

The Design chapter outlines the proposed protocol, its methodologies, and evaluation criteria. The Work Plan chapter describes the research steps and schedule, ensuring

the structured and viable execution of the project. Finally, the Conclusion chapter summarizes the results, discusses limitations, and proposes future directions, highlighting horizontality as a strategic asset for organizational security.

The first chapter presented the context that support this research, identifying challenges faced by non-hierarchical organizations in the area of digital security and threat modeling. The structural particularities of these organizations were described, highlighting the need for specific methods for risk analysis and mitigation. Having defined the objectives, intended contributions and structure of this research, Chapter 2 will address the essential concepts that underpin the study, including threat modeling, the conceptual differences between horizontal organizations and organizations without explicit leadership, and the role of democratic centralism as an organizational principle.

2

Background

2.1 Foundations of Threat Modeling

Threat modeling is a central component of cybersecurity, as it allows the identification of valuable assets, analysis of potential attack vectors, and establishment of controls capable of mitigating risks. This practice goes beyond technical factors, incorporating organizational and human elements that shape security in diverse contexts, especially in horizontal structures, where internal processes and trust relationships become even more critical due to the absence of formal hierarchies [9]. In a scenario of rapid technological evolution and constant diversification of threats, a broad and flexible approach gains relevance, meeting the particularities of changing contexts [38].

Studies such as [28], [29], and [45] demonstrate the need for structured, yet adaptable, methods to keep up with changing environments. Adopting the adversary perspective [25] is crucial to anticipate complex scenarios and strengthen resilience in decentralized environments, where the multiplicity of actors and the distribution of power require a holistic analysis of threats. In this type of context, threat modeling needs to reflect the distribution of power and the multiplicity of actors, including potential internal, external and hybrid threats [36].

In addition, Microsoft's experience, documented in [37], emphasizes the importance of involving diverse stakeholders and applying collaborative tools. These elements become crucial when decision-making is democratic or decentralized, since identifying and mitigating risks requires collective engagement and strategic flexibility, connecting the threat modeling exercise to organizational dynamics [43].

2.1.1 Conceptual Definitions

Threat modeling can be understood as a systematic protection effort that considers both technical vulnerabilities and social and organizational factors. By adopting a holistic view, as suggested by [28] and [29], security analysis is not restricted to infrastructure, but incorporates internal practices, information flows, and the organization's culture. In non-hierarchical contexts, the absence of clear lines of authority and the participatory

nature make an analysis that considers the distribution of responsibilities and the reliance on collective mitigation mechanisms essential [9].

In turn, [45] emphasizes that there is no single solution for threat modeling. In this sense, threat modeling becomes an iterative process, adapting to structural changes and incorporating innovations such as participatory decision-making tools and collaborative security practices [6, 43].

2.1.2 Main Methodologies

Widely discussed methodologies such as Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, Elevation of privilege (STRIDE), attack trees, and scenario-based frameworks [40] provide a tested starting point, but they often fail to capture the complexity of horizontal structures.

The documentation [36] provides an overview of existing methods, cautioning that the effectiveness of each approach depends on the context. For example, STRIDE and attack trees are useful for identifying straightforward attack vectors, but the lack of formal hierarchies intensifies the need to explore complex scenarios such as insider threats coupled with external attacks, as well as flaws in distributed authentication, consensus, and governance mechanisms, as suggested in [9, 35, 20].

Integrating diverse approaches, such as collaborative cryptography practices [1] and hybrid approaches [48], enables horizontal organizations to identify less obvious risk patterns and strengthen their collective resilience. This integration becomes crucial in distributed structures, where the absence of a 'center' turns security into a collective effort, and resilience emerges from the continuous interaction between members, systems and decentralized governance mechanisms [33].

2.2 Taxonomy of Organizational Structures

Understanding the relationship between organizational form and security is essential to adjust threat modeling to the reality of each institution [11]. While hierarchical structures rely on central decision points for control and coordination, these same points can become critical vulnerabilities [38]. Horizontal or cooperative organizations can disperse vulnerabilities and increase resilience through decentralized governance, although they can also create multiple entry points that require collaborative control [9]. Analysis of this taxonomy, as [15, 21], provides a basis for identifying how the distribution of power in different organizational forms affects the effectiveness of security measures, including the ability to respond to internal and external threats.

2.2.1 Traditional Hierarchical Structures

Hierarchical organizations have clear lines of authority, which facilitates control but can concentrate vulnerabilities in critical areas [34]. These organizations are characterized by a

well-defined chain of command, where decisions flow from the top down. Classic examples include large multinational corporations, where the board of directors establishes policies that are implemented by layers of managers, supervisors, and employees. On the other hand, in small businesses, such as family offices, the hierarchy may be less formal but still based on a clear, centralized command structure [15].

In large organizations, such as banks or automotive manufacturers, hierarchy allows for efficient allocation of resources and tight control over operations. For example, IT departments in these environments often use security frameworks such as STRIDE for threat modeling, focusing on protecting critical assets and centralized access management [34, 51]. Centralization facilitates rapid incident response, but it also creates single points of failure, such as vulnerabilities in key servers or administrative credentials [49, 27].

In contrast, small businesses face different challenges. In these contexts, lack of resources may lead to fewer hierarchical layers, but decisions are still centered on a single owner or manager [50]. This reduces organizational complexity but increases reliance on specific individuals, making them prime targets for attacks [15]. Furthermore, the lack of dedicated security teams can limit the ability to implement sophisticated frameworks such as STRIDE, requiring more streamlined solutions.

The difference between large and small organizations also impacts threat modeling [15, 36]. In larger organizations, hierarchical structures allow for detailed segmentations to identify and mitigate risks at specific levels of the organization. However, this segmentation can lead to communication gaps between departments, making it difficult to implement integrated solutions [51]. On the other hand, smaller organizations have greater flexibility to quickly adapt their security strategies, although they often lack the resources to implement robust solutions [50].

Therefore, while hierarchical organizations offer advantages in terms of control and clarity, they also introduce specific challenges for threat modeling. These challenges vary significantly with the size and complexity of the organization, requiring adaptations to traditional frameworks to meet the specific needs of each type of hierarchy.

2.2.2 Horizontal Organizations

Horizontal organizations are distinguished by their rejection of traditional hierarchies, prioritizing distributed decision-making processes and equitable participation of all members. This model contrasts directly with hierarchical structures, which centralize power at higher levels, perpetuating inequalities in access to information and organizational control [11, 31].

Horizontality is both a tool and an objective in itself [9]. In Argentine social movements, as analyzed by Marina Sitrin, horizontality has emerged as an essential mechanism for establishing relationships based on trust and consensus, overcoming traditional forms of organization. Neighborhood assemblies and collectives of unemployed workers exemplify how horizontality can be applied to self-management and collective planning [42].

In the field of cybernetics, the COLBAC protocol demonstrates the relevance of horizontality in digital security systems, promoting collaborative access control that reduces the centralization of power. This model avoids the vulnerabilities created by the dependence on single owners of passwords or permissions, reinforcing the coherence between organizational practices and technological tools [9].

In addition, historical examples, such as Athenian democracy, illustrate that horizontal structures can be complemented by temporary centralization mechanisms in times of crisis, ensuring flexibility and efficiency without compromising the basic principles of distributed governance [2].

Despite the challenges, such as the risk of domination by more influential voices or the management of conflicts in collective spaces, horizontal organizations demonstrate that, with adequate mechanisms, it is possible to promote autonomy, inclusive participation and efficiency in decentralized structures [10].

2.2.3 Leaderless Organizational Models

The discourse of leaderless organizations hides an additional complexity, where the absence of a formal hierarchy does not necessarily imply genuine horizontality [10]. Critical studies highlight how these organizations often replicate veiled power dynamics and informal centralizations [10, 42].

Marina Sitrin, in her analysis of horizontal movements in Argentina, points out that although horizontality is declared as an objective, many movements face significant challenges in sustaining truly participatory practices. The lack of formal hierarchy often leads to informal power structures, where dominant voices assume leadership roles without oversight or clear collective responsibility [42].

In the digital context, movements such as Occupy Wall Street demonstrate that the absence of recognizable leadership does not eliminate internal conflicts [10]. Studies of social media teams in these movements reveal that account management, as on Twitter, was often marked by struggles for control, illustrating how power and influence can consolidate even in supposedly horizontal structures [10].

Furthermore, research on worker cooperatives in the United States indicates that these organizations, although often seen as non-hierarchical alternatives, tend to develop informal leaders who influence decisions in significant ways, challenging the narrative of absolute horizontality [50].

Technologies used by these organizations also carry political implications [49, 41]. Langdon Winner argues that technical artifacts can perpetuate existing power structures, even when employed in decentralized contexts [49]. For example, digital platforms, often designed for individual use, create challenges in building effective collective governance, exacerbating latent inequalities [49, 27].

These examples highlight that while the idea of formal leadershiplessness is appealing, its practical execution often results in informal forms of hierarchy [42, 10]. Thus,

the success of these organizations depends on the ability to identify and mitigate hidden power dynamics, promoting clear mechanisms of collective governance and mutual accountability that truly sustain the desired horizontality [9].

2.3 Democratic Centralism

In the midst of increasingly challenging scenarios in terms of organizational coordination, whether in the management of large corporations or in the maintenance of decentralized networks such as blockchain, the need arises to reconcile efficiency in decision-making with the active participation of all those involved [49]. Democratic centralism, formulated to meet the demands of revolutionary movements, maintains its relevance by proposing a dynamic balance between collective deliberation and centralized execution [31]. This approach has proven to be relevant both in political and social contexts and in contemporary technological scenarios [9].

2.3.1 Fundamental Principles and Theoretical Origins

Democratic centralism is based on the idea that opinion gathering and deliberation (democracy) must be harmonized with the ability to implement decisions in a unified manner (centralism) [44]. This system was originally conceived as a response to the need for organization in contexts of high structural complexity [52, 31].

Its initial formulation, associated with the Communist Party of the Soviet Union, inspired the adoption of the model by different groups around the world [31]. In China, for example, the practices of democratic centralism demonstrate remarkable resilience, being reconfigured as political and social situations change [44].

2.3.2 Contemporary Models of Application

The transposition of the precepts of democratic centralism to modern contexts has been observed in different organizational and technological structures [49, 9]. One example is the Collective based access control system (COLBAC) protocol, which adapts the bases of democratic centralism to a collaborative access control environment, enabling participatory decisions combined with cohesive implementation [9].

Trade unions and social organizations have also integrated principles of democratic centralism [8]. The General Confederation of Portuguese Workers, for example, combines collective deliberation and periods of strategic centralization to provide effective responses to organizational challenges [8].

2.3.3 Implications and Potentials for Governance

The influence of democratic centralism transcends the limits of political parties and can be extended to diverse scenarios that require both broad participation and efficient execution

[44, 5]. Horizontal or distributed organizations can use this model to reconcile the voice of their members with the need to make centralized decisions at critical moments [52, 44].

In this sense, the application of elements of democratic centralism in digital platforms highlights how historical concepts can be reappropriated to meet contemporary governance demands [9].

Chapter 2 presented the theoretical foundations related to threat modeling, addressing structured and flexible methods needed to deal with decentralized organizational contexts. Approaches that incorporate adversarial perspectives were discussed and emphasized the importance of considering social and organizational factors, in addition to technical vulnerabilities, highlighting especially the relevance of collaboration between different actors in the process. Building on these fundamental concepts, Chapter 3 will analyze related works, describing in detail traditional threat modeling approaches, such as STRIDE, and exploring their applications and limitations in decentralized and collaborative environments.

3

Related Work

The references discussed in this chapter were chosen based on an initial selection suggested by the advisor of this work, Professor Kevin Gallagher. As a starting point, the professor provided three fundamental articles related to his research area: COLBAC, ABCcrypto and PGP's Web of Trust. In particular, the work on COLBAC is directly linked to Professor Gallagher's academic production, offering a solid and highly relevant basis for the development of this research. Based on these initial references, traditional and emerging methods for threat modeling were explored, aiming to establish a comprehensive overview that would allow a critical analysis of the applicability of these methods to the specific context of non-hierarchical organizations.

3.1 Traditional Approaches to Threat Modeling

3.1.1 STRIDE

Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, Elevation of privilege (STRIDE), developed by Microsoft, is a systematic threat modeling methodology designed to identify potential vulnerabilities in software systems [34]. The acronym STRIDE stands for six main categories of threats [38]. Each of these categories reflects a specific violation of desired security properties such as authenticity, integrity, non-repudiation, confidentiality, availability, and authorization [34].

Applying STRIDE begins with the creation of Diagramas de Fluxo de Dados (DFDs) to map the movement of information within the system [12]. DFDs help identify elements such as external entities, processes, data flows, and data stores. For each element in the diagram, the six STRIDE threat categories are analyzed to identify potential vulnerabilities [38].

Each threat in STRIDE has a clear definition and practical examples to aid in identification and mitigation. For example:

1. **Spoofing:** Threats that involve falsification of the identity of users or processes, compromising authenticity.

2. **Tampering:** Manipulation of data in transit, in storage, or in memory, affecting integrity.
3. **Repudiation:** Scenarios where users deny actions performed, often due to the lack of adequate logging mechanisms.
4. **Information Disclosure:** Exposure of sensitive information to unauthorized parties, compromising confidentiality.
5. **Denial of Service:** Attacks that overload system resources, impairing availability.
6. **Elevation of Privilege:** Cases where a malicious actor obtains privileges above those authorized, compromising authorization.

The STRIDE methodology can be adapted to different contexts. For example, in cyber-physical systems, it is possible to assess threats related to hardware and software components, such as failures in synchronization or communication [20]. In addition, variants such as STRIDE-per-Element and STRIDE-per-Interaction offer more focused approaches to identify threats in specific elements or in interactions between components [38].

Although widely used, STRIDE has limitations. It relies heavily on analyst experience and may not capture emerging threats in decentralized systems or dynamic environments [13]. Therefore, it is often complemented with other frameworks, such as Damage, Reproducibility, Exploitability, Affected users, Discoverability (DREAD), to prioritize threats based on impact and probability [22].

In summary, STRIDE provides a solid foundation for threat identification, but its effective application requires integration with other methodologies and adaptations to meet the specific needs of modern, decentralized systems [12].

3.1.1.1 STRIDE Complementary Models

Several complementary models have been derived from or used in conjunction with STRIDE to improve its effectiveness and adaptability in different contexts [32]. Notable among these is the DREAD model, which complements STRIDE by providing a quantitative approach to threat prioritization [22]. DREAD uses five main categories: Damage Potential, Reproducibility, Exploitability, Affected Users, and Discoverability, allowing analysts to assign values and create scores to rank threats according to their severity [32, 22].

The combined use of STRIDE and DREAD can improve risk assessment in more complex systems. However, the inherent subjectivity in assigning values in DREAD can compromise the consistency of [22] analyses. To mitigate these limitations, some organizations have integrated STRIDE with more comprehensive frameworks, such as Process for Attack Simulation and Threat Analysis (PASTA), which takes an iterative approach to identifying and prioritizing threats [32].

Additionally, the use of attack trees has proven effective in complementing STRIDE, allowing teams to visually represent complex threat scenarios and identify multiple attack paths [24]. This integration is particularly useful in horizontal organizations, where the lack of centralization increases the need for collaborative threat mapping [38].

3.1.2 Attack Trees

Attack trees, introduced by Bruce Schneier [35], provide a hierarchical framework for threat modeling, where the attack objective is represented by the root node, and the sub-objectives and steps required to achieve it are arranged in child nodes. Each node can be detailed with logical operators such as AND and OR, representing conditions that must be met either together or alternatively [24].

A key advantage of attack trees is their ability to decompose complex threats into smaller, more manageable components, enabling systematic analysis [17]. This methodology makes it easier to identify multiple attack paths, allowing organizations to prioritize countermeasures based on metrics such as cost, impact, and likelihood [16].

Practical applications of attack trees include their use in wireless sensor networks to assess location privacy risks [16], as well as in energy theft detection in advanced metering infrastructures such as smart grids [17]. In both cases, the approach enables organizations to map specific threat scenarios and design effective countermeasures.

In addition, studies such as [24] highlight the reuse of subtrees to increase efficiency in complex systems. This practice allows shared elements across different scenarios to be modeled once and incorporated into future analyses, saving time and resources.

Although broadly applicable, attack trees present challenges related to the effort required for their initial construction and the complexity in large-scale systems [35, 17]. Collaboration between stakeholders, including technical and operational experts, is essential to ensure that threat representation is accurate and comprehensive [17].

Attack trees also stand out as complementary tools to methodologies such as STRIDE and can be used both to identify threats and to organize those already discovered [24, 51]. Furthermore, reusing existing trees, such as those focused on fraud or elections, saves time and provides a solid foundation for analysis [24]. Despite their versatility, effective use of trees depends on clear representations of AND/OR nodes and continuous evaluation to avoid oversights or gaps [38].

3.2 Emerging Methodologies

3.2.1 PASTA

PASTA is a risk-centric threat modeling methodology designed to integrate security throughout the software development lifecycle. Proposed by Tony UcedaVelez and Marco M. Morana [47], the framework consists of seven sequential stages that allow for in-depth and iterative analysis of threats and vulnerabilities.

The main goal of PASTA is to align security concerns with business objectives, ensuring that mitigation measures address both technical risks and organizational impacts. The methodology promotes a risk-oriented approach by integrating attack simulations to evaluate the effectiveness of proposed countermeasures [47].

1. **Definition of the Objectives (DO):** In this initial stage, the security requirements, risk profile, and potential business impacts are defined.

2. **Definition of the Technical Scope (DTS):** This stage details the technical aspects, such as users, software components, third-party infrastructure, and external dependencies.

3. **Application Decomposition and Analysis (ADA):** The application is broken down into basic functional elements to identify data flows, user types, and existing security controls.

4. **Threat Analysis (TA):** Identification of potential threats based on the analyzed elements and assets, considering the most likely attack vectors.

5. **Weakness and Vulnerability Analysis (WVA):** At this stage, threats are associated with specific vulnerabilities, evaluating the effectiveness of existing controls and identifying weaknesses.

6. **Attack Modeling and Simulation (AMS):** Performing simulations to determine the most likely attack paths, using attack trees and other models to explore risk scenarios.

7. **Risk Analysis and Management (RAM):** Identifying technical and business impacts, proposing measures to mitigate priority risks [47].

PASTA stands out for its flexibility and analytical depth, making it particularly effective in dynamic and distributed environments [51]. The methodology encourages collaboration between stakeholders from different areas, promoting a unified understanding of risks and organizational priorities [43]. In addition, the integration of attack simulations allows organizations to test the effectiveness of their security strategies under realistic conditions, improving their resilience against emerging threats [47].

One of the most relevant aspects of PASTA is its compatibility with horizontal organizations. The collaborative approach of the methodology, which involves multiple stakeholders at all stages of the process, is aligned with the principles of distributed governance [9]. In non-hierarchical structures, where responsibility for security is shared, PASTA offers a structured framework to identify and mitigate risks in a participatory and efficient way.

3.2.2 Security Cards

Security Cards are a tool designed to facilitate brainstorming of security threats, using a deck of cards that address different aspects of potential attacks [6]. Created by Tamara Denning, Batya Friedman, and Tadayoshi Kohno, the cards cover four main dimensions: adversary motivations, adversary capabilities, adversary methods, and human impact [4]. This approach aims to foster creativity and collaboration among stakeholders, encouraging a more holistic and comprehensive analysis of threats [39].

Each card provides examples and scenarios related to its dimension, helping teams explore vulnerabilities that might otherwise be missed by traditional methods [6]. For example, in the "Human Impact" dimension, cards can highlight how security breaches can affect privacy, emotional or financial well-being, providing a more user-centric perspective [4].

Security Cards have been used in a variety of applications, such as protecting biometric systems against presentation attacks [23, 4]. Their flexible structure allows them to be adapted to different organizational contexts, including decentralized environments [43]. In horizontal organizations, Security Cards facilitate the participation of multiple stakeholders, promoting distributed governance and reinforcing collaboration [39].

Despite their potential, the methodology can generate a high number of false positives, which requires additional effort to filter relevant threats [4]. However, their emphasis on creativity and inclusion of multiple perspectives makes Security Cards a valuable tool for exploring emerging threats and strengthening organizational resilience [39].

3.2.3 Personae Non Grata

Personae Non Gratae (PnGs) represent an innovative approach to threat modeling, notable for their focus on malicious users and undesirable behaviors [3]. Inspired by traditional user experience design personas, PnGs help anticipate how adversaries might exploit vulnerabilities in a system, providing a detailed adversarial perspective [26].

PnGs are created through techniques such as crowd-sourcing, allowing different stakeholders to contribute insights to threat identification [26]. This collaborative approach increases the breadth and diversity of attacker profiles considered, allowing for more robust modeling that is adapted to different contexts [3].

One of the main advantages of PnGs is their ability to capture specific attacker motivations, capabilities, and behaviors [26]. For example, a PnG might describe an adversary who uses phishing to obtain credentials or exploits security flaws in financial transactions [3]. This level of detail helps in prioritizing countermeasures and allocating security resources [26].

In addition, PnGs are particularly effective in contexts where internal and external threats overlap [3]. In flat organizations, where governance is distributed and responsibility is shared, PnGs help map potential risks that may arise from internal actors, such as employees, or external actors, such as competitors [3].

Despite its benefits, implementing PnGs requires significant effort to ensure that profiles are accurate and relevant [26]. However, when integrated with other methodologies such as attack trees or STRIDE, PnGs provide an additional layer of analysis, making them an indispensable tool for organizations seeking to comprehensively understand and mitigate threats [26].

3.3 Hybrid and Collaborative Approaches

Hybrid and collaborative approaches seek to integrate different methodologies to create adaptable and effective frameworks in different contexts [25, 48]. Among these, Hybrid Threat Modeling Method (hTMM) stands out, combining elements of different frameworks for a comprehensive risk analysis. It is particularly useful in scenarios involving multiple stakeholders and requiring alignment between security objectives and business priorities [25].

Another example is Collaborative and Remote Threat Modeling (CoReTM), designed to facilitate threat modeling in distributed or remote teams. Using collaborative tools, such as shared annotation platforms, CoReTM makes the process more accessible and inclusive, being ideal for horizontal and global organizations [48].

Finally, Participatory Threat Modeling (PTM) promotes the inclusion of a wide range of stakeholders in the threat modeling process [43]. This approach values the diversity of perspectives, being especially relevant in decentralized contexts where transparency and collective participation are essential [43]. These methodologies reinforce distributed governance and strengthen organizational resilience, complementing the work developed by more traditional frameworks [9].

3.4 Decentralized Trust and Cryptographic Frameworks

3.4.1 COLBAC

Collective based access control system (COLBAC) is an access control model designed to address the specificities of horizontal organizations, promoting a democratic and participatory approach to access authorization. Its proposal seeks to overcome the challenges imposed by traditional access control models, such as Discretionary Access Control (DAC), Mandatory Access Control (MAC) and Role-Based Access Control (RBAC), which often reinforce hierarchical dynamics that are inadequate for horizontalized structures [9].

One of the most striking features of COLBAC is its ability to align access control with horizontal governance practices, allowing decisions to be made collectively through democratic processes [43]. The model organizes resources and processes into three main spheres: the Collective Sphere, which concentrates critical resources subject to collective approval; the User Sphere, which encompasses individually managed resources based on

traditional controls; and the Immutable Sphere, responsible for storing logs and records in an unalterable manner, ensuring transparency and traceability [9].

In the context of COLBAC, interactions with the Collective Sphere follow a process structured in three phases: in the Draft Phase, the user creates a token that specifies the permissions and objectives of the action; in the Petition Phase, the token is submitted to a vote by the members of the organization; and, finally, in the Authorization Phase, the results of the vote determine the approval or rejection of the action, with all records being stored in the Immutable Sphere [9]. This structure offers flexibility by allowing the adaptation of the level of horizontality according to the needs of the organization, including in crisis situations that may require temporary centralizations [9].

Despite its advantages, COLBAC faces challenges inherent to its democratic approach [9]. Frequent voting processes can result in user fatigue, especially in larger organizations [9, 42]. In addition, democratic attacks, such as quorum manipulation or the abuse of emergency tokens, pose significant risks. Such issues can be mitigated by implementing independent audits, dynamic adjustments to quorum criteria, and mechanisms that limit the use of emergency tokens [9]. Another important issue is the learning curve associated with the model, which requires familiarity with democratic practices and an understanding of how tokens work [9].

COLBAC offers an innovative solution for organizations that want to align their horizontal governance with robust digital security practices [9]. Its transparency, flexibility, and commitment to democratic participation position it as a strategic tool to overcome security challenges in decentralized structures, transforming potential vulnerabilities into opportunities to strengthen collective autonomy [9, 42].

3.4.2 ABCcrypto

Asset-Based Cryptocurrency (ABC) is a threat modeling framework specifically developed to address the peculiarities of cryptocurrencies and blockchain-based systems. In contrast to generalist frameworks such as STRIDE, ABC is designed to address the unique security challenges presented by distributed and permissionless systems, where actors distrust each other and economic incentives play a central role [1].

The main differentiator of ABC is the introduction of collusion matrices, which allow the analysis of threat scenarios involving collaborations between different malicious actors. This systematic approach reduces the complexity of the modeling process by eliminating irrelevant cases and grouping scenarios with similar effects [1]. Furthermore, the framework uses threat categories specific to cryptocurrencies, considering not only tangible assets, such as blockchains and tokens, but also abstract assets, such as privacy and reputation [1].

A key feature of ABC is its ability to tailor threat categories to the objectives and assets of each [1] system. The process begins with a detailed characterization of the system model, identifying participants, assets, and financial motivations. Threat categories are

then derived based on potential violations of the security properties of the assets, such as service corruption, payment theft, and blockchain inconsistencies. Finally, concrete attack scenarios are enumerated and analyzed using the collusion matrix, which considers all possible combinations of attackers and [1] targets.

ABC also highlights the importance of incorporating economic analysis and financial incentives into the risk mitigation process. For example, "detect and punish" mechanisms can be implemented to discourage dishonest behavior by making it financially unviable. This use of game theory and economic modeling is particularly effective for addressing threats that cannot be neutralized by cryptographic means alone [1].

The effectiveness of ABC has been demonstrated in case studies involving real systems such as Bitcoin, Filecoin, and CacheCash. In the case of Filecoin, the framework revealed significant gaps in the public design, particularly in collusion scenarios that had not previously been considered. In CacheCash, ABC was used from the early design stages to identify 525 cases of collusion and implement incentive-based countermeasures [1].

While ABC has clear benefits, it is not without its challenges [1]. Creating collusion matrices and analyzing threat categories in detail can be resource-intensive, especially in systems with multiple participants and complex assets. However, these efforts are rewarded by the identification of critical threats and the robustness of the proposed solutions [1].

ABC offers an advanced and adaptable approach to cryptocurrency threat modeling, demonstrating that specialized frameworks can significantly improve the security and resilience of distributed systems [1].

3.4.3 PGP and the Web of Trust

Pretty Good Privacy (PGP), developed by Philip Zimmermann, is a cryptographic system that combines privacy, authentication, and convenience to protect messages and files [30]. PGP uses public-key cryptography to enable secure communication between individuals, even without prior trust or key exchange over secure channels [30].

In the PGP operating model, each individual has a pair of keys — a public key, which is widely disseminated, and a private key, which is kept secret. The public key is used to encrypt messages, while the private key is used to decrypt them. This scheme not only ensures the privacy of messages, but also allows authentication through digital signatures, ensuring the integrity and origin of a content [30].

A central feature of PGP is the Web of Trust (WoT) model, which adopts a decentralized approach to identity validation. Unlike centralized hierarchies of Certificate Authorities, WoT allows any user to digitally sign another's public key, certifying its authenticity. These signatures create a distributed trust network, in which the validation of a public key depends on the accumulated trust of the signatures of other trusted users [30].

In PGP, each user can assign different levels of trust to other individuals to act as "trusted introducers". This mechanism allows the trust network to be built organically,

reflecting natural social relationships. For example, a user may fully trust another to certify keys, or only marginally, depending on their perception of the introducer's competence and integrity [30].

In addition to promoting decentralization, WoT also provides resilience against [30] attacks. Rather than relying on a single point of failure, as is the case in centralized systems, WoT allows users to validate public keys based on multiple signatures, reducing the impact of individual compromises. However, this approach also presents challenges, such as the difficulty of managing large key rings and the subjectivity in assigning trust levels [30].

The relevance of PGP and WoT to horizontal organizations is evident [9]. Non-hierarchical structures can take advantage of the decentralized nature of WoT to create security systems aligned with the principles of collective autonomy and distributed governance . By allowing each participant to build their own web of trust, PGP strengthens security without compromising the horizontality of these organizations [42, 9].

3.5 Comparative Perspectives

3.5.1 Evaluation Criteria

The evaluation of threat modeling frameworks requires objective criteria to compare their effectiveness, applicability, and suitability to different organizational contexts [40]. Key criteria include the ability to identify specific threats, adaptability to changes in the operational environment, implementation costs, and the integration of social, economic, and technical dimensions into the modeling process [51]. In addition, scalability and the ability to handle complex organizational structures are critical factors [1].

STRIDE, for example, is widely used for its simplicity and applicability in traditional software systems [38]. However, it faces limitations in decentralized environments, such as cryptocurrencies or horizontal organizations, due to its reliance on predefined threat categories [20]. In contrast, frameworks such as ABC offer a specialized approach, using collusion matrices to explore threats in distributed and permissionless systems, while PASTA focuses on iterative risk assessment for dynamic systems [1, 47].

Frameworks such as COLBAC, on the other hand, stand out for integrating democratic processes into threat modeling, allowing horizontal organizations to align security and participatory governance. Despite its innovation, COLBAC faces usability challenges due to the need for familiarization with democratic processes and the risk of voting fatigue in large groups.

Finally, scalability is essential for organizations dealing with multiple participants and complex interactions [1]. Techniques such as scenario fusion in ABC demonstrate that specialized frameworks can mitigate operational costs while maintaining analytical robustness [1]. This makes them suitable for modern systems that require both technical precision and organizational flexibility [9].

3.5.2 Applicability in Non-Hierarchical Organizations

The applicability of threat modeling frameworks to non-hierarchical organizations depends on their ability to address specific dynamics of these structures [9, 51]. Horizontal organizations operate on the basis of distributed governance, equitable participation, and the absence of formal centralization, requiring approaches that respect and reinforce these principles [42].

Frameworks such as COLBAC and ABC demonstrate strong compatibility with horizontal organizations due to their emphasis on collaborative processes and adaptability to decentralized contexts. COLBAC uses collective authorization tokens to align security and democratic governance, allowing flexibility between temporary centralization and horizontal control. ABC incorporates economic analysis and financial incentives to mitigate risks in blockchain ecosystems, offering tools such as collusion matrices to map complex threat scenarios.

Traditional frameworks such as STRIDE and attack trees provide a solid foundation for threat identification, but their applicability is limited in horizontal organizations due to their reliance on formal hierarchies and centralized control points [38, 35]. In contrast, emerging approaches such as PTM promote greater alignment with the needs of these organizations by integrating stakeholders at every stage of the process.

The choice of a framework for horizontal organizations must balance technical effectiveness with transparency and collective engagement. Solutions such as ABC and COLBAC excel at integrating social and economic dimensions, demonstrating that security can be strengthened through collaborative practices and inclusive governance [1, 9]. These approaches are particularly relevant for organizations seeking to align security with organizational autonomy and resilience [9].

Chapter 3 analyzed traditional threat modeling approaches, highlighting established methodologies such as STRIDE, as well as their applications and limitations in decentralized and collaborative environments. The fundamental categories used in vulnerability identification were described and the necessary adaptations for non-hierarchical contexts were discussed. Based on these analyses, Chapter 4 presents the preliminary design of a protocol specifically adapted for horizontal organizations, describing the fundamental security and governance requirements necessary to maintain organizational resilience and ensure democratic participation in threat analysis and mitigation processes.

Table 3.1: Comparison of Threat Modeling Methods

Method	Threat categories considered	Scope	Key Features	Limitations
STRIDE	Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, Elevation of Privilege	Software and application security (design-phase threat analysis)	Six-category mnemonic covers major attack types (acts as a checklist). Applied to system models (DFDs) to systematically identify threats per component. Simple and widely adopted; easy for teams new to threat modeling.	Focused on technical threats – may miss threats outside its six categories (e.g., collusion or social attacks). No built-in risk ranking, so a long list of threats may need additional prioritization. Considered less detailed for complex, business-specific threats.
Attack Trees	No fixed set – any threat goal can be modeled (e.g., malware attack, insider abuse, social engineering) as the root, with sub-goals as attack steps.	Broadly applicable (software, cyber-physical systems, critical infrastructure) for visualizing attack paths.	Graphical tree diagrams map out attacker goals and all possible paths (leaf nodes are specific attack methods). Can capture complex multi-step attacks (including technical exploits or human tactics) by branching logic. Often used in combination with other methods (e.g., STRIDE or risk scoring) for comprehensive analysis.	Time-consuming to create for large or complex systems (trees can become very large). No inherent risk scoring – requires supplemental analysis (like CVSS) to prioritize threats. Maintenance can be difficult as systems evolve (manual updates needed for tree changes).
PASTA	Not predefined by categories – addresses all threat types identified through its stages, from technical vulnerabilities to misuse and fraud (prioritized by business impact).	Enterprise and critical systems where business impact and risk alignment are key (e.g., fintech, large applications). Integrates with SDLC and organizational risk management.	Seven-stage, risk-centric methodology (from defining scope to attack simulation) providing in-depth analysis. Aligns threats with business objectives – focuses on likely attacks that matter most to the organization's mission. Emphasizes risk prioritization: analyzes likelihood/impact of each threat to guide resource allocation.	Complex and resource-intensive – the 7-step process is lengthy and requires cross-functional expertise. Not widely used compared to simpler models; has a steep learning curve and longer implementation time. Geared toward comprehensive analysis, which may be overkill for small projects or early-phase designs.
Trike	Doesn't use classic threat categories – focuses on unauthorized actions on assets. Threats are modeled as any action (Create, Read, Update, Delete) by any actor that violates assigned permissions (e.g., an insider modifying data they shouldn't, or an outsider gaining illicit access).	Primarily for security audit and risk management in software systems. Suitable for organizations wanting to set and verify acceptable risk levels for each asset/user role.	Risk-based audit approach – assigns a risk score to each asset's threat scenarios, ensuring risk is acceptable to stakeholders. Uses a unique actor–asset matrix (CRUD matrix) to identify where an actor's permitted actions differ from potential malicious actions. Provides a structured way to involve stakeholders in deciding which risks to mitigate vs. accept (tying into governance).	Can be too granular/complex in large IT environments – requires detailed mapping of all assets, roles, and permissions. Less mainstream and fewer tooling options and community support compared to STRIDE/PASTA. Focuses on internal policy violations; may require augmentation to cover threats like external social engineering or collusion not captured by role-permission analysis.
ABC	Collusion-based threats (multiple actors cooperating maliciously) and financial attacks specific to crypto systems. Derives custom threat categories for new blockchain assets (e.g., double-spend fraud, consensus manipulation) beyond traditional threat lists.	Blockchain and decentralized systems (cryptocurrencies, decentralized organizations). Tailored for systems with economic incentives and distributed trust, but also applied to other large-scale distributed systems (e.g., cloud-native architectures).	Introduces collusion matrices – a novel tool to systematically enumerate complex collusion scenarios among actors (forcing analysis of combinations of insider/outsider attacks). Asset-centric: identifies unique assets (crypto tokens, consensus, smart contracts) and derives system-specific threat categories incorporating financial impact.	Specialized scope – designed for cryptocurrency and blockchain context, so it may require adaptation to use in other domains. Complexity: Collusion analysis can become intricate (though the matrix approach manages complexity, it still demands detailed domain knowledge).

4

Solution

4.1 Threat Modeling Protocol for Horizontal Organizations

Horizontal cooperatives face unique security challenges. Without a central authority, they are vulnerable to attacks that exploit their open, democratic nature. For example, an attacker might create fake member identities to sway decisions (a Sybil attack) or abuse the consensus process to cause chaos. At the same time, horizontality can be a security strength: distributing decisions means there's no single point of failure. This protocol treats horizontality as an asset, making security a collective endeavor rather than a top-down mandate.

4.1.1 Key Design Principles

- **Transparency**: Security activities (like decisions, configurations, and incidents) should be visible to members. Open logs and auditable records ensure nothing happens "behind closed doors, building trust and accountability among the group.

- **Decentralization**: No single person should have unchecked power over systems or data. Access and control are distributed. This prevents a "single admin from being a weak link and avoids creating a digital vanguard (where a tech-savvy few hold all the keys).

- **Democratic Participation**: All members can participate in identifying and addressing threats. Security decisions are made through inclusive discussions or votes, so measures have collective buy-in. This keeps the process aligned with the coop's democratic governance.

- **Traceability**: Every important action (granting access, making a change, etc.) leaves an immutable trail. For example, changes can be logged on tamper-proof ledgers and digitally signed by those who approved them. This way, if something goes wrong, the coop can trace what happened and who was involved, without relying on memory or hearsay.

- **Resilience**: The protocol aims to strengthen the coop's ability to withstand and recover from threats. By eliminating single failure points and planning for crises (with backup plans and rapid response mechanisms), the organization stays resilient even under attack. If one safeguard fails, others are in place to limit damage and bounce back quickly.

These principles ensure that improving security will not undermine the organization nature of the group. Instead, security measures will reinforce collaboration, shared responsibility, and trust. In practice, this means building security into everyday cooperative workflows and governance. What follows is a step-by-step threat modeling process designed with these values in mind. It's written in accessible language so that any member can take part. Each step includes guidance and checklists for participatory activities, and the protocol can be scaled or adapted for organizations of different sizes and structures.

(Note: While this protocol is inspired by established frameworks like PASTA, it does not use the formal seven-stage PASTA terminology. Instead, it presents an equivalent logic in a more accessible format.)

4.1.2 Target Audience

The protocol is designed for members of horizontal organizations without specialized cybersecurity expertise. Unlike traditional expert-oriented models, our protocol emphasizes simplicity and accessibility. It supports stakeholders involved in decision-making, operational activities, conflict resolution, and coordination tasks, as well as informal community groups, by providing clear guidance and intuitive methods for effectively dealing with security threats.

4.1.3 Ethics and Protection of Organizations

When constructing and applying the threat modeling protocol in non-hierarchical organizations, it is imperative to consider ethical principles as structuring elements that transcend the merely technical aspect of cybersecurity. Ethical concerns are based on the explicit commitment to protecting not only technological integrity, but also the individuals and groups involved in organizational processes.

A crucial aspect in this context is the responsibility regarding the confidentiality of sensitive information and the protection of the privacy of organizational members. The inadvertent exposure of security flaws can cause significant damage, not only operational, but also personal and social. Therefore, the protocol must incorporate strict guidelines on the ethical treatment of identified vulnerabilities, ensuring that such information is managed in a restricted manner and shared only with authorized individuals or groups, always respecting the principle of least privilege.

Additionally, it is vital to establish clear internal communication mechanisms to ensure that any identified vulnerabilities are immediately reported, mitigated and documented without unnecessary public exposure.

4.2 Threat Modeling Process Overview

The threat modeling process is broken into eight collaborative steps. In a small cooperative, most steps can be taken in all-hands meetings or workshops with everyone. In larger groups, you might delegate initial work to committees or working groups, but every member should have a chance to review and contribute at each stage. The process is iterative and modular, so you can adjust the depth or format of each step based on your organization's size and needs. For each step below, we outline the purpose, activities, and participatory approach, along with tips to adapt to different situations.

4.2.1 Step 1: Establish Context and Goals (What Are We Protecting?)

4.2.1.1 Purpose

Set the stage by agreeing on what assets and operations you need to protect, and what your security objectives are. This ensures everyone is on the same page about why you are doing threat modeling and what "success looks like. In traditional terms, this is like defining the business/mission objectives and scope for security.

4.2.1.2 Activities/Checklist

- **Identify Critical Assets**: As a group, list out what is most valuable to your organization. This can include digital assets (member data, documents, the website, chat platforms), physical assets (office space, devices, servers), and intangible assets (the organization's reputation, member trust, know-how). Ask yourselves: "What would really hurt if it were stolen, destroyed, or made public? Write these down for all to see.

- **Outline Key Operations/Workflows**: Describe in simple terms what the organization does day-to-day. For example, "We coordinate orders through an online platform, or "We have weekly meetings to make decisions, or "We run a community space with an entry badge system. Understanding these workflows helps identify where disruptions would be most damaging.

- **State Security Objectives and Requirements**: Discuss what security means for your organization. Do you need to keep member data private? Ensure your service is always available? Meet any legal/privacy regulations (like GDPR)? Also consider organization statutes or policies about confidentiality and data handling. For instance, if your statutes say all financial info must be accessible to members, that influences

how you balance transparency with confidentiality. Jot down these objectives and any compliance requirements.

- **Define the Scope and Boundaries**: Decide what will and won't be covered in this threat modeling exercise. Maybe you want to focus on a particular system (e.g. your member database and communication tools) and not on unrelated areas. Or include only digital systems but not physical office security or vice versa. Clearly defining scope prevents the discussion from going off-track. It's okay to start with a narrow scope (like "our shared Google Drive and Slack workspace) and expand later if needed.

- **Agree on Terminology**: Ensure everyone understands basic terms you will use. For example, define what you mean by asset, threat, vulnerability, etc., in plain language. This avoids confusion later (a quick glossary on a whiteboard or shared doc can help).

4.2.1.3 Participation Tips

In a small coop, you can do this step in a single meeting where everyone contributes to the list of assets and goals. In a larger coop, consider sending a short survey or having breakout groups to gather input, then consolidating the results in a plenary session. Make sure the final list of assets and scope is reviewed or approved by the group (e.g. via a show of hands or online poll) so you have collective agreement on "what we care about protecting. This collaborative start sets a democratic tone for the whole process.

4.2.2 Step 2: Map Systems and Trust Boundaries (How Do We Work?)

4.2.2.1 Purpose

Create a shared understanding of how information and processes flow in your coop, and where important trust boundaries lie. Essentially, draw a map of your organization's sociotechnical system including people, tech, and their interactions. In threat modeling, this is similar to diagramming your system architecture and identifying entry points. For a cooperative, it also means noting social trust assumptions (who/what we trust and in what ways).

4.2.2.2 Activities/Checklist

List Components and Assets:

- **Hardware/Infrastructure**: e.g. member laptops, a server or NAS, routers, smartphones used for work, IoT devices in the office.

- **Software/Tools**: e.g. the platforms you use – Google Drive, Slack/Matrix, Nextcloud, Loomio for decisions, etc., as well as any custom software or website your coop has.

- **Data Stores**: e.g. databases, cloud storage, email archives – what data is stored and where.
- **People/Roles**: e.g. general members, IT volunteers, finance coordinators, any outsiders like an accountant or platform provider.
- **Processes**: e.g. how a decision is made (proposal -> discussion -> vote), how a new member is onboarded, how finances are managed.

Write these out, possibly in categories. Essentially, you are enumerating what pieces make up your organization "system". After that, diagram the Workflow: On a large paper or using a simple online diagram, sketch how these components connect. Draw who interacts with what: e.g. members (people) log into the chat platform (software) to discuss, or the website communicates with a payment processor. Draw arrows for data flow or interaction: emails sent, files shared, money transferred, etc. Keep it understandable – you can use simple icons or just labeled boxes and arrows. Mark any external services (like a third-party payment gateway or cloud provider) clearly, maybe with a different color or a cloud icon, since those are partly outside your control.

Identify Trust Boundaries: A trust boundary is a point in the system where the level of trust changes. For instance:

- Between an external user and your internal system: e.g. a public website vs. your internal database.
- Between a regular member and an admin interface: (if any).
- Between your organization network and the open internet.
- Social trust boundaries: e.g., you trust members not to leak info from private discussions, or you trust a core team with certain credentials.

On your diagram, draw a dotted line or a firewall symbol where these boundaries are. Essentially ask: At what points do we assume things are safe on one side and potentially risky on the other? (For example, data inside our Nextcloud is trusted to be seen by members, but anything coming from outside (like file uploads from a new user) might be untrusted until checked.)

Document Who Has Access to What: Alongside the map, list which roles or people have access to which assets. E.g., "Only tech team members can access the server", or "All members can post in the forum", or "Treasurer has the bank account login". This helps spotlight any concentrations of access (if one person holds many keys, that's noted) and areas where trust is placed in individuals. Note External Dependencies: Write down services or partners you rely on and mark them on the diagram (for example, your website host, email service, or any software provider). These are outside your organization but critical; threats can come through them (a concept known as supply chain or third-party risk).

4.2.2.3 Participation Tips

Building the system map can be a fun group exercise. In a small organization, do it together on a whiteboard or shared screen, asking everyone to call out components and connections ("Don't forget we also use Pad for meeting notes!"). In larger organizations, you might have a smaller group draft the diagram (say, an IT working group or a mix of tech and non-tech members) and then present it to others for additions and corrections. Make it interactive: people can stick Post-its on a draft poster or add comments on an online diagram with "Did you include the volunteer's laptop that runs X?" Keep refining until ,members feel the map is an accurate picture of "how things work." The final diagram should be saved in a place where all members can see it, this is now a shared reference for your threat discussions.

4.2.3 Step 3: Identify Threats Collaboratively (What Could Go Wrong?)

4.2.3.1 Purpose

Brainstorm all the potential threats and bad things that could happen to the assets and processes you identified. The goal is a comprehensive list of threat scenarios, covering both technical attacks and social/governance risks. At this stage, quantity is more important than quality. We want to surface as many ideas as possible, without judging them yet. This step harnesses the diverse perspectives in your organization: digitally skilled members might think of hacking scenarios, whereas others might point out process failures or insider issues that a pure tech focus could miss.

4.2.3.2 Activities/Checklist

- **Brainstorm in a Safe Environment**:

  Gather a group of members (ideally representing different roles or viewpoints in the coop) for a threat brainstorming session. Set some ground rules: no idea is too small or too "out there," and everyone's input is valued. It's important people feel comfortable mentioning even unpleasant hypotheticals ("What if one of us turned rogue?") – assure everyone this is about hypothetical situations, not personal distrust.

- **Use Prompts and Creative Tools**:

  - Walk through the system map from Step 2 and pause at each component or boundary: ask "What could go wrong here?" For instance, at the database: "Could someone steal or delete this data? Who might and how?" At a boundary like the internet connection: "What if an attacker intercepts data here or floods us with traffic?"
  - Introduce the STRIDE categories (a classic security mnemonic) in simple terms as a checklist:

    - Spoofing (pretending to be someone else – e.g. fake member login, impersonating an admin),
    - Tampering (messing with data or systems – e.g. altering records, defacing the website),
    - Repudiation (denying an action – e.g. a member does something and later claims they didn't, which is an issue if there's no proof),
    - Information Disclosure (leaks – e.g. private member info exposed),
    - Denial of Service (disrupting service – e.g. someone takes down your communication channel or overloads your server),
    - Elevation of Privilege (gaining higher access – e.g. a regular member somehow gets admin rights). Use these as thought-starters: "Do we have a risk of someone spoofing identity? Tampering with records? etc."
  - Use Security Cards (if available) or hypothetical scenario prompts: Security Cards are a deck with categories like Attackers' Motivations, Methods, Impacts. You can simulate this by asking questions in those dimensions:
    - Motivations: "Who might want to attack us and why? (Ex: for money, for political reasons, disgruntled ex-member, random mischief)"
    - Methods/Resources: "What skills or tools could they use? (Ex: Phishing emails, malware, physical break-in, legal threats, bribes, social engineering phone calls)"
    - Impacts: "What would be the impact if they succeeded? (Ex: website down for days, loss of member trust, financial loss, legal trouble)"

    These prompts help the group consider not just obvious IT threats, but also things like internal misuse, mistakes, or external events.

- **Distinguish Different Threat Sources**: As ideas emerge, note whether each threat scenario is external (coming from outside, e.g. a hacker, a virus, a competitor, a random troll) or internal (coming from within the coop, e.g. a member error, an insider attack, conflict/miscoordination). Both are important. For example:
  - External threat example: "A hacker defaces our website or steals our member list."
  - Internal threat example: "A coop member accidentally shares a private document publicly," or "Two factions in the coop conflict and someone locks others out of an account."
  - Hybrid threat: "An external attacker tricks a member (social engineering) to gain access" or "An ex-member colludes with an outsider."

  By labeling these, you ensure internal governance vulnerabilities (like abuse of trust or poor processes) get attention alongside technical attacks.

- **Write Down Concrete Scenarios**: For each idea, capture it as a short scenario description. For instance:

  - "Sybil Attack on decision-making:" An attacker (or someone from the community) creates multiple fake member accounts to gain extra votes in an online poll, influencing a cooperative decision illicitly.
  - "Insider data leak:" A discontented member with access to sensitive data decides to leak member emails and addresses to the public.
  - "Ransomware on shared drive:" Malware infects a member's computer and encrypts the shared cloud drive files, making them inaccessible until a ransom is paid.
  - "Lost credentials:" A member who manages the Twitter account leaves suddenly, and no one else has the password – the coop loses control of its own social media for a time.
  - "Miscoordination outage:" In a crisis, no one is designated to respond (because everyone thinks someone else will) and a small issue (like a certificate expiry) escalates, taking the website offline for days.
  - "Service provider failure:" The third-party platform (e.g. web host or payment processor) goes down or is compromised, affecting the coop's operations.

  Aim for a broad list, covering cyber-attacks, human mistakes, physical events (the office gets robbed or a server gets wet), and governance failures. Don't worry at this stage if some scenarios seem very unlikely – list them if someone is concerned about it.

- **Ensure Social/Process Threats are Included**: Cooperatives might face threats like quorum manipulation (exploiting the rules of consensus/voting), abuse of emergency powers (someone invoking a crisis to grab authority), or "digital vanguard" accumulation (one person quietly gaining control of many digital assets because no one else steps up). Include these in your brainstorming. For example, "Member X holds all the keys and if they quit or go rogue, we're locked out" is a valid threat scenario to record (it's an internal risk).

4.2.3.3 Participation Tips

Use sticky notes or a shared online document during the brainstorm so everyone can contribute simultaneously. One approach is to give participants 5–10 minutes to silently write as many "What if. . . ?" threats as they can (one per sticky note), then post them and discuss. This helps include those who might be quieter in a big group. After the brainstorm, group similar threats together (cluster duplicates) but do not dismiss any threat yet just because it seems minor or absurd. The point is to capture the collective worries and creative

ideas. If you have a very long list (which is good), you can categorize them into groups like "Tech Infrastructure," "Member Behavior," "Outsider Attacks," "Natural/External," etc., for clarity.

Finally, thank everyone for their ideas – emphasize that all contributions (even wild ones) help build a complete picture. Save this threat list somewhere accessible (it becomes an input to later steps). In a coop, making this list visible (on the wall or online) also signals transparency: everyone sees what threats are on the radar.

4.2.4 Step 4: Profile Adversaries (Who Might Attack Us?)

4.2.4.1 Purpose

Humanize the threats by creating adversary personas – fictional characters that represent types of attackers or sources of threats. This helps the group think from an attacker's perspective ("what would X do?") and ensures you consider the motivations and capabilities behind the threats. In a cooperative, adversaries aren't only outside hackers; they could be internal (like a frustrated member) or systemic (like software bugs or accidents – though we focus on personas for intentional actors here). Developing these profiles makes later analysis more concrete and relatable.

4.2.4.2 Activities/Checklist

- Identify Key Threat Actors: Look at the threat scenarios from Step 3 and ask, "Who would carry out these actions?" You'll likely find a few recurring archetypes. For example:
  - A malicious outsider (generic hacker or vandal) with no stake in the coop, just attacking for personal gain or fun.
  - A state or corporate actor who opposes your coop's mission (if relevant politically or competitively).
  - A disgruntled member or ex-member who knows the internal system and might seek revenge or change outcomes.
  - A negligent insider (not malicious, but someone who might make mistakes – although this is more a cause of accidents than an "attacker," it's still a persona to consider for unintentional threats).
  - Technical vs non-technical adversaries: e.g., a script-kiddie hacker with moderate skills, vs. an internal activist who might misuse rules rather than code.
- Create Persona Descriptions: For each actor type, write a short profile including:
  - Name and Role: Give them a nickname that captures their role, like "Mallory the Malicious Member" or "Ingrid the Inattentive Intern" or "Oscar the Outside Hacker." This makes it easy to refer to them.

  - Motivations/Goals: What do they want? Money, disruption of the coop's activities, ideological reasons, personal vendetta, thrill? For instance, Mallory (disgruntled ex-member) might be motivated by revenge for being voted out of a project, whereas Oscar (outsider) might just want to steal data to sell.
  - Capabilities/Resources: What skills or resources do they have? Oscar might have hacking tools or know how to exploit software. Mallory has insider knowledge of how your coop operates and where the weaknesses in process are, but maybe not advanced technical skills. Ingrid (the negligent intern) isn't trying to attack but might unknowingly introduce risk by ignoring policies or falling for scams.
  - Possible Methods: Given their motivation and skills, how might this persona attack or cause an incident? E.g., Oscar might use phishing emails or find a bug in your website; Mallory might exploit their still-active login or sow misinformation in meetings; Ingrid might click a malicious link or use a weak password that gets guessed.
  - Scenario Tie-In: Connect each persona to one or more of the scenarios from your threat list. For example, note that "Sybil attack on voting" would be done by someone like Mallory (if internal) or an outsider Sam the Sockpuppet if external. "Ransomware on shared drive" could be set off by Ingrid's mistake or by Oscar deliberately.

- Document in an Accessible Format: Make one page or slide for each persona. You can even include a representative image (e.g., an icon or cartoon; avoid using real people's photos to prevent bias or confusion). The idea is to make these threat actors memorable. Write in plain language – this is an internal tool for understanding, so it doesn't need to be formal. For example:

  - Persona: Mallory (Disgruntled Ex-Member)
    * Background: Former member who left after a conflict. Still has some credentials left over.
    * Motivation: Revenge, proving a point about weak security.
    * Skills/Resources: Knows our systems well, not a hacking expert but savvy enough to abuse any oversight. Possibly still in contact with current members (could socially engineer info).
    * Likely Actions: Try to log in to systems with old account, misuse a still-active access, or spread false messages pretending to be someone else (impersonation). Might collude with an outsider for technical help.
    * Targeted Assets: Member directory (to expose personal info), internal chat (to disrupt communications), decision platform (to sway outcomes or just cause chaos).

Do this for each major adversary types.

- Include at Least One Insider Persona: It may be uncomfortable but include a scenario of a malicious or careless insider. Cooperatives thrive on trust, yet history shows sometimes insiders can cause harm (intentionally or not). By creating, say, "Insider Irene" who is well-meaning but prone to bypassing rules, or "Rogue Ray" who turns against the group, you can discuss those threats without pointing fingers at a real person. Make it clear this is hypothetical to improve security for everyone.

- Use Personas in Discussion: Once you have personas, you can use them in future steps. For example, when thinking about mitigations, you might ask "Would this stop Mallory?" or "How would we detect Oscar's actions?" Personas help ground these discussions.

4.2.4.3 Participation Tips

Developing personas can be done in small breakout groups, with each group taking one persona to draft. It's a creative exercise – encourage storytelling, but keep it grounded in plausibility. After drafting, share with the whole team for feedback. Non-technical members often contribute richly here ("Actually, a really angry member might do X, not Y"), and technical members can refine how an external hacker persona might behave. Keep the tone collaborative and even a bit playful (people can get quite engaged coming up with hacker nicknames!), but always tie it back to real threats you identified. In the end, post the personas alongside the threat list. They become part of your "threat model" documentation that everyone can reference.

4.2.5 Step 5: Analyze Attack Scenarios (How Could Attacks Happen?)

4.2.5.1 Purpose

Now take your threat list and personas, and dive deeper into how those threats could play out step-by-step. This scenario analysis helps you understand the sequence of events in an attack and where your weak points are. In practice, this means building attack narratives or attack trees and possibly simulating some scenarios in a tabletop exercise. This step turns abstract threats into concrete stories, revealing exactly what vulnerabilities enable an attack and how severe the consequences would be. It's a bridge from brainstorming to action: by visualizing attacks, you prepare to figure out defenses.

4.2.5.2 Activities/Checklist

- Construct Attack Trees or Flowcharts: Pick a high-priority threat scenario (you will prioritize formally in the next step but start with one that seems obviously serious or emblematic). For example, take "steal member data" or "Sybil attack on voting".

Write that as the attacker's goal at the top (root of the tree). Then brainstorm the possible paths to reach that goal. For each path, break it into steps:

  - Example: Goal = "Unauthorized person accesses member list." Path 1 might be: External hacker exploits a software vulnerability in the database -> gains admin privileges -> extracts the data. Path 2: Malicious insider uses their legitimate access -> downloads the data -> shares it out of malice. Path 3: Social engineer tricks a member into giving up their password -> logs in as them -> navigates to data and copies it.
  - Draw this as a tree: the goal at top, then branches for each different method, and sub-branches for steps/prerequisites. (It can be as simple as bullet indentations if not drawing: e.g. a numbered list of steps for each scenario.)
  - Mark points on the tree where a defense exists or fails. E.g., "Does our system have a vulnerability? Possibly if software not up-to-date." Or "Insider route: relies on that member having access – do all members have access to full member list? Maybe that's a policy question." This highlights vulnerabilities at each step.
  - Keep the tree at a detail level that's useful – not every keystroke, but significant steps and decision points. Two or three levels deep is usually fine (primary ways -> specific steps).

- Conduct Tabletop Simulations: For some scenarios, especially ones involving multiple people or processes, do a role-play tabletop exercise. Assemble a small group and narratively walk through the scenario:

  - Assign someone to play the adversary (using one of your personas) and others to play defenders or just observers.
  - Example: Simulate "Sybil attack during an online vote." Narrator says: "It's the day of a big proposal vote. Unknown to the group, Mallory has created three fake member identities over the last month." Then step by step: "Vote opens. Mallory votes as herself and as Alice, Bob, and Charlie (her fake profiles). The system tallies four votes from what appears to be four people." Discuss: Would anything at that moment flag an issue? How would the coop notice? Perhaps another member finds it odd that there were three new members who never spoke but voted. Or maybe nobody notices until later. Continue: "The vote passes with those extra votes. Later, someone questions the outcome. . . " This kind of storytelling helps highlight if your current processes have detection or not. Participants can chime in with "At that point, I would check the member list. . . " or ask "Do we verify new online voters somehow?" Write down these insights.

  - Another example: "Insider incident response." Simulate what happens if an insider is caught doing something suspicious – do you have an agreed response or does chaos ensue?
  - Or "Server ransomware attack at 2 AM." Who gets the call? Is there a backup? Walk through who does what.

  The idea is to practice an attack in theory to see where your response or system breaks. It's much cheaper to find gaps this way than during a real incident.

- Identify Vulnerabilities at Each Step: As you chart out these scenarios, explicitly list the vulnerabilities or weak points that make the attack possible. These could be technical (e.g. "Outdated plugin allows injection", "No backup exists for database"), or organizational (e.g. "New members are not verified, allowing fakes", "Only one person knows how to reset the server"). Also note any existing controls and whether they work:
  - For each step in the scenario, ask "What should stop this? Do we have something to stop it? Does it actually stop it or can it be bypassed?"
  - E.g., in the Sybil scenario: Vulnerability = lack of strict member verification in the voting system. Existing control = new accounts require admin approval (does that happen? maybe someone auto-approved without checks).
  - In the hacker scenario: Vulnerability = software not patched; Control = we have a firewall, but if the attack comes through a web port, firewall doesn't stop it; or Control = we rely on strong passwords, which might not help if exploit exists.
  - Write these vulnerabilities next to the steps or in a separate list mapped to the scenario. This will directly feed into deciding mitigations.

- Assess Impact and Likelihood for Scenarios: As part of analysis, discuss for each scenario, how bad would it be if this happened? and how likely is it to happen? Use qualitative terms (you will formalize in next step, but start conversation):
  - Impact: High (e.g. coop might dissolve or face legal action), Medium (painful but survivable), Low (minor inconvenience or embarrassment).
  - Likelihood: High (we've seen attempts or it's easy to do), Medium, Low (requires many unlikely failures or very targeted effort).
  - Example: Ransomware encrypting files – Impact High (work comes to halt, data loss) but Likelihood maybe Medium if members are generally careful. Sybil attack – Impact High on governance legitimacy, Likelihood Low/Medium depending on how easy it is to create fake accounts in your system.
  - Capture these impressions because they will help when you formally prioritize in Step 6.

- Leverage Past Incidents: If your coop or similar groups have experienced incidents, incorporate those into scenarios. "This happened before – could it happen again in a worse way?" Learning from near-misses or history makes scenarios very concrete. For example, "Last year someone guessed our Twitter password – what if they had tweeted offensive stuff? Let's play out that scenario."

4.2.5.3 Participation Tips

For attack trees, it can help to have a tech-savvy member sketch the first draft with input, then review with the group to fill gaps. For tabletop simulations, try to involve a mix of roles – perhaps an IT person, a governance person, and a regular member. Keep the tone educational, not fearful or accusatory. Encourage people to "think like an attacker" for a moment – it's often eye-opening for non-security folks and can be even a bit fun in a serious way. Time-box the simulation (e.g. 20-30 minutes each) to keep it focused. After each scenario, have a brief "debrief" discussion: What did we learn? What worked or failed in our current setup? This will set you up well for the next step, because you'll have identified where the biggest weaknesses are.

4.2.6 Step 6: Prioritize Risks Together (Which Problems Matter Most?)

4.2.6.1 Purpose

Not all threats are equal. In this step, the cooperative evaluates all the identified threat scenarios and decides which ones to address first. This is essentially a risk analysis and ranking. Risk is usually judged by two factors: how severe the impact would be and how likely the threat is to occur. By scoring or discussing these, the group can focus on the most critical issues. Importantly, this is done participatorily – everyone's perspective on what is important is considered, keeping the process democratic. The output will be a clear list of top-priority risks that the coop will invest effort in mitigating.

4.2.6.2 Activities/Checklist

- Set Risk Criteria (Impact and Likelihood): As a group, establish a simple shared understanding of risk levels:
  - Define what High, Medium, Low Impact mean for your coop. For example:
    - High Impact: Threatens the existence of the coop, causes major financial loss or legal violation, or fundamentally damages trust (e.g. member personal info exposure, loss of critical systems for long time, major public scandal).
    - Medium Impact: Disruptive and costly, but manageable (e.g. service outage for a day, loss of some data that has backups, temporary hit to reputation).

    - Low Impact: Annoying but minor consequences (e.g. defacement of a webpage, a single incorrect transaction that can be fixed).
  - Define Likelihood similarly:
    - High Likelihood: We have evidence or reason to believe this could happen frequently or easily (e.g. known attempts have occurred, or known vulnerabilities exist and are unpatched).
    - Medium: It could happen under certain conditions or if an attacker dedicates effort, but not trivial or not seen yet.
    - Low: Very rare or would require a perfect storm of failures or an extremely capable adversary.

    If you prefer numbers, you can rate both on a 1-5 scale or 1-10. But keeping it simple with words or a 3-point scale is often fine for organizations, as long as everyone talks through the meanings.

- Evaluate Each Threat Scenario: Go through the list of scenarios (from Step 3/5) one by one and discuss:
  - "If this happened, how bad would it be? (Impact)" and "How likely is it to happen given what we know? (Likelihood)".
  - Encourage input: maybe the IT person knows that a certain attack is actually quite hard, so likelihood is low; but a governance person might point out that even if low, that attack's impact on member trust would be catastrophic. Both points are valid.
  - You can do this discussion informally and just mark each scenario High/Med/Low, or use a more structured method like voting or rating:
    - For instance, create a grid on a whiteboard with Impact on one axis and Likelihood on the other, and place each threat (on a sticky note) in the grid where the group thinks it belongs. This visual "risk map" often sparks debate ("Should this be higher impact than that?" etc.).
    - Alternatively, have each member or a small group give a quick rating for each scenario anonymously (e.g. via a Google Form or pieces of paper), then average the scores. This can help if the group is large to get a baseline, then discuss outliers.
  - Aim for consensus or at least a general agreement on where each major threat sits (exact precision not needed – you mainly want to clearly identify the High-High items).

- Rank the Risks: Based on the evaluations, identify the top tier of risks that demand action. These are typically the scenarios with High impact (even if low likelihood) and those with High likelihood (even if medium impact), especially anything that's

High in both. Often you'll find a handful that stand out as "we absolutely must address these."

  - Also pay attention to "low-hanging fruit" – a scenario that might be Medium risk but can be fixed easily. Sometimes the group might say, "This isn't our worst problem, but it's so simple to prevent that we should just do it." (Example: "Data loss due to no backup" might be medium likelihood and medium impact, but the fix – implement backups – is straightforward, so you might prioritize it early.)
  - You might end up with categories like: Critical (must fix ASAP), Moderate (plan to address), Acceptable or Low (acknowledge but no immediate action).

- Document Rationale: For transparency, write down a brief note next to each top risk about why it's ranked high and what the group's judgment was. E.g., "Sybil attack on voting – High Impact (could undermine our governance legitimacy), Low Likelihood (member sign-up currently requires approval, and we've seen no attempts) – Still a Top Concern because of impact on trust." This record helps if later someone asks "Why are we focusing on this threat over that one?" Everyone can see the reasoning agreed upon.

- Verify Against Goals: Revisit Step 1's assets and objectives. Ensure that the top risks you've chosen indeed relate to what you care about most. If an important asset has no high risks listed, double-check: did we miss a threat for it, or is it truly low-risk? Conversely, if a scenario came up high risk but involves something not central to your mission, consider if it's truly a priority. This sanity check keeps the process aligned with cooperative values and priorities.

- Obtain Group Endorsement: Formally or informally, get the group's nod that "Yes, these are the threats we're most concerned about." In a smaller coop, that might be as simple as everyone saying "Sounds right." In a larger one, you might do a quick vote to approve the risk ranking or have the board/steering committee ratify it. This is important for accountability – it shows the decision on what to do next was made collectively.

4.2.6.3 Participation Tips

Risk prioritization can sometimes lead to debate or disagreement, because people perceive risks differently. That's okay – the discussion is part of the participatory process. Ensure that louder voices don't completely drown out quieter ones; actively ask for input from different members ("We've heard a lot about technical likelihood – how about the user experience perspective, what do you think the impact on members would be?"). If consensus is hard, you can fall back to a vote or averaging scores, as long as the group accepts that outcome. Remember the principle of democratic decision-making followed

by unity in execution: once the group decides that, say, "Insider data leak" is a top risk and "DDoS attack on our website" is lower priority, everyone should respect that and focus on the top risks first. This avoids second-guessing later and ensures a coherent security effort.

By the end of Step 6, you should have a short list of the most important threats to tackle, with group support behind that list. Now it's time to figure out how to mitigate those threats.

4.2.7 Step 7: Mitigations and Governance Decisions (How Do We Fix or Prevent Issues?)

4.2.7.1 Purpose

For each of the top-priority threats, decide on countermeasures and integrate those decisions into the coop's action plan. In other words, figure out what security measures to implement and how to approve and enforce them democratically. This step is where you turn analysis into concrete changes: technical fixes, new or improved policies, training, etc. It's also where you make sure that implementing these fixes doesn't accidentally centralize power or violate cooperative principles. We design the mitigations to both reduce risk and fit into the coop's governance structure.

4.2.7.2 Activities/Checklist

- Brainstorm Mitigation Options: For each high-priority threat scenario, list possible ways to mitigate it. Mitigations can fall into different categories:
  - Technical solutions: e.g. apply a software patch or upgrade, enable two-factor authentication for logins, encrypt data at rest, set up an intrusion detection system, require multi-signature for financial transactions.
  - Process or workflow changes: e.g. institute a checklist for onboarding/offboarding members (so fake identities are caught and departing members lose access promptly), establish a routine backup procedure, require a second pair of eyes (peer review) before pushing changes to the website, add a step to verify any important request (like fund transfers) through a secondary channel.
  - Education and training: e.g. do a phishing awareness session so members can spot suspicious emails, create a quick guide on how to choose strong passwords or use the password manager the coop adopts, run an orientation on security policies for all members annually.
  - Governance/policy measures: e.g. create a policy that "All admin passwords are stored in our shared password manager vault accessible by at least 3 people," or "For emergency decisions, at least 2 out of 3 designated responders must

agree," or "Every proposal in the decision platform must be made by a verified member identity." Essentially, rules that formalize the security practices.

Use the adversary personas as a lens: ask "If we do this, would it stop or deter [Persona] from succeeding?" For instance, to counter Mallory (disgruntled ex-member) misusing credentials, one mitigation might be "immediately deactivate accounts when someone leaves (offboarding policy)." To counter Oscar (outside hacker) exploiting a bug, mitigation is "keep software updated, and maybe run security scans." To counter the Sybil scenario, mitigation could be "implement stricter member verification for online voting (like each account must link to a real person's membership record)."

List multiple options if they exist, then discuss feasibility: How hard or expensive is it? Does it require outside help or new tools? Does it slow down any workflow (introduce friction)? Does it align with our values (e.g. a mitigation "ban all new members for safety" would harm the coop's openness, so probably unacceptable)?

- Collective Deliberation on Mitigations: For each threat and its potential solutions, discuss with the group (or relevant sub-group) to choose the best course of action. This might happen in a dedicated security meeting or as part of regular meetings. Ensure to involve those who will be responsible for implementing or affected by the change. For example, if the mitigation is "use a new encrypted chat tool," the IT team and regular members should both weigh in (IT on how to implement, members on usability). Some mitigations might be contentious (maybe someone feels requiring 2FA is too burdensome, or rotating admin duties is too chaotic). Strive for consensus by addressing concerns: perhaps provide alternatives or phased approaches. If consensus can't be reached, use your coop's normal decision process (majority vote, etc.) to finalize the decision on that mitigation. Important: Document the decision for each major mitigation: the group agrees "Yes, we will do X to address threat Y." This can be recorded in meeting minutes or an action plan.
- Assign Responsibility and Resources: For each chosen mitigation, decide who will carry it out and by when. Since this is a horizontal organization, avoid dumping everything on one person. Instead:
  - Spread technical tasks among tech-skilled members or a tech working group, but maybe pair them with a non-tech member for transparency.
  - Assign policy drafting to a small team that includes people from governance and operations.
  - If training is needed, maybe identify a member who's good at teaching, or an external expert if budget allows.
  - Ensure each mitigation has an "owner" or small team responsible for seeing it through. This distributes responsibility and builds shared security knowledge.

  - If certain mitigations require budget (e.g. buying a security certificate, or paying for a backup service), note that and ensure it goes through the normal budgeting process of the coop.

  Essentially, integrate these tasks into your coop's project management – treat them like important initiatives with accountability. This also helps avoid the situation where plans are made but no one follows up.

- Implement with Democratic Oversight: As mitigations are implemented, keep the group informed. For example:

  - If the tech team is deploying a new secure tool, they should report progress at meetings or in the group chat.
  - If a policy is drafted (e.g. an "Acceptable Use Policy" for members), circulate it for comments and formally adopt it through the coop's decision mechanism.
  - Unity in execution: Once a security measure is agreed on, all members should cooperate in making it work, even if it's a bit inconvenient. This might mean everyone actually enables 2FA on their accounts, or actually uses the new password manager, not just a few. Culturally, frame it as "we all agreed to this because it protects all of us." When needed, gently remind folks that this was a collective decision (the democratic centralism idea: we discussed openly, now we implement uniformly).

- Plan for Emergency Situations (Temporary Delegation): One big governance aspect to settle is: How will we handle a security emergency or critical incident? In a crisis (like an active cyber-attack, or a major breach in progress), organizations might need to act faster than usual consensus allows. The protocol encourages having a pre-agreed emergency response team or procedure:

  - Decide if you will appoint a small Security Incident Response Team (SIRT) or similar. This could be, say, 3 trusted members with the technical know-how (or quick learners) who are empowered to make snap decisions when an incident is detected.
  - Define the scope and limits of their power: e.g., "They can temporarily shut down a server, or revoke someone's access, or spend up to X euros on emergency help, without full group approval in that moment." Also define the timeframe: their actions are meant to contain the issue, and must be reported ASAP to the membership and reviewed, say, within 24-48 hours.
  - This delegation is temporary and accountable: as soon as the immediate threat is handled, they must explain their actions to the whole coop (transparently via an incident report), and if any decision needs to be permanent (like firing an IT provider or changing a policy), it goes to the group for normal decision-making.

  - If your coop is small, your "team" might just be the few people who know what to do, but still explicitly acknowledge it ("If something blows up, Alice and Bob will take charge for the moment, and inform everyone as they do").

  Having this plan prevents paralysis in a crisis and avoids someone unilaterally taking over just because nothing was defined. Everyone knows in advance who will act and trusts that it's been agreed upon.

- Integrate Security into Governance Documents: Once decisions are made (normal mitigations or emergency roles), codify them. Update your coop's handbook, wiki, or policy docs to include the new security measures. For example, add an "Access Control Policy" section that outlines how accounts are managed collectively, or a note in the decision-making section that describes the emergency procedure. This ensures new members will learn these practices during onboarding, and it institutionalizes the security improvements.

- Set Review Milestones: Before closing this step, decide when you will check back on these mitigations. It could be:

  - A quick check-in at the next general meeting ("Did we do what we said for threat X? Is it working?").
  - A formal review after 3 or 6 months to see progress on longer tasks.
  - Incorporating it into an annual coop review or similar. By scheduling a follow-up, you create accountability and also acknowledge that mitigations may need tweaking.

4.2.7.3 Participation Tips

Mitigation planning might involve different sub-groups for different items, but make sure to bring it back to the full coop for transparency. For instance, let's say a tech group comes up with a plan for backups and multi-sig wallets – they should present it to others in plain language ("We propose to require two people to sign off on any expense over 1000€ – here's how that works... do we all agree?"). Use the coop's normal proposal-and-decision process to adopt significant changes, so they have legitimacy. Keep records of all these decisions (perhaps in a Security section of your meeting notes).

Also, acknowledge limitations: the group should be aware of any risks that you decide not to fully mitigate due to resource constraints. For example, "We know our website could be DDoS'ed (taken down by an overload attack), but at this time we can't afford a mitigation for that; we accept that risk and will focus on higher priorities." Being transparent about accepted risks is part of the security culture too.

By the end of Step 7, you have an action plan: who will do what to improve security, and the coop has agreed on these moves. Now the key is to implement and maintain these practices.

4.2.8 Step 8: Ongoing Improvement and Monitoring (How Do we Keep Alive Security?)

4.2.8.1 Purpose

Threat modeling isn't a one-time project – especially in a cooperative that evolves. Step 8 is about making security a continuous, integrated part of your coop's governance and operations. This means regularly monitoring for new threats, reviewing the effectiveness of your defenses, and adjusting to changes, all without losing the cooperative spirit. In short: build a living security program that grows and adapts with your organization.

4.2.8.2 Activities/Checklist

- Include Security in Routine Governance: Treat security as a standing topic in meetings. For example:
  - In your monthly general meeting or weekly check-in, have a brief agenda item like "Security and Privacy Check." This could be a quick report: "Any incidents or near-misses this month? Any new vulnerabilities found or new tools added that we should consider in our model?"
  - Quarterly or semi-annually, do a slightly deeper review. This could coincide with other reviews (financial, operational). Go over the threat list: have any new threats emerged? (Perhaps the coop started using a new platform – that introduces new risks to consider.) Are there threats that can be downgraded because of improvements? Basically, keep the threat model updated.
  - If you have a rotating security committee, they can take charge of preparing these updates. But even if not, someone can volunteer each time to gather relevant info.
- Maintain Logs and Audit Trails: Earlier we emphasized logging and traceability. Make sure those logs (system logs, decision logs, access records) are actually being looked at. For example:
  - If you set up an immutable log of administrative actions (like who changed what settings), maybe once a month a couple of members review it to see if all actions were expected and legit.
  - If you require multi-signature on an account, periodically check that the rule is being followed (no one found a shortcut).
  - Conduct an internal audit once a year: pick a couple of key policies and verify they are enforced. E.g., "Check 5 random user accounts to ensure they all have 2FA on as required," or "Verify that all ex-members in the past year have indeed been removed from our systems within the agreed 24 hours of departure." Such audits can be done by peers (any member can do it following a script) and

results shared openly: "We checked and found 1 of 5 accounts didn't have 2FA; we helped that member set it up."

This might sound formal, but it can be lightweight and it greatly helps catch when practices slip. Because organizations have no boss checking up, the members collectively check up in a transparent way.

- Periodic Retraining and Onboarding: Over time, members may forget certain security practices or new people join who weren't part of the initial threat modeling. Address this by:
  - Creating a short "Security Handbook" or section in your member handbook that summarizes key policies ("We use a shared password manager; here's how to get access," "Never share your login links with others," "Report any lost device immediately," etc., and also the why – referencing threats like "to prevent incidents like [scenario]" which makes it relatable).
  - Holding a yearly refresher workshop or sending a summary of "Our coop's top 5 threats and how we mitigate them – reminders for all members."
  - Simulating a phishing email test or an impromptu drill (with consent) to keep awareness sharp, then discussing the results in a blameless way ("Hey, 3 of us clicked that fake link – let's go over how to spot those signs!").
  - Updating onboarding processes: when new members join, include a quick orientation on security norms (for instance, how accounts are created, who to ask for tech help, basic do's and don'ts). If they know from day one that "security is everyone's responsibility here," it sets a good culture.
- Adapt to Changes in the Coop: If your cooperative grows from 5 members to 50, or starts a new project, revisit the threat model more formally. New scale or activities can mean new threats:
  - With more members, the risk of insider issues might increase simply by volume, and communication overhead grows – maybe you need more structured processes or automation.
  - With new projects (say you start handling money for members, or you open a second location), add those assets and processes to the model and run through the threat steps again in a lighter form.
  - Schedule a full threat modeling redo or refresh perhaps annually or bi-annually. It doesn't have to start from scratch each time; you can use the previous notes and just update. But having a cycle ensures that, for example, in two years you're not using the exact same assumptions while the world changed around you.

- Engage with the Wider Community: Horizontal organizations can learn a lot from each other. Consider sharing non-sensitive parts of your security approach with other cooperatives or participating in forums/discussion groups about security in decentralized organizations. You might discover common threats others have seen, or tools specifically helpful for organizations (for example, an open source tool for consensus that has anti-Sybil features, etc.). By contributing your experiences (without giving away your secrets), you also help the movement as a whole become more secure. This external input can be brought back to your coop's next threat modeling session ("We heard another coop had an issue with X, should we consider that scenario for us?").

- Monitor and Respond to Evolving Threats: Stay informed about general cybersecurity news that might affect you (e.g., a new vulnerability in a software you use – ensure someone updates it; a wave of phishing targeting small nonprofits – maybe alert members to be extra careful). Since no one person is "the security officer," create channels where such information can be shared. Maybe a security Slack/Matrix channel or an email list for anyone who finds something relevant. Collective eyes can watch more ground.

- Governance Feedback Loop: Evaluate how well the balance of democracy and efficiency in your security process is working:

  - After any incident or even after a year, ask: Did our emergency response plan work? Did anyone feel left out of the loop unnecessarily? If so, adjust the protocol.
  - Are members generally comfortable with the security measures or is there grumbling about complexity? If many find it too cumbersome (say the multi-sig process is taking too long for routine tasks), maybe tweak the thresholds or invest in a smoother tool. Always weigh these changes against the risk: maybe it's fine to ease up if the threat is low, or maybe the group agrees the inconvenience is worth the protection.
  - Essentially, treat the security policies themselves as living documents subject to the coop's governance. Amend them when needed, just like you would bylaws or other procedures, through the collective decision process.

4.2.8.3 Participation Tips

Keeping momentum is often the hardest part – enthusiasm is high during initial modeling, but can fade. To counter this, make security part of the coop's culture. Celebrate successes: "We successfully thwarted a phishing attempt because Maria remembered the training – kudos!" or "Our audit showed 100compliance on backups – great job team!" Positive reinforcement shows that these efforts matter and work. Also, distribute the load: rotate

who leads the security check-in, who runs the next training game, etc., so it's not always one person driving it (avoid creating a de facto security czar). The more members have a role over time, the more ingrained it becomes.

4.3 Security and Governance Requirements

The proposed protocol for horizontal organizations must meet security and governance requirements that ensure both participatory inclusion and robustness against internal and external threats. In decentralized structures, where the absence of formal hierarchy can be seen as both an advantage and a challenge, it is imperative that the protocol be designed to integrate mechanisms that preserve horizontality without compromising organizational resilience. Transparency, in this context, emerges as one of the central pillars [9]. All events related to the authorization of actions or modifications must be recorded in an immutable manner, ensuring that actions can be reliably audited by any member of the organization. These immutable records, built on technologies such as blockchain or similar cryptographic structures, ensure traceability and eliminate the possibility of unauthorized changes, maintaining cohesion between organizational principles and technological mechanisms.

Democratic participation also plays a fundamental role in the design of the protocol [9]. In a horizontal organization, decisions must reflect the collective will, and to this end, the protocol must integrate digital voting systems that guarantee both the privacy and security of votes [9]. In addition, it must be possible to temporarily delegate authority to individuals or groups to deal with situations that require specific expertise, always ensuring that such delegation can be revoked and that records of actions are accessible for collective audit [9].

Another crucial aspect is the flexibility of the protocol, especially in scenarios where the organization needs to switch between centralized and distributed governance modes. This transition capacity must be implemented in a way that the levels of centralization are temporary and properly tracked, ensuring that control can quickly return to the collective. To this end, the protocol must provide mechanisms such as emergency tokens, which allow the execution of critical actions in exceptional situations, as long as such actions are recorded and subject to retroactive validation by the members of the organization [9].

For the protocol to be scalable, it is essential that it supports organizations of different sizes and degrees of complexity. This includes implementing distributed validation systems that allow collaborative verification of actions without burdening individuals or single points of control [9]. Furthermore, threat modeling should be iterative and adaptive, incorporating techniques such as simulations based on real-world scenarios to identify emerging vulnerabilities and adjust countermeasures accordingly. These mechanisms ensure that the protocol remains functional and effective even when the organizational scale increases or its internal dynamics change.

4.4 Evaluation Strategy

The evaluation of the proposed protocol will be conducted through an experimental and comparative approach, involving real case studies in organizations with different degrees of horizontality. The main objective is to validate the effectiveness of the protocol in identifying, mitigating and preventing threats in horizontal organizational contexts, directly comparing its performance with established frameworks, such as Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, Elevation of privilege (STRIDE).

Candidate organizations will be selected to represent a variety of horizontal structures, ensuring diversity in power distribution, size and operational complexity. Participants from each organization will receive structured training sessions covering both the proposed protocol and STRIDE, ensuring familiarity and a level playing field for comparison.

Parallel Threat Modeling Sessions: Each organization will conduct simultaneous threat modeling sessions using both the proposed protocol and STRIDE in identical scenarios. These sessions will be observed to systematically document the process and collect data, analyzed according to the following clearly defined metrics:

- **Precision**: The ratio of valid threats correctly identified to the total number of threats identified by the protocol.
- **Coverage (Recall)**: The proportion of existing threats identified by the protocol relative to a reference set established by experts.
- **Operational Efficiency**: Total time (latency) required to complete each phase of the threat modeling process, including identification, analysis, and mitigation planning.
- **Usability**: Participants' opinions on the ease of use, clarity, and overall effort required to learn and effectively apply the protocol.

By systematically applying these clearly defined metrics and evaluation steps, this strategy ensures a comprehensive comparative analysis, providing robust empirical evidence on the effectiveness of the protocol relative to STRIDE in horizontal organizations.

Chapter 4 presented the preliminary design of a protocol specifically aimed at threat modeling in horizontal organizations, detailing fundamental security and governance requirements, such as mechanisms for democratic participation, temporary delegation of authority, and auditable records. The importance of aligning technological and organizational processes to ensure robustness against internal and external threats was highlighted. Based on this conceptual development, Chapter 5 summarizes the main results achieved by this research, critically evaluating the proposed protocol, identifying its contributions in relation to traditional methodologies, and exploring practical scenarios for future validations in real environments.

5

Evaluation

6

Conclusion

This research proposes a threat modeling protocol adapted to the specificities of horizontal organizations, integrating security and distributed governance as strategic elements [9]. The work is based on the recognition that non-hierarchical structures face unique challenges in security, from the informal centralization of digital resources to vulnerability to attacks that exploit participatory processes [42, 46]. By aligning horizontality with collaborative security practices, the protocol seeks to transform decentralization into an asset, mitigating critical points of failure without compromising collective autonomy.

A critical analysis of traditional methodologies, such as Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, Elevation of privilege (STRIDE) and attack trees, revealed gaps in the approach to non-hierarchical dynamics [38, 35]. While these frameworks are effective in centralized contexts, their dependence on formal hierarchies and linear decision flows limits their applicability in distributed environments. On the other hand, emerging approaches such as Collective based access control system (COLBAC) and Asset-Based Cryptocurrency (ABC) have demonstrated potential to fill these gaps by incorporating transparent consensus mechanisms and incentive-driven economic analyses [9, 1]. These solutions inspired the modular structure of the proposed protocol, which combines collaborative cryptography, immutable ledgers, and democratic authorization processes [9, 38].

The main contribution of this work lies in the integration of technical security and participatory governance. The protocol not only identifies specific threats to horizontal organizations, such as quorum manipulation, Sybil attacks, and centralization of secrets, but also establishes guidelines to mitigate them through distributed mechanisms [9, 46]. For example, the adoption of auditable logs and verifiable digital signatures reinforces transparency, while secure voting systems and dynamic delegation of authority preserve decision-making agility [9]. This approach balances operational efficiency and inclusiveness, allowing organizations to adapt their level of horizontality according to the context, whether in crisis situations that require temporary centralization or in fully decentralized day-to-day operations.

The defined evaluation criteria, effectiveness, efficiency, user acceptance and resilience,

provide a robust framework to validate the protocol in different scenarios [47]. Case studies with worker cooperatives and community networks will allow testing its practical applicability, while simulations of democratic attacks and consensus failures will assess its robustness [42]. The results are expected to demonstrate how horizontality, when structured coherently, can strengthen security by distributing responsibilities and reducing critical dependencies [9, 42].

Bibliography

[1] G. Almashaqbeh, A. Bishop, and J. Cappos. "ABC: A Cryptocurrency-Focused Threat Modeling Framework". In: *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. 2019, pp. 859–864. DOI: 10.1109/INFCOMW.2019.8845101 (cit. on pp. 1, 2, 7, 18–21, 49).

[2] C. W. Blackwell. "Athenian Democracy: An Overview". In: *Dēmos: Classical Athenian Democracy*. Ed. by C. W. Blackwell. www.stoa.org: The Stoa: A Consortium for Electronic Publication in the Humanities, 2003. URL: http://www.stoa.org (cit. on pp. 2, 9).

[3] J. Cleland-Huang. "How Well Do You Know Your Personae Non Gratae?" In: *IEEE Software* 31.4 (2014), pp. 28–31. DOI: 10.1109/MS.2014.85 (cit. on p. 16).

[4] J. Cleland-Huang et al. "Keeping Ahead of Our Adversaries". In: *IEEE Software* 33.3 (2016), pp. 24–28. DOI: 10.1109/MS.2016.75 (cit. on p. 16).

[5] N. Couldry and U. A. Mejias. *The Costs of Connection: How Data Is Colonizing Human Life and Appropriating It for Capitalism*. Stanford, California: Stanford University Press, 2019. ISBN: 9781503609754. URL: https://lccn.loc.gov/2019010213 (cit. on p. 11).

[6] T. Denning, B. Friedman, and T. Kohno. *The Security Cards: A Security Threat Brainstorming Toolkit*. Accessed: 2024-12-09. 2013. URL: http://securitycards.cs.washington.edu/assets/security-cards-information-sheet.pdf (cit. on pp. 1, 7, 16).

[7] J. R. Douceur. "The Sybil Attack". In: *Peer-to-Peer Systems*. Ed. by P. Druschel, F. Kaashoek, and A. Rowstron. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 251–260. ISBN: 978-3-540-45748-0 (cit. on p. 1).

[8] *Estatutos da Confederação Geral dos Trabalhadores Portugueses – Intersindical Nacional: Declaração de Princípios e Objectivos Programáticos*. Acesso em: 08 dez. 2024. Confederação Geral dos Trabalhadores Portugueses (CGTP), 2020. URL: https://www.cgtp.pt/images/images/2020/02/ESTATUTOSCGTP.pdf (cit. on p. 10).

[9] K. Gallagher et al. "COLBAC: Shifting Cybersecurity from Hierarchical to Horizontal Designs". In: *Proceedings of the 2021 New Security Paradigms Workshop*. NSPW '21. Virtual Event, USA: Association for Computing Machinery, 2022, pp. 13–27. ISBN: 9781450385732. DOI: 10.1145/3498891.3498903. URL: https://doi.org/10.1145/3498891.3498903 (cit. on pp. 1–4, 6–11, 15, 17, 18, 20, 21, 46, 49, 50).

[10] P. Gerbaudo. "Social media teams as digital vanguards: The question of leadership in the management of key Facebook and Twitter accounts of Occupy Wall Street, Indignados and UK Uncut". In: *Information, Communication & Society* 20.2 (2017), pp. 185–202. DOI: 10.1080/1369118X.2016.1161817. URL: https://doi.org/10.1080/1369118X.2016.1161817 (cit. on pp. 1, 9).

[11] P. Herbst. "Non-Hierarchical Forms of Organization". In: *Acta Sociologica* 19.1 (1976), pp. 65–75. DOI: 10.1177/000169937601900106. URL: https://doi.org/10.1177/000169937601900106 (cit. on pp. 2, 4, 7, 8).

[12] S. Hernan et al. *Uncover Security Design Flaws Using The STRIDE Approach*. Accessed: 2024-Dec-09. 2006-11. URL: https://learn.microsoft.com/en-us/archive/msdn-magazine/2006/november/uncover-security-design-flaws-using-the-stride-approach (cit. on pp. 12, 13).

[13] M. Howard and S. Lipner. *The Security Development Lifecycle: SDL, a Process for Developing Demonstrably More Secure Software*. Secure software development series. Microsoft Press, 2006. ISBN: 978-07356-2214-2 (cit. on p. 13).

[14] S. Hussain et al. "Threat Modelling Methodologies: A Survey". In: vol. 26. 2014-01, pp. 1607–1609. URL: https://api.semanticscholar.org/CorpusID:111533730 (cit. on p. 2).

[15] R. Jackall and H. M. Levin, eds. *Worker Cooperatives in America*. Berkeley and Los Angeles, California: University of California Press, 1984. ISBN: 0-520-05117-3 (cit. on pp. 2, 4, 7, 8).

[16] R. Jiang, J. Luo, and X. Wang. "An Attack Tree Based Risk Assessment for Location Privacy in Wireless Sensor Networks". In: *2012 8th International Conference on Wireless Communications, Networking and Mobile Computing*. 2012, pp. 1–4. DOI: 10.1109/WiCOM.2012.6478402 (cit. on p. 14).

[17] R. Jiang et al. "Energy-theft detection issues for advanced metering infrastructure in smart grid". In: *Tsinghua Science and Technology* 19.2 (2014), pp. 105–120. DOI: 10.1109/TST.2014.6787363 (cit. on pp. 2, 14).

[18] A. Kavada. "Creating the collective: social media, the Occupy Movement and its constitution as a collective actor". In: *Information, Communication & Society* 18.8 (2015), pp. 872–886. DOI: 10.1080/1369118X.2015.1043318. eprint: https://doi.org/10.1080/1369118X.2015.1043318. URL: https://doi.org/10.1080/1369118X.2015.1043318 (cit. on p. 4).

[19] A. Kavada and T. Poell. "From Counterpublics to Contentious Publicness: Tracing the Temporal, Spatial, and Material Articulations of Popular Protest Through Social Media". In: *Communication Theory* 31.2 (2020-10), pp. 190–208. ISSN: 1050-3293. DOI: 10.1093/ct/qtaa025. eprint: https://academic.oup.com/ct/article-pdf/31/2/190/37900340/qtaa025.pdf. URL: https://doi.org/10.1093/ct/qtaa025 (cit. on p. 1).

[20] R. Khan et al. "STRIDE-based threat modeling for cyber-physical systems". In: *2017 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*. 2017, pp. 1–6. DOI: 10.1109/ISGTEurope.2017.8260283 (cit. on pp. 7, 13, 20).

[21] J. W. Kuyper and J. S. Dryzek. "Real, not nominal, global democracy: A reply to Robert Keohane". In: *International Journal of Constitutional Law* 14.4 (2017-01), pp. 930–937. ISSN: 1474-2640. DOI: 10.1093/icon/mow063. eprint: https://academic.oup.com/icon/article-pdf/14/4/930/9607155/mow063.pdf. URL: https://doi.org/10.1093/icon/mow063 (cit. on p. 7).

[22] D. LeBlanc. *DREADful*. Accessed: 2024-Dec-09. 2007-08. URL: https://learn.microsoft.com/en-us/archive/blogs/david_leblanc/dreadful (cit. on p. 13).

[23] E. Marasco et al. "Attack Trees for Protecting Biometric Systems Against Evolving Presentation Attacks". In: *16th Annual IEEE International Conference on Technologies for Homeland Security (HST) 2017*. 2017 (cit. on p. 16).

[24] S. Mauw and M. Oostdijk. "Foundations of Attack Trees". In: *Information Security and Cryptology - ICISC 2005*. Ed. by D. H. Won and S. Kim. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 186–198. ISBN: 978-3-540-33355-5 (cit. on p. 14).

[25] N. Mead and F. Shull. *The Hybrid Threat Modeling Method*. Carnegie Mellon University, Software Engineering Institute's Insights (blog). Accessed: 2025-Feb-11. 2018-04. URL: https://insights.sei.cmu.edu/blog/the-hybrid-threat-modeling-method/ (cit. on pp. 6, 17).

[26] N. Mead et al. "Crowd Sourcing the Creation of Personae Non Gratae for Requirements-Phase Threat Modeling". In: *2017 IEEE 25th International Requirements Engineering Conference (RE)*. 2017, pp. 412–417. DOI: 10.1109/RE.2017.63 (cit. on pp. 16, 17).

[27] E. Morozov. *Big Tech: A Ascensão dos Dados e a Morte da Política*. Ed. by R. Lemos. São Paulo: Ubu Editora, 2018. ISBN: 978-85-7126-012-2 (cit. on pp. 8, 9).

[28] S. Myagmar, A. J. Lee, and W. Yurcik. "Threat modeling as a basis for security requirements". In: (2005) (cit. on p. 6).

[29] P. Nancy R. Mead. *Advanced Threat Modeling (ATM)*. Tech. rep. Pittsburgh, PA 15213: Software Engineering Institute, Carnegie Mellon University, 2017. URL: https://apps.dtic.mil/sti/trecms/pdf/AD1089727.pdf (cit. on p. 6).

[30] P. Zimmermann. *PGP User's Guide, Volume I: Essential Topics*. Revised Edition. PGP Version 2.6.2. Massachusetts Institute of Technology: Phil's Pretty Good Software, 1994. URL: https://web.pa.msu.edu/reference/pgpdoc1.html (cit. on pp. 19, 20).

[31] P. C. Português. *Programa e Estatutos do PCP*. Revisão tipográfica: Edições «Avante!». Lisboa, Portugal, 2013. URL: https://www.pcp.pt/estatutos-do-pcp (cit. on pp. 8, 10).

[32] B. Potteiger, G. Martins, and X. Koutsoukos. "Software and attack centric integrated threat modeling for quantitative risk assessment". In: *Proceedings of the Symposium and Bootcamp on the Science of Security*. HotSos '16. Pittsburgh, Pennsylvania: Association for Computing Machinery, 2016, pp. 99–108. ISBN: 9781450342773. DOI: 10.1145/2898375.2898390. URL: https://doi.org/10.1145/2898375.2898390 (cit. on p. 13).

[33] Y. Saito and J. A. Rose. "Reputation-based Decentralized Autonomous Organization for the Non-Profit Sector: Leveraging Blockchain to Enhance Good Governance". In: *Frontiers in Blockchain* 5 (2023). ISSN: 2624-7852. DOI: 10.3389/fbloc.2022.1083647. URL: https://www.frontiersin.org/articles/10.3389/fbloc.2022.1083647 (cit. on pp. 2, 3, 7).

[34] R. Scandariato, K. Wuyts, and W. Joosen. "A Descriptive Study of Microsoft's Threat Modeling Technique". In: *Requirements Engineering* 20.2 (2015-06), pp. 163–180. ISSN: 1432-010X. DOI: 10.1007/s00766-013-0195-2. URL: https://doi.org/10.1007/s00766-013-0195-2 (cit. on pp. 1, 7, 8, 12).

[35] B. Schneier. *Attack Trees*. Tech. rep. 12. 1999. URL: https://tnlandforms.us/cs594-cns96/attacktrees.pdf (cit. on pp. 1, 7, 14, 21, 49).

[36] N. Shevchenko et al. "Threat modeling: a summary of available methods". In: *Software Engineering Institute | Carnegie Mellon University* (2018), pp. 1–24 (cit. on pp. 1, 2, 6–8).

[37] A. Shostack. "Experiences Threat Modeling at Microsoft". In: *MODSEC@ MoDELS* (2008) (cit. on p. 6).

[38] A. Shostack. *Threat Modeling: Designing for Security*. Available in print and electronic formats. Indianapolis, Indiana: John Wiley & Sons, Inc., 2014. ISBN: 978-1-118-80999-0 (cit. on pp. 1, 3, 4, 6, 7, 12–14, 20, 21, 49).

[39] F. Shull and N. Mead. *Cyber Threat Modeling: An Evaluation of Three Methods*. Carnegie Mellon University, Software Engineering Institute's Insights (blog). Accessed: 2024-Dec-9. 2016-11. URL: https://insights.sei.cmu.edu/blog/cyber-threat-modeling-an-evaluation-of-three-methods/ (cit. on p. 16).

[40] F. Shull et al. *Evaluation of Threat Modeling Methodologies*. Tech. rep. Carnegie Mellon University, Software Engineering Institute, 2016-10. URL: https://insights.sei.cmu.edu/documents/4027/2016_017_001_474200.pdf (cit. on pp. 1, 7, 20).

[41] S. A. da Silveira. *Democracia e os códigos invisíveis: como os algoritmos estão modulando comportamentos e escolhas políticas*. São Paulo: Edições Sesc São Paulo, 2019. ISBN: 978-85-9493-180-1 (cit. on pp. 3, 9).

[42] M. A. Sitrin. *Everyday Revolutions: Horizontalism and Autonomy in Argentina*. London, UK; New York, USA: Zed Books Ltd, 2012. ISBN: 9781780320502 (cit. on pp. 2–4, 8, 9, 18, 20, 21, 49, 50).

[43] J. Slupska et al. "Participatory Threat Modelling: Exploring Paths to Reconfigure Cybersecurity". In: *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*. CHI EA '21. Yokohama, Japan: Association for Computing Machinery, 2021. ISBN: 9781450380959. DOI: 10.1145/3411763.3451731. URL: https://doi.org/10.1145/3411763.3451731 (cit. on pp. 3, 6, 7, 15–17).

[44] P. M. Thornton. "Of Constitutions, Campaigns and Commissions: A Century of Democratic Centralism under the CCP". In: *The China Quarterly* 248.S1 (2021), pp. 52–72. DOI: 10.1017/S0305741021000758 (cit. on pp. 10, 11).

[45] P. Torr. "Demystifying the threat modeling process". In: 3.5 (2005), pp. 66–70. DOI: 10.1109/MSP.2005.119 (cit. on pp. 1, 6, 7).

[46] Z. Trifa and M. Khemakhem. "Sybil Nodes as a Mitigation Strategy Against Sybil Attack". In: *Procedia Computer Science* 32 (2014). The 5th International Conference on Ambient Systems, Networks and Technologies (ANT-2014), the 4th International Conference on Sustainable Energy Information Technology (SEIT-2014), pp. 1135–1140. ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2014.05.544. URL: https://www.sciencedirect.com/science/article/pii/S1877050914007443 (cit. on pp. 1, 3, 49).

[47] T. UcedaVelez and M. M. Morana. *Risk Centric Threat Modeling: Process for Attack Simulation and Threat Analysis*. John Wiley & Sons, 2015-05, p. 696. ISBN: 978-0-470-50096-5 (cit. on pp. 1, 14, 15, 20, 50).

[48] J. Von Der Assen et al. "CoReTM: An Approach Enabling Cross-Functional Collaborative Threat Modeling". In: *2022 IEEE International Conference on Cyber Security and Resilience (CSR)*. 2022, pp. 189–196. DOI: 10.1109/CSR54599.2022.9850283 (cit. on pp. 7, 17).

[49] L. Winner. "Do Artifacts Have Politics?" In: *Daedalus* 109.1 (1980), pp. 121–136. ISSN: 00115266. URL: http://www.jstor.org/stable/20024652 (visited on 2024-12-08) (cit. on pp. 3, 8–10).

[50] C. Wright. *Worker Cooperatives and Revolution: History and Possibilities in the United States*. Bradenton, Florida, USA: BookLocker.com, Inc., 2014. ISBN: 978-1-63263-432-0 (cit. on pp. 2, 8, 9).

[51] W. Xiong and R. Lagerström. "Threat modeling - A systematic literature review". In: 84 (2019), pp. 53–69. ISSN: 0167-4048. DOI: https://doi.org/10.1016/j.cose.2019.03.010. URL: https://www.sciencedirect.com/science/article/pii/S0167404818307478 (cit. on pp. 1, 8, 14, 15, 20, 21).

[52] G. Yang. "Still a Century of the Chinese Model? Exploring Dimensions of Democratic Centralism". In: *Chinese Political Science Review* 1.1 (2016), pp. 171–189. DOI: 10.1007/s41111-016-0005-3. URL: https://doi.org/10.1007/s41111-016-0005-3 (cit. on pp. 10, 11).