U.PORTO
FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO

PhD

PhD
3rd CYCLE
FCUP
2026

U.PORTO

SkyNet: Towards a Dynamic and Adaptive Intrusion Tolerant System

Tadeu Freitas

FC

U.PORTO
FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO

SkyNet: Towards a Dynamic and Adaptive Intrusion Tolerant System

Tadeu Freitas
Doctoral Program in Computer Science
Department of Computer Science
Faculty of Sciences of the University of Porto
2026

PhD

SkyNet: Towards a Dynamic and Adaptive Intrusion Tolerant System

Tadeu Freitas
Thesis carried out as part of the Doctoral Program in Computer Science
Department of Computer Science
2026

Supervisor
Prof. Dr. Rolando Martins, Assistant Professor, Faculty of Sciences of the University of Porto

Co-supervisor
Prof. Dr. Manuel E. Correia, Associate Professor, Faculty of Sciences of the University of Porto

Dedicated to my brother.

“ The journey begins where the road ends. ”

Author unknown (seen on a whiteboard at the Arctic University of Norway)

Acknowledgements

[PT]

Quero expressar a minha profunda gratidão à minha família, cujo apoio inabalável e crença em mim têm sido fundamentais nesta caminhada. A o meu irmão e aos meus pais, agradeço pelo amor constante e pela confiança depositada em mim.

O meu sincero agradecimento estende-se também a todos os meus amigos que me acompanharam ao longo deste percurso, com especial destaque para o Nuno Silva, o Pedro Ângelo, o João Soares e a Eduarda Almeida, cujo contributo, motivação e perspetiva foram inestimáveis.

Há ainda aqueles cujo apoio silencioso e compreensão, muitas vezes de forma discreta, fizeram uma diferença profunda nos momentos em que a perseverança mais se exigia. Ainda que a sua presença já não seja tão próxima, o incentivo e a orientação que me transmitiram continuam vivos e ressoam em mim, inspirando-me e sustentando a minha perseverança.

Todo o meu trabalho ao longo dos últimos quatro anos contou com o apoio da Fundação para a Ciência e a Tecnologia (FCT), através de uma bolsa de doutoramento, pelo que agradeço o apoio concedido.

Finalmente, estou profundamente grato aos meus orientadores, o Prof. Rolando Martins e o Prof. Manuel E. Correia, cuja orientação, paciência e encorajamento foram fundamentais para a concretização desta tese.

[EN]

I would like to express my deepest gratitude to my family, whose unwavering support and belief have been the cornerstone of this journey. To my brother and my parents, thank you for your continuous love and confidence in me.

My heartfelt thanks also go to all my friends who have supported me along the way, with a special mention to Nuno Silva, Pedro Ângelo, João Soares, and Eduarda Almeida, whose insight and motivation have been invaluable.

There are those whose quiet support and understanding, often behind the scenes, have made a profound difference in moments when perseverance was most needed. Though their presence is now missed, the gentle encouragement and guidance they offered continues to resonate deeply, inspiring me and sustaining my perseverance.

All my work over the last four years was supported by Fundação para a Ciência e a Tecnologia (FCT) through a PhD scholarship, so I am grateful for their support.

Finally, I am deeply indebted to my supervisors, Prof. Rolando Martins and Prof. Manuel E. Correia, whose expert guidance, patience, and encouragement have been vital to the successful completion of this thesis.

Resumo

A tese apresenta o Skynet, uma arquitetura nova para Intrusion Tolerant System (ITS) adaptativa e dinâmica, projetada para resistir e mitigar ameaças cibernéticas cada vez mais sofisticadas e multidomínio, integrando conceitos de cibersegurança, Artificial Intelligence (AI) e sistemas distribuídos. Abordando as limitações das defesas perimetrais tradicionais e dos protocolos convencionais Byzantine Fault Tolerance (BFT), o Skynet reconhece a inevitabilidade das violações, especialmente provenientes de Advanced Persistent Threats (APTs), e foca em manter a resiliência do sistema através da diversidade, redundância, recuperação proativa e reconfiguração dinâmica. A arquitetura distingue entre um Plano de Controlo confiável e seguro—que gere a avaliação de riscos, configuração e orquestração utilizando módulos impulsionados por AI, como o gestor de risco HAL 9000 (HAL)—e um Plano de Execução não confiável, exposto a adversários. A tese detalha a integração do Skynet na recolha de informação de ameaças em tempo real (via o módulo Vulnerability Exploit Hunter Killer (VExHK)), pontuação de vulnerabilidades baseada em Machine Learning (ML), e deteção de intrusões aprimorada por Federated Learning (FL) para otimizar configurações do sistema, minimizando o risco de segurança e maximizando a resiliência a comprometimentos correlacionados. Sistemas complementares como o Electric Vehicle Security orchestration, automation, and response (EVSOAR), que utiliza a infraestrutura de carregamento de Electric Vehicles (EV) para uma deteção e resposta a intrusões segura e com baixa latência, e o Linked Entities for Governance of Intrusion and Operational Norms of Intrusion Tolerant Systems (LegionITS), uma arquitetura federada de ITS que permite o partilha de inteligência contra ciberameaças entre organizações preservando a privacidade, demonstram a aplicabilidade prática e escalabilidade da arquitetura proposta. Avaliações experimentais destacam a superioridade do HAL na redução das janelas de exposição ao risco, através de pontuações automáticas e preditivas de vulnerabilidades e avaliação de risco, enquanto as avaliações de rede do EVSOAR mostram melhorias significativas na latência, capacidade de throughput e eficácia de deteção em comparação com operações tradicionais de segurança veicular. A abordagem federada do LegionITS aborda os desafios de privacidade, escalabilidade e confiança inerentes à colaboração multi-organizações. A tese defende que um design multidisciplinar que integra protocolos de consenso BFT, gestão adaptativa de risco impulsionada por AI, colaboração federada preservadora da privacidade

e orquestração automatizada constitui uma base poderosa para arquiteturas de cibersegurança resilientes de próxima geração, capazes de proteger infraestruturas críticas e sistemas complexos contra ameaças cibernéticas em evolução.

Abstract

The thesis presents Skynet, a novel architecture for adaptive, dynamic Intrusion Tolerant System (ITS) designed to withstand and mitigate increasingly sophisticated, multi-domain cyber threats by integrating concepts from cybersecurity, Artificial Intelligence (AI), and distributed systems. Addressing the limitations of traditional perimeter defenses and conventional Byzantine Fault Tolerance (BFT) protocols, Skynet acknowledges the inevitability of breaches, particularly from Advanced Persistent Threats (APTs), and focuses on maintaining system resilience through diversity, redundancy, proactive recovery, and dynamic reconfiguration. The architecture distinguishes between a trusted, secure Control Plane—which manages risk assessment, configuration, and orchestration using AI-driven modules such as the HAL 9000 (HAL) risk manager—and an untrusted Execution Plane exposed to adversaries.

The thesis elaborates on Skynet's integration of real-time threat intelligence gathering (via the Vulnerability Exploit Hunter Killer (VExHK) module), Machine Learning (ML)-based vulnerability scoring, and Federated Learning (FL)-enhanced intrusion detection to optimize system configurations for both minimal security risk and maximal resilience to correlated compromises. Complementary systems such as Electric Vehicle Security orchestration, automation, and response (EVSOAR), which leverages Electric Vehicles (EV) charging infrastructure for secure, low-latency intrusion detection and response, and Linked Entities for Governance of Intrusion and Operational Norms of Intrusion Tolerant Systems (LegionITS), a federated ITS architecture enabling privacy-preserving cyber threat intelligence sharing across organizations, demonstrate the practical applicability and scalability of the proposed architecture.

Experimental evaluations highlight HAL's superiority in reducing risk exposure windows through automated, predictive vulnerability scoring and risk assessment. In contrast, EVSOAR's network evaluations demonstrate significant improvements in latency, throughput, and detection efficacy compared to traditional vehicle security operations. The federated approach of LegionITS addresses privacy, scalability, and trust challenges inherent in multi-organization collaboration.

The thesis argues that a multidisciplinary design integrating BFT consensus protocols, AI-driven adaptive risk management, privacy-preserving federated collaboration, and automated orchestration forms a powerful basis for next-generation resilient cybersecurity

architectures, capable of protecting critical infrastructures and complex systems against evolving cyber threats.

Foreword

The work of this thesis was performed in the Center for Research in Advanced Computing Systems (CRACS) of the Institute for Systems and Computer Engineering, Technology and Science (INESC TEC), and financed by a PhD grant from FCT – Fundação para a Ciência e a Tecnologia, with the reference 2021.04529.BD.

Table of Contents

List of Figures

List of Tables

Terminology

Atomic Broadcast: A communication abstraction related to consensus, ensuring all correct replicas deliver the same set of messages in the same order.

Byzantine Faults: Failures where nodes in a distributed system act arbitrarily, including lying, collusion, or malicious behavior, making them harder to detect and mitigate than simple crash faults.

BFT: A class of distributed consensus and state-machine-replication protocols that keep a replicated service safe and available as long as no more than f of $3f + 1$ replicas behave arbitrarily (Byzantine faults), including when compromised by an adversary. BFT provides agreement and consistent ordering among replicas; it is a core *mechanism* for building intrusion-tolerant systems, but does not by itself constitute intrusion tolerance, which additionally requires diversity, proactive recovery, intrusion detection, and dynamic reconfiguration.

Consensus Algorithm [1] (also known as agreement or steady state): A protocol that allows all non-faulty nodes in a distributed system to agree on a single value, maintaining consistency even when some nodes exhibit Byzantine faults.

Complexity Metrics [2]: Measures such as message complexity, communication complexity, and step/round complexity used to evaluate protocols.

Cryptographic Primitives: Fundamental building blocks such as digital signatures, Message Authentication Codes (MACs), threshold cryptography, and verifiable random functions (VRFs) that ensure authenticity, integrity, and sometimes anonymity.

Diversity: The use of distinct versions of software or hardware components within a system to reduce the risk that a single vulnerability affects all components.

Equivocation: Behavior where a faulty node sends conflicting or inconsistent messages to different replicas. Traditional BFT protocols prevent this by constructing quorums that intersect in honest replicas. Protocols that use Trusted Components stop equivocation by relying on secure elements that adversaries cannot compromise.

DP [3]: A privacy technique that adds statistical noise to data or model updates to prevent information leakage.

FL [4]: A decentralized machine learning approach where models are trained across multiple devices without sharing raw data, exchanging only model updates to enhance privacy.

Intrusion Tolerance [5]: The ability of a system to operate correctly even when some components are compromised by attackers. Unlike traditional security, which focuses only on prevention, intrusion tolerance assumes breaches occur and emphasizes resilience and mitigation.

Homomorphic Encryption (HE) [6]: A cryptographic method allowing computations on encrypted data without decrypting it, preserving privacy.

Quorum: A subset of nodes whose collective agreement is required for consensus decisions. In BFT systems, a quorum is usually at least $2f + 1$ out of $3f + 1$ nodes, where f is the maximum tolerated faulty nodes.

Redundancy: The intentional duplication of critical components or functions to increase system reliability and availability even if some components fail or are attacked. Redundant parts may be active or standby.

Self-Healing: The ability of a system to automatically detect, isolate, and restore compromised components to recover from intrusions.

SMR [7]: A technique for ensuring consistency by replicating services across nodes and executing the same sequence of operations.

Threshold Cryptography: A cryptographic scheme where a secret is divided among nodes, requiring cooperation of a threshold subset to perform operations, enhancing security.

Trusted Component: A secure hardware or software element immune to Byzantine faults, used to improve fault tolerance or performance in BFT protocols. Trusted components perform simple tasks securely (e.g., signature verification), reducing replica requirements while preserving safety and liveness.

Unsupervised Learning: A type of machine learning where models identify patterns from unlabeled data without predefined outputs.

View Change: In BFT protocols, the process of switching to a new leader node when the current one is suspected faulty or compromised.

AI: The field of computer science focused on creating systems capable of tasks that require human intelligence, such as learning, reasoning, and problem-solving.

ML: A subset of AI where systems learn patterns from data to make predictions or decisions without explicit programming.

List of Abbreviations

AI Artificial Intelligence. v, vii, xi, xx, 1–4, 8, 17, 67, 69, 82, 92, 135, 137

API Application Programming Interface. 59, 61, 62, 73, 80, 81, 86, 91, 92, 127, 130

APT Advanced Persistent Threat. v, vii, 1, 2, 4, 12, 35, 117, 121, 122, 135

ATT&CK Adversarial Tactics, Techniques, and Common Knowledge. 4, 117

BFT Byzantine Fault Tolerance. v, vii, xi, xv, xvii–xix, 1–4, 8, 13, 17–33, 35–40, 43–45, 54, 56, 57, 67, 68, 135, 137

CIS Cybersecurity Information Sharing. 7, 52, 112–114, 136

CSN Charging Station Network. 9, 95, 98, 99, 101–104, 108, 136

CTI Cyber Threat Intelligence. 2, 7, 111–117, 119, 120, 122, 130, 136

CVE Common Vulnerabilities and Exposures. xv, 4, 5, 43, 44, 65, 66, 74–76, 81–85, 92

CVSS Common Vulnerability Scoring System. xii, xiii, 4, 5, 44, 45, 54, 65–67, 69, 70, 73–78, 81–85, 89, 90, 92, 136

DL Deep Learning. 69, 70, 76, 84, 89

DP Differential Privacy. xvii, xviii, 3, 7, 112–115, 119, 120, 122, 123, 125, 126, 131, 133

EPSS Exploit Prediction Scoring System. 4, 5, 44, 77, 81, 85, 93

EV Electric Vehicles. v, vii, 6, 9, 95, 97–99, 101–108, 136

EVSOAR Electric Vehicle Security orchestration, automation, and response. v, vii, xiii, xvi, 6, 9, 95–99, 101–109

ExploitDB Exploit Database. 1, 12, 20, 65, 69, 79, 80, 83, 86, 90–92

FL Federated Learning. v, vii, xvii, xix, 6–9, 17, 95, 96, 100–102, 104–108, 111, 112, 114, 115, 119, 122–124, 126, 129, 133, 136

HAL HAL 9000. v, vii, xii, xv, xvi, 5, 9, 10, 43, 44, 52, 54–57, 59, 60, 62, 65–67, 69, 71, 73–75, 77–83, 85, 87, 89–94, 135, 136

HE Homomorphic Encryption. xix, 3, 7, 113, 114

HIDS Host-based Intrusion Detection System. xv, 38, 42–44, 46, 47, 49, 54, 56, 57, 59, 62, 117, 121, 124

IDS Intrusion Detection System. 6, 7, 12, 21, 23, 95, 97–100, 102, 104–106, 117

IOC Indicators of Compromise. 72, 73, 114, 129, 131

ITS Intrusion Tolerant System. v, vii, xi, xvii, 1–5, 7–18, 21–23, 25, 27, 29, 31, 33–37, 39, 41, 43–45, 47, 49, 51, 53, 55, 57, 59, 61, 63, 65–67, 69, 73, 78, 79, 81–83, 90, 93–97, 108, 109, 111, 112, 114–117, 119–122, 126, 131, 132, 135, 136

LegionITS Linked Entities for Governance of Intrusion and Operational Norms of Intrusion Tolerant Systems. v, vii, xiii, xvi, 7, 9, 10, 111–133

MAC Message Authentication Code. xviii, 24, 29, 31

MHE Multiparty Homomorphic Encryption. 7, 112, 114, 119–122, 125, 126, 131, 133

MISP Malware Information Sharing Platform. xv, 43, 56–58, 112, 115, 126

ML Machine Learning. v, vii, xvii, xx, 2, 6, 8, 35, 36, 39–41, 44, 62, 65–69, 75, 76, 80–82, 84, 89, 92, 93, 95, 100–102, 104–108, 121, 135, 136

NIST National Institute of Standards and Technology. 4, 12, 39, 113

NLP Natural Language Processing. 44, 67, 69, 70, 84, 89

NVD National Vulnerability Database. 1, 12, 20, 43, 44, 65, 67, 69, 73, 75, 79, 80, 83, 86, 91

OEM Original Equipment Manufacturer. 9, 95, 97, 99, 100, 104, 105

OS Operating System. 5, 19, 20, 37, 38, 41, 43, 47, 50, 68, 78, 81, 83, 85, 117

OSINT Open Source Intelligence. xvi, 2, 4, 5, 8, 9, 44, 66, 69, 72, 74, 79, 80, 82, 83, 86, 89–93, 115

OSV Open Source Vulnerabilities. 69, 79, 80, 86, 90, 91

PLC Power Line Communication. 6, 9, 95, 99, 102, 103

RC3 Resilient Computing and Cybersecurity Center. 6, 95

SDN Software-Defined Networking. 35, 39, 46, 52

SIEM Security Information and Event Management. 4, 35, 36, 41, 46, 61, 97, 98, 111, 112, 117, 120, 124, 133

SMPC Secure Multi-Party Computation. 3, 7, 17

SMR State Machine Replica. xv, xix, 13, 23–28, 31, 33, 36

SOAR Security orchestration, automation, and response. 2, 4–6, 9, 95–102, 105, 106, 108, 111, 112, 120, 133, 136

SOC Security Operations Center. xi, 2, 5, 6, 46, 61, 97, 98, 112

STIX Structured Threat Information eXpression. 113, 114, 125, 130

TAXII Trusted Automated eXchange of Indicator Information. 113, 125, 130

TCB Trusted Computing Base. 18, 37, 41

TEE Trusted Execution Environment. 36–38, 40, 47, 48, 50, 135

TLS Transport Layer Security. 47, 59, 99

TTCB Trusted Timely Computing Base. xvii, 18, 19, 28

VExHK Vulnerability Exploit Hunter Killer. v, vii, xii, 5, 42, 66, 74, 80–83, 91, 93

VM Virtual Machine. 19, 67, 117

VSOC Vehicle Security Operations Center. xiii, 6, 9, 95, 97, 98, 101–103, 108

ZKP Zero-Knowledge Proof. 7, 112–114, 119–121, 125, 131, 133

1. Introduction

1.1. Context and Motivation

The digitalization of services and the rise of cyberattacks have exposed the limits of traditional perimeter-based defenses, which lack adaptive capabilities once breached. Modern systems' complexity requires resilient mechanisms that prioritize continuous service over pure prevention.

Intrusion Tolerant Systems (ITSs) address this challenge by counteracting unauthorized actions through redundancy, diversity, rejuvenation, and N-version programming. Originating from the DELTA-4 project [8] and formalized by Veríssimo et al. [9], ITS shift the focus from solely detecting intrusions to actively mitigating, recovering, and sustaining system availability.

Intrusion tolerance is a non-functional dependability property: the ability of a system to continue delivering correct service despite the partial compromise of some of its components by an adversary. An *intrusion-tolerant system* (ITS) is a system engineered to provide this property—typically through redundancy, diversity, and Byzantine Fault Tolerance (BFT) replication—so that the compromise of a bounded subset of components cannot subvert the system as a whole.

The emergence of Advanced Persistent Threats (APTs) [10], stealthy, targeted, and state-sponsored, has accelerated the evolution of ITS. Since APTs often bypass conventional defenses, ITS emphasizes resilience by isolating compromised components, employing self-healing, and leveraging multi-layer protection through consensus, intrusion detection, and dynamic reconfiguration. Foundational work, such as Castro's BFT model [11] and Bessani's Crutial Information Switch [5], laid the groundwork for these approaches.

However, the growing sophistication of attacks, including zero-day exploits used in incidents like Stuxnet [12], Operation Aurora [13], and the Equifax breach [14], highlights the shortcomings of traditional ITS assumptions. Vulnerabilities disclosed in repositories such as National Vulnerability Database (NVD) or Exploit Database (ExploitDB) can be exploited if unpatched, while adversaries exploit homogeneity in system nodes to compromise more than $f + 1$ replicas.

To strengthen resilience, ITS now combine heterogeneous configurations, layered defenses, and adaptive learning. By integrating threat intelligence, Artificial Intelligence (AI)-driven analytics, and robust cyber-hardening, modern ITS makes coordinated compromises

significantly harder, ensuring dependable service in adversarial environments.

1.2. Objectives and Contributions

Recent advancements in ITS have spurred increased research efforts, leading to a range of proposed solutions to enhance resilience against intrusions, as discussed in Chapter 2. However, the complexity and multidisciplinary strategies employed by contemporary and emerging APT groups, along with other malicious actors, necessitate reevaluation of ITS architectures.

This thesis presents a novel ITS solution that integrates expertise from cybersecurity, AI, and distributed systems. This interdisciplinary approach facilitates the development of a robust and reliable architecture capable of mitigating intrusions across multiple domains.

The main contributions of this thesis are outlined in the following sections, along with the corresponding related publications.

1.2.1. Research Questions

The motivation above leads to the following research questions, which this thesis sets out to answer. They are addressed across the contributions described in the subsequent subsections and consolidated in the hypothesis of Section 1.2.7.

RQ1. How do deterministic and probabilistic BFT protocols differ in their trade-offs and suitability as the replication foundation of an ITS?

RQ2. How can AI, cybersecurity, and distributed systems be combined into an ITS architecture that stays operational and adaptive under compromise?

RQ3. How can Machine Learning (ML)-driven risk assessment over Open Source Intelligence (OSINT) data reduce exposure windows and guide adaptive, diversity-aware reconfiguration?

RQ4. How can Security Operations Center (SOC) and Security orchestration, automation, and response (SOAR) capabilities automate detection and response for an ITS in constrained, real-world deployments?

RQ5. How can organisations share Cyber Threat Intelligence (CTI) without exposing sensitive data, strengthening collective resilience?

1.2.2. Understanding the BFT protocol

Since ITSs rely on BFT protocols, it was necessary to understand the functionalities of these protocols, particularly their consensus mechanisms, models, and behavior under faults. Additionally, it was required to analyze the different variations of BFT protocols and their associated trade-offs.

This analysis led to a comparative study of deterministic and probabilistic BFT protocols, focusing on their distinctions and roles within the broader consensus landscape. This foundational understanding was a first step in designing an improved ITS architecture, as it required identifying the needs of replicated services and the dependencies that BFT protocols impose on underlying software and hardware. For example, some BFT protocols [15, 16] require the presence of trusted components.

This investigation helped address key research questions: *How do BFT protocols function? Within a Byzantine fault model, how do they mitigate intrusions? Can different BFT protocols be integrated effectively, considering their design requirements and constraints?* The findings presented in the journal paper demonstrate that deterministic BFT protocols generally adhere to the foundational principles established by PBFT [11]. In contrast, probabilistic BFT protocols, such as those based on the work of Ben-Or et al. [17], exhibit more significant deviations from this seminal framework.

The contributions of this work collectively resulted in the following publication:

- **"Deterministic or probabilistic? - A survey on Byzantine fault tolerant state machine replication."**, Tadeu Freitas, João Soares, Manuel E. Correia, and Rolando Martins, in Computers & Security 129 (2023): 103200, June 2023, `10.1016/j.cose.2023.103200`.

1.2.3. A multidisciplinary ITS

Cyber threats are rapidly evolving, underscoring the need for resilient and adaptive ITSs. Multidisciplinary approaches that integrate advances in AI, cybersecurity, and distributed systems can strengthen defense mechanisms and address the trade-offs of tool integration. Modularity and practicality are critical for seamless adoption in both new and legacy environments. Recent socio-economic trends have spurred investment in advanced cyber defense technologies [18, 19, 20]. Through the combination of data-driven behavioural analysis and automation (AI), modern cryptographic techniques such as Homomorphic Encryption (HE), Secure Multi-Party Computation (SMPC), and Differential Privacy (DP), and

distributed-systems mechanisms like replication and diversity, organisations can enhance ITS architectures.

Skynet addresses these challenges by integrating methodologies from these domains to create an adaptive ITS capable of mitigating current threats and learning for future defense. Key features include automated threat detection and mitigation, proactive vulnerability management, full process automation, as well as integration with OSINT and Security Information and Event Management (SIEM) for bidirectional feedback and SOAR adoption. Skynet's modular design allows tailored deployments, ranging from complete integration to selective component use, balancing complexity and cost.

Despite these advances, deploying ITS faces notable challenges: combining BFT and intrusion resilience despite differing failure assumptions, managing resource and performance overheads [21], detecting and mitigating persistent threats like APTs [22], and ensuring usability for practical adoption. Effective solutions must unify BFT with security, minimize resource impact, enhance anomaly detection and adaptive response, and provide user-friendly interfaces for seamless integration.

The contributions of this work collectively resulted in the following publication:

- **"Skynet: a cyber-aware intrusion tolerant overseer."**, Tadeu Freitas, João Soares, Manuel E. Correia, and Rolando Martins, Disruptive track in the 2023 53rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks-Supplemental Volume (DSN-S), 2023, `10.1109/dsn-s58398.2023.00034`.

1.2.4. Integration of AI for ITS

Building on ITSs, it is essential to structure knowledge of vulnerabilities, exploits, and attacks derived from both benign sources (e.g., white-hat hacking*) and malicious activities. Such data are disclosed via online OSINT sources and organized using standardized frameworks (Common Vulnerabilities and Exposures (CVE), Common Weakness Enumeration, Common Vulnerability Scoring System (CVSS), Common Platform Enumeration, Adversarial Tactics, Techniques, and Common Knowledge (ATT&CK), Detection, Denial, and Disruption Framework Empowering Network Defense, Exploit Prediction Scoring System (EPSS)). These resources help identify and mitigate risks, but they require data cleansing,

*According to the National Institute of Standards and Technology (NIST), a "white hat" hacker is a cybersecurity specialist who breaks into systems to evaluate and improve an organization's security—retrieved from `https://www.nist.gov/itl/smallbusinesscyber/training/glossary`

such as filtering noise and validating indicators, which is often performed manually. This manual evaluation introduces delays that adversaries can exploit [23].

To address this, HAL 9000 (HAL) was proposed as a novel Risk Manager that integrates automated OSINT retrieval via the Vulnerability Exploit Hunter Killer (VExHK) module into a local database. HAL extends CVSS [24] using Lazarus's methodology [25], incorporating vulnerability age, patch availability and status, exploit existence, and EPSS scoring [26]. The system reduces exposure windows by promptly updating threat scores.

HAL combines the advisory role of risk managers with predictive scoring. For each available Operating System (OS), it retrieves vulnerabilities, exploits, and EPSS entries from the local database. A trained model assigns scores to new CVEs, then clusters related entries by description similarity to assess resilience (inter-OS threats) and security (per-OS) scores, prioritizing resilience minimization before security. Patch availability and application are included in every step of the scoring and risk assessment process.

Experiments against risk managers by Garcia et al. [25] and Heo et al. [27], covering vulnerabilities from 1999–2023, showed that HAL provides more secure and resilient configurations. Monthly simulations validated its ability to adapt over time, and evaluation of CVSS predictors confirmed Khazaei et al. [28] achieved 99% accuracy. Results demonstrate that HAL outperforms alternatives by integrating EPSS, correct patching, and predictive threat scoring, thereby rapidly reducing exposure windows.

The contributions of this work collectively resulted in the following publications:

- **"HAL 9000: a Risk Manager for ITSs."**, Tadeu Freitas, Carlos Novo, João Soares, Inês Dutra, Manuel E. Correia, Behnam Shariati, and Rolando Martins, in the 2024 IEEE 6th International Conference on Trust, Privacy and Security in Intelligent Systems, and Applications, 2024, `10.1109/tps-isa62245.2024.00044`.
- **"A Risk Manager for Intrusion Tolerant Systems: Enhancing HAL 9000 with New Scoring and Data Sources"**, Tadeu Freitas, Carlos Novo, João Soares, Inês Dutra, Manuel E. Correia, Behnam Shariati, and Rolando Martins, in Software: Practice and Experience (2025) special edition, `10.1002/spe.70016`.

1.2.5. The Integration of a SOC for ITS

Building on HAL's adaptive configuration strategies for Skynet, this work enhances resilience by integrating SOC with SOAR capabilities. This work leverages insights from the

Resilient Computing and Cybersecurity Center (RC3) research group at KAUST, where the development of a Vehicle Security Operations Center (VSOC) combined with advanced Electric Vehicle Security orchestration, automation, and response (EVSOAR) designs exemplifies sophisticated SOC/SOAR integration.

Previously, the CyberResil team introduced ScalOTA [29], a secure and scalable over-the-air update framework utilizing wired Electric Vehicles (EV) charging sessions. Building on these foundations, a novel EVSOAR architecture was designed, establishing a dedicated vehicle SOAR framework as its operational hub. This design supports secure updates, rapid threat alert transmission, and automated responses: each EV hosts a local SOAR agent, charging stations deploy Edge-SOAR modules, and the centralized EVSOAR orchestrates detection, response, and update dissemination.

Experimental results showed that Power Line Communication (PLC) outperforms wireless methods for cybersecurity data exchange, while application-level tests highlighted trade-offs between ML-based and Federated Learning (FL)-based Intrusion Detection System (IDS) approaches. These findings directly informed the Skynet architecture, which adopts a distributed paradigm where multiple nodes, each equipped with a local SOAR agent, analyze logs and threats in real time and forward critical data to a centralized SOAR for coordinated incident response.

The contributions of this work collectively resulted in the following publication:

- **"EVSOAR: Security Orchestration, Automation and Response via EV Charging Stations."**, Tadeu Freitas, Erick Silva, Rehana Yasmin, Ali Shoker, Manuel E. Correia, Rolando Martins, and Paulo Esteves-Verissimo, in the IEEE 101st Vehicular Technology Conference, 2025, `10.1109/VTC2025-Spring65109.2025.11174597`.

Additional contributions.

During this collaboration with RC3, the author also contributed to the development of EVOLVE [30], a platform that enhances conventional electric vehicle charging stations by integrating dedicated hardware and software infrastructure to support third-party services.

EVOLVE served as the foundational platform for deploying EVSOAR and enabled comprehensive experimental evaluations. While a detailed discussion of this contribution lies beyond the scope of this thesis, it highlights the advantages of modularity and service segregation. Notably, EVOLVE demonstrates that services available at one charging

station may differ from those at another, reflecting a key principle of the Skynet architecture: distinct Skynet instances provide differentiated services tailored to their deployment context.

The contributions of this work collectively resulted in the following publication:

- **"EVOLVE: a Value-Added Services Platform for Electric Vehicle Charging Stations."**, Erick Silva, Tadeu Freitas, Rehana Yasmin, Ali Shoker, and Paulo Esteves-Verissimo, in the IEEE 101st Vehicular Technology Conference, 2025, 10.1109/VTC2025-Spring65109.2025.11174448.

1.2.6. Federated approach for ITS for information sharing

Following this collaboration, the trade-offs were analyzed between using FL-based IDS with single versus multiple information sources. This analysis led to the concept of a Federated ITS framework, designed to share private and critical data securely both within and across organizations. Central to this framework are techniques that either eliminate the need to share raw data or enable the sharing and processing of sensitive data without exposing it.

Among the privacy-preserving methods considered are HE, Multiparty Homomorphic Encryption (MHE), DP, SMPC, Zero-Knowledge Proof (ZKP), and encrypted anonymized data. Building upon these foundations, Linked Entities for Governance of Intrusion and Operational Norms of Intrusion Tolerant Systems (LegionITS) was introduced, a Federated ITS tailored for the sharing of CTI and Cybersecurity Information Sharing (CIS) among participating organizations and the broader global cybersecurity ecosystem. This approach draws on surveys [31, 32] highlighting the benefits and challenges of information sharing, the types of shareable data, privacy and security concerns, and system improvements made possible through collaborative data exchange in cybersecurity.

1.2.7. Thesis Hypothesis

We summarize the findings of this thesis in the following hypothesis:

A more reliable and advanced ITS architecture can be designed and implemented by leveraging interdisciplinary expertise across AI, cybersecurity, and distributed systems, thereby enhancing defenses against current and future threats. Furthermore, by integrating OSINT data, feedback from Chief Information Security Officers, and robust information sharing, such an architecture will achieve increased resilience through enhanced diversity.

1.3. Thesis Overview

Chapter 2: Background & Related Work

This chapter provides essential background knowledge to understand the foundation of this thesis. It focuses on methods and techniques designed to enhance intrusion resilience within relevant frameworks, covering key topics such as BFT protocols, replica rejuvenation, diversity, FL, the distinction between data plane and control plane, and cybersecurity approaches.

In addition to foundational concepts, the chapter surveys related work on ITSs and other frameworks aimed at intrusion resilience. It examines how these systems integrate with or complement advances in AI/ML, distributed systems, and cybersecurity domains, providing a comprehensive overview of the current state of the art.

Chapter 3: From BFT to ITS

This chapter presents a systematic review of BFT approaches, emphasizing both probabilistic and deterministic consensus mechanisms. Building upon the thesis author's prior work, the review identifies crucial trade-offs involving reliability, scalability, and performance—key considerations that directly shape the design of ITSs. The insights derived from this survey provide a foundation for the architectural decisions and design strategies elaborated in the subsequent chapter.

Chapter 4: Skynet, an ITS Overseer

This chapter presents the Skynet architecture, an ITS designed to ensure resilience against sophisticated cyber threats, that combines the knowledge and techniques from three distinct fields of research, namely, distributed systems, AI, and cryptography. Skynet represents an initial core design developed by the author, providing key mechanisms for proactive

recovery, fault masking, and distributed trust following the teachings of separation between control plane and data plane. This architecture forms the baseline from which later extensions—such as the federated LegionITS framework—are developed and analyzed.

Chapter 5: HAL

Chapter 5 introduces HAL, a risk management module integrated within Skynet's architecture. HAL continuously assesses operational risks, prioritizes threat responses, and coordinates proactive defense actions across intrusion-tolerant components. By embedding adaptive decision-making capabilities, HAL shifts traditional static fault tolerance into dynamic, risk-aware resilience, thereby advancing the core objectives of ITS.

The system combines threat assessment derived from network-available OSINT databases and social media with predictive processes analyzing recently discovered vulnerabilities and emerging threats, enabling rapid adaptation to evolving risks.

Chapter 6: EVSOAR

This chapter presents EVSOAR, a SOAR architecture tailored for electric vehicle cybersecurity. Leveraging the EV Charging Station Network (CSN) as a distributed edge computing platform with PLC, EVSOAR enables real-time threat detection, FL–based intrusion detection, and automated responses close to the vehicle. The architecture comprises an in-vehicle SOAR agent, Edge SOAR components at charging points, and a centralized SOAR system coordinating analysis and threat intelligence sharing across Original Equipment Manufacturers (OEMs). Experimental evaluation demonstrates EVSOAR's superior latency, scalability, and detection performance compared to existing VSOC solutions. This work exemplifies the application of orchestration and federated security principles within cyber-physical systems, offering valuable insights for ITS design.

Chapter 7: LegionITS

This chapter presents LegionITS, a federated intrusion-tolerant framework that extends the principles of Skynet to multi-organizational environments. LegionITS enables autonomous ITS instances to collaborate through privacy-preserving mechanisms, federated learning, and decentralized trust management. This chapter details the system architecture, coordination protocols, and threat intelligence-sharing strategies that collectively enable large-scale, intrusion-tolerant federations.

Chapter 8: Conclusion

Chapter 8 concludes the thesis by summarizing the key contributions of Skynet, HAL, and LegionITS in advancing resilient, adaptive, and federated ITSs. It reflects on the implications of these architectures for securing critical infrastructure, while also discussing the limitations encountered during their design and evaluation. Finally, the chapter outlines future research directions, including enhanced integration of threat intelligence, development of advanced risk modeling techniques, and the progression toward fully autonomous federated defense systems that operate with minimal human intervention.

2. Background & Related Work

This chapter provides the foundational background required for a comprehensive understanding of the topics presented throughout the thesis. It introduces the methods and techniques from the different research fields.

Subsequently, the chapter provides a revision of the related work on ITSs, analyzing the key contributions, trade-offs, and limitations identified in each contribution. These gaps highlight areas where the proposed *Skynet* architecture can contribute to meaningful innovation.

2.1. Background

ITS: Definition and Motivation

ITS [9] is a fault-tolerant architecture designed to maintain operational functionality and security even when intrusions occur. These systems prevent intrusions from causing total system failure or exposing sensitive data. This resilience is achieved through mechanisms that enforce confidentiality, integrity, and availability. In practice, ITSs implements strategies such as redundancy, diversity, active self-healing, and dynamic reconfiguration. Consequently, even if an attacker compromises part of the system, the overall system remains secure and operational, effectively containing and mitigating the intrusion.

This paradigm is necessary in domains where uninterrupted operation is paramount, such as critical infrastructure, cyber-physical systems, and defense.

Threat and Failure Models

ITSs are designed under assumptions about the types of faults and adversarial behaviors that may affect the system during operation. At the core of these assumptions is the understanding that systems do not fail solely due to accidental or random faults but may also be subject to intentional, coordinated, and sophisticated attacks by malicious actors. Therefore, both traditional failure models and advanced threat models must be considered in tandem.

From the perspective of traditional fault tolerance, several types of faults are typically accounted for. **Crash faults** represent one of the simplest failure modes, where a process or component ceases to function and stops producing output. Although such faults are relatively easy to detect and recover from, they are only a subset of possible disruptions. The

most general and challenging class of faults is **Byzantine faults**, where a component can behave arbitrarily, possibly producing incorrect, inconsistent, or malicious outputs. Byzantine behavior may result from software bugs, hardware errors, or deliberate compromise by an adversary and poses a significant challenge to achieving consensus and maintaining system integrity.

In addition to these fault types, ITSs must also account for a broad range of intrusion scenarios. **Active intrusions** involve direct interference with system behavior, such as code injection, privilege escalation, or the manipulation of data and control flows. **Passive intrusions**, by contrast, may involve eavesdropping or surveillance, allowing adversaries to gather intelligence without directly altering the system state. Some of the most dangerous threats are coordinated attacks, including APTs, where an adversary maintains long-term unauthorized access to a system to exfiltrate data, disrupt operations, or weaken trust mechanisms over time. Malicious adversaries often exploit software vulnerabilities and system weaknesses such as misconfigurations or weak access controls. When vulnerabilities are discovered by security professionals, they follow a disclosure and patching process, often being published in repositories such as NIST's NVD or ExploitDB. Conversely, if attackers discover them first, they may leverage them in zero-day exploits before mitigations are available.

High-profile incidents such as Stuxnet [12], Operation Aurora [13], the Equifax data breach [14], and the Vodafone Portugal attack [33] illustrate how zero-day exploits and coordinated campaigns can compromise critical services, cause significant disruption, and highlight the necessity of resilient architectures like ITS.

The sophistication of these threats necessitates the use of attacker models that extend beyond simple assumptions. An effective ITS design must consider adaptive attackers, i.e., capable of changing tactics in response to system defenses, and who may possess insider access or knowledge of internal system architecture. In addition, contemporary attackers frequently employ stealth as a core strategy, utilizing techniques that evade detection by IDSs or mimic legitimate behavior.

Recognizing and addressing these characteristics is essential for ensuring the effectiveness and resilience of ITSs in adversarial environments. In parallel, complementary initiatives such as bug bounty programs [34, 35, 36] incentivize ethical hacking, enabling early vulnerability discovery and reducing the risk of adversaries weaponizing undisclosed exploits. As such, designing effective ITS architectures requires a holistic view of resilience.

The system must simultaneously address fault tolerance, intrusion resilience, and operational continuity under uncertainty.

Foundations of Fault and Intrusion Tolerance

The design of ITSs builds upon a foundation of techniques developed within the field of fault-tolerant computing, extending them to address adversarial threats and malicious faults. At the core is the replication of critical system components to avoid scenarios of single point of failure. The use of redundancy supports mechanisms for failover, consensus, and masking, allowing the system to continue functioning correctly even with reduced performance. While BFT components may behave arbitrarily, ITS extend this model by actively detecting, counteracting, and recovering from malicious intrusions, rather than merely masking their effects.

In addition, diversity plays a critical role in enhancing system resilience to avoid scenarios of "break one, break all". By deploying heterogeneous software and hardware implementations of the same function, ITSs can reduce the risk of common-mode failures triggered by a single exploit or vulnerability affecting all instances. Diversity increases the difficulty for attackers to successfully compromise multiple components simultaneously, particularly when those components vary in design, execution environments, or defensive configurations.

Another foundational concept leveraged in ITS is State Machine Replica (SMR), which guarantees consistency across distributed components. SMR allows multiple replicas of a service to process the same sequence of inputs and maintain equivalent internal states. To achieve this, ITSs employ quorum-based consensus protocols that ensure all non-faulty replicas agree on the system's current state and progression, even when facing Byzantine behavior or partial network failures. Such protocols are critical for maintaining data integrity and system coherence in distributed environments where some components may be compromised or under attack.

What distinguishes ITS from traditional fault-tolerant systems is the assumption that some components may behave maliciously rather than fail silently. As such, ITSs are designed not only to recover from unintentional faults but also to detect, isolate, or neutralize components that exhibit adversarial behavior. This shift in focus from passive fault tolerance to active adversary mitigation necessitates architectural designs in which intrusion detection, recovery mechanisms, and containment strategies are treated as integral. ITSs

do not depend solely on the assumption that components will fail by halting; instead, they anticipate that compromised elements may attempt to subvert the system's logic, requiring robust mechanisms to mask or counteract such actions while preserving availability and correctness.

System Assumptions and Models

The assumptions on the underlying system model influence the design of ITSs. This defines the operational context in which tolerance mechanisms are expected to function. One of the most critical dimensions is the timing model. A system may be **synchronous**, where known bounds exist on message delays and process execution times; **asynchronous**, where no such bounds are guaranteed; or **partially synchronous**, where timing guarantees may eventually hold after an unknown but finite period. These assumptions directly affect the choice and correctness of consensus protocols, recovery mechanisms, and coordination strategies, particularly under fault or intrusion scenarios.

Equally important are the guarantees provided by the communication model. ITS mechanisms often rely on the availability of reliable message delivery, ordered transmission, and authenticated communication channels. Without these properties, fundamental tasks such as replica coordination, state agreement, or intrusion signaling may fail or become vulnerable to adversarial manipulation. For instance, an attacker who can tamper with message integrity or spoof identities may undermine the consistency or trust assumptions of the system.

In addition to timing and communication, ITSs can also consider the application of trusted components that serve as anchors for security and recovery. These may include secure enclaves, hardware security modules, Trusted Platform Module, or trusted time sources. These are essential for attestation, secure logging, and rollback-safe recovery. However, these elements' availability, trustworthiness, and correct configuration are not guaranteed by default and must be explicitly incorporated into the system model.

Clearly articulating these assumptions is essential for protocol correctness and ensuring that the strategies remain valid when deployed in real-world conditions. Failure to accurately model timing, communication, or trust boundaries can result in brittle designs that break under adversarial pressure or operational variability. As such, system models form the foundation upon which ITS architectures must be critically evaluated, especially in

heterogeneous and federated environments where conditions may differ significantly across components or administrative domains.

Resilience and Survivability

Beyond fault and intrusion tolerance, ITSs are expected to contribute to the broader goals of system **resilience** and **survivability**. Resilience refers to the system's ability to maintain an acceptable level of service in the face of unexpected disruptions, whether caused by faults, failures, or cyber-attacks. This encompasses the ability to mask the system's degradation and adapt dynamically to adverse conditions while maintaining its operational requirements.

Survivability, in turn, emphasizes the system's ability to restore critical functionality after a compromise has occurred. While resilience addresses the system's behavior during ongoing disturbances, survivability focuses on recovery. This ensures that the system can recover essential operations within acceptable time and quality bounds even if core components are rendered inoperative or corrupted. In high-assurance or safety-critical domains, this is a crucial property that determines whether the system can fulfill its mission after a successful attack or cascading fault.

To assess the effectiveness of ITSs in fulfilling these roles, systems are often evaluated using resilience metrics such as the Mean Time to Recovery, which quantifies how quickly the system can return to an operational state after a failure or intrusion. Other important metrics include the degree of service continuity maintained during an attack and the system's capacity for graceful degradation, wherein non-essential functionality may be temporarily suspended to preserve core operations. These metrics provide a quantitative basis for comparing different ITS architectures and aligning their performance with domain-specific risk tolerance and regulatory requirements.

Integration of resilience and survivability into the design of ITS ensures that such systems are robust against isolated faults and targeted attacks and capable of withstanding and recovering from complex, multi-stage disruptions in a predictable and controlled manner.

ITS in Critical Infrastructure and Cyber-Physical Systems

ITS play a crucial role in the protection of Cyber-Physical Systems and critical infrastructure, where the consequences of cyber attacks can extend beyond digital boundaries and

result in physical harm, financial losses, and risks to human safety. In these environments, the impact of a successful intrusion can be catastrophic. Considering the practical example where an attack on the power grid might disrupt the electricity supply, damage equipment, or even endanger lives by impairing vital services like hospitals or emergency communications, it becomes clear that robust security measures are essential. Ensuring real-time guarantees is essential, as many critical applications rely on predictable system behavior with minimal latency to maintain safety and operational continuity.

These systems often operate under stringent regulatory and safety requirements, complicating the deployment of traditional security solutions. Many rely on legacy components with limited processing capacity and outdated security models, making them difficult to patch or upgrade without interrupting service. Furthermore, the long lifespans and certification cycles of critical systems demand resilient security architectures that can continue to function reliably even in the presence of ongoing attacks.

ITS address these with the application of redundancy, diversity, and proactive recovery mechanisms that allow systems to withstand and recover from intrusions without compromising their core functionalities. Their application spans various domains, including power generation and distribution networks, smart transportation systems, industrial automation and control systems, healthcare delivery infrastructures, and coordinated emergency response systems. In each case, the integration of ITS enhances resilience, ensuring that critical services remain operational even when parts of the system have been compromised.

Multidisciplinary Challenges

ITSs serve as a bridge between multiple research fields, each contributing essential principles and techniques that shape their design, implementation, and evaluation. From cybersecurity, ITS draw upon methods for detecting, responding to, and recovering from active threats, often under adversarial conditions where assumptions about system trustworthiness are routinely violated. At the same time, the foundations of distributed systems play a vital role, as ITSs must maintain availability, consistency, and fault tolerance even when parts of the system are compromised or unresponsive. These properties are particularly complex to ensure in the presence of byzantine faults, where components may behave arbitrarily or maliciously.

Systems engineering introduces further complexity, as ITSs must be dependable by design while also satisfying safety, reliability, and resilience requirements over long operational periods. This often involves rigorous architectural modeling, formal verification, and redundancy strategies spanning hardware and software layers. Additionally, human factors and governance considerations are increasingly recognized as central to ITS success. Additionally, systems must be explainable, auditable, and manageable by human operators, especially in high-stakes environments where trust, accountability, and regulatory compliance are non-negotiable.

Balancing these diverse demands leads to intricate trade-offs. For example, introducing stronger security mechanisms can increase system complexity and reduce performance, while overly simplifying interfaces to improve usability might obscure critical decision-making contexts. Regulatory and ethical constraints further influence design choices, requiring systems to justify their actions, support oversight, and operate transparently in environments subject to public scrutiny.

Despite these challenges, the evolving landscape of AI and distributed computing offers promising prospects to enhance ITS. Advances in AI-driven cybersecurity enables faster, more adaptive threat detection and response through anomaly detection, behavior modeling, and reinforcement learning. These tools can complement traditional defenses by identifying subtle or emerging attack patterns that evade static rule-based systems. In parallel, new developments in distributed consensus, SMPC, and FL provide more robust and privacy-preserving coordination mechanisms for systems that must operate across organizational or geographic boundaries. Integrating these innovations into ITSs can lead to more intelligent, scalable, and resilient architectures capable of tolerating faults and intrusions and evolving in response to a continuously shifting threat landscape.

2.2. Related Work

ITSs have been researched through theoretical frameworks and practical implementations. Initial foundational research established fault-tolerant architectures, but due to their complexity and high performance costs, they saw limited adoption. The concept of intrusion tolerance was first articulated by Fraga and Deswarte, emphasizing that systems must not only prevent but also tolerate intrusions [37]. Deswarte et al. [8] presented early practical mechanisms; however, their overheads hindered progress for over a decade. A pivotal advancement occurred with Castro and Liskov's practical BFT protocol [11], which spurred projects like the European Union's MAFTIA [38, 39, 40] and DARPA's OASIS [41]. These

initiatives proposed ITS architectures by embedding redundancy, diversity, and replication, enabling robust service continuity amid adversarial conditions. Improvements in computing capabilities and the evolving threat landscape have since facilitated the transition from prototypes to deployable solutions.

MAFTIA introduced a modular, multi-layered design incorporating a Trusted Timely Computing Base (TTCB) [42] distinguished from traditional Trusted Computing Base (TCB) by its minimalism and isolation [43]. This TTCB provided trusted services like accurate global time and BFT consensus primitives, allowing the system to rely on replication, diversity, and voting to tolerate intrusions. MAFTIA's middleware further layered secure group communication, replicated service execution, and intrusion-tolerant transactions, shifting focus from intrusion prevention to system survivability. While highly resilient, this approach introduced scalability and integration complexities alongside performance overhead.

Building on this, ITUA [44], an OASIS component [45], innovated by embedding unpredictability into adaptive responses—randomizing replica activations, task reallocations, and component isolations, to thwart adversarial anticipation. This strategy combined redundancy, adaptive reconfiguration, diversity, and BFT protocols within compartmentalized security domains. Its adaptive middleware facilitated graceful service degradation under attack, though it incurred overhead in redundancy, communication, and system observability.

TRONE [46] targeted large-scale cloud and telecom infrastructures, balancing robustness and performance by employing BFT protocols on critical communication channels and Crash Fault Tolerant protocols for less critical or non-adversarial channels. Its publish-subscribe event broker supported scalable and secure dissemination, enhanced by automated diagnosis through peer-similarity analysis that detects faults and attacks. Features like multi-homing enabled rapid network reconfiguration. TRONE's design balanced increased cryptographic and protocol complexity against improved resilience.

CRUTIAL [47] addressed critical infrastructures such as power grids by proposing a reference architecture with replicated CRUTIAL Information Switches (CIS), which act as secure gateways between local area networks. These CIS enforce secure information flows, access policies, BFT consensus, and intrusion detection, augmented with proactive replication rejuvenation and trustworthiness monitoring. This layered, macro-level defense

approach introduced additional resource consumption and latency, requiring careful trust and integration management, particularly given legacy system heterogeneity.

Middleware-focused advances include ITDOS [48], which integrated BFT consensus into the CORBA Object Request Broker [49], employing replicated asynchronous state machines and BFT multicast to mask faults, though at the cost of requiring $3f+1$ replicas and fault detection. Worm-IT [15] reduced replica numbers to $2f+1$ by leveraging the TTCB for trusted agreement, authentication, and timestamping services, while admission control mitigated Denial-of-Service attacks, with effectiveness dependent on the TTCB's security and dynamic membership management.

VM-FIT [50] exploited virtualization to isolate service replicas within separate virtual machines managed by secure Virtual Machine (VM) monitors, preventing lateral attacker movement. It deliberately diversified OS and software stacks among replicas and incorporated proactive rejuvenation to limit attacker persistence. These measures enhanced resilience but increased operational complexity, resource use, and dependence on hypervisor integrity.

PRRW [51] proposed a hybrid recovery model combining periodic proactive rejuvenation with reactive recovery triggered by anomaly detection. Scheduled recoveries maintained sufficient correct replicas to uphold BFT guarantees, improving survivability against ongoing threats while balancing overhead and recovery accuracy.

Efforts to ensure perpetual and adaptive resilience extended into FOREVER [52], which coordinated periodic and event-triggered recoveries with diversity to continuously refresh the software stacks and configurations of replicas. By isolating this service within a trusted subsystem and enforcing synchronized coordination, FOREVER overcame limitations of static BFT in the face of evolving threats. Its complexity and coordination overheads represent key trade-offs.

The Self-Cleansing Intrusion Tolerance (SCIT) framework [53] similarly limited adversary residence time by alternating cleansing cycles between active and standby components, proactively rebooting systems regardless of detection status. This method is particularly suited to stateless, short-session services, reducing persistent compromises but requiring hardware or virtual redundancy and introducing transient service disruption.

COCA [54] secured online certification authorities via replication, Byzantine quorum systems, threshold cryptography, and proactive recovery. It addressed mobile adversaries by periodically refreshing cryptographic shares and recovering server state, supporting

availability under asynchronous networks and mitigating Denial-of-Service via resource partitioning and caching. These protections introduced significant cryptographic and communication overhead and depended on sustaining a quorum of uncompromised servers and secure recovery protocols.

SPIRE [55] enhanced the security of power grid Supervisory Control and Data Acquisition systems by deploying geographically distributed replicas running the Prime BFT consensus protocol [56] alongside proactive recovery and automated software diversification. Its Spines overlay network ensured resilient communication under targeted attacks. Though multi-site replication and proactive rejuvenation posed latency and complexity challenges, SPIRE demonstrated practical viability, meeting stringent availability and latency requirements crucial for critical infrastructure.

Similarly, ITCIS-PRR [57] employed replicated, diverse access control components with voting mechanisms and combined proactive and reactive recovery. Housed within a secure subsystem, it assured trustworthiness despite ongoing attacks. While increasing overhead and temporarily reducing capacity during recovery, it eliminated single points of failure common in traditional firewalls.

The SITAR architecture [58] built on COTS servers by integrating layered defenses: proxy servers managing requests, acceptance monitors validating inputs and outputs, ballot monitors adjudicating responses, and an audit control module performing active probing for stealthy intrusions. Its adaptive reconfiguration module dynamically adjusted system configuration in response to threat changes, balancing redundancy and validation with operational costs and complexity.

ITSI [59] tackled high availability in server clusters by equipping nodes with smart network interface cards, enforcing security policies independently of the host OS. Combined with heterogeneous platforms and centralized intrusion detection feeding the Availability and Integrity Controller (AIC), it contained lateral attacks and managed dynamic failover and forensic analysis, trading off increased hardware and management complexity for improved resilience.

More recent advances include Lazarus [25], which automated proactive replica diversity management in BFT systems by harvesting real-time vulnerability intelligence from databases such as the NVD and ExploitDB. Its control plane assesses risk and orchestrates replica replacement, patching, and quarantine via secure Local Trusted Units. This

automation enhances fault independence but demands increased resource overhead, incurs transient reconfiguration costs, and relies on timely, accurate intelligence and trusted infrastructure.

TOLERANCE [60] maximizes availability and minimizes costs by dynamically managing recovery and replication through a two-level feedback control: local controllers analyze IDS alerts to estimate compromise and trigger recovery, while a global controller adjusts replication factors. Using a reconfigurable BFT consensus, it adapts to fluctuating threats and detection inaccuracies, trading monitoring and coordination overhead against operational efficiency and responsiveness.

Table 2.1 synthesizes these foundational ITS contributions across critical dimensions: trusted components (TC), Byzantine fault tolerance (BFT), proactive/reactive recovery (PRR), automated adaptation (AA), dynamic reconfiguration (DR), automated diagnosis (AD), software diversity (SD), and resource management (RM). Collectively, these works chart a maturation from static fault tolerance toward architectures emphasizing continuous recovery, adaptive diversity, and intelligent automation—essential developments to sustain service correctness and availability amid persistent, evolving adversaries. While incurring inevitable trade-offs in complexity, performance, and operational overhead, these advances collectively push the state of the art in intrusion tolerance and secure dependable computing.

Table 2.1: Summary of ITS Contributions by Topic.

A ✓indicates a contribution by the work in the respective topic area.
(Legend: TC - Trusted Component; BFT - Byzantine Fault Tolerance; PRR - Proactive/Reactive Recovery; AA - Automated Adaption; DR - Dynamic Reconfiguration; AD - Automated Diagnosis; SD - Software Diversity; RM - Resource Management.)

Research	**TC**	**BFT**	**PRR**	**AA**	**DR**	**AD**	**SD**	**RM**
MAFTIA [38]	✓	✓	✓		✓			
ITUA [44]		✓		✓	✓	✓	✓	
TRONE [46]		✓	✓	✓	✓	✓		✓
CRUTIAL [47]	✓	✓	✓		✓	✓		
ITDOS [48]		✓					✓	
Worm-IT [15]	✓	✓	✓		✓			✓
VM-FIT [50]	✓	✓	✓			✓	✓	
PRRW [51]		✓	✓					
FOREVER [52]	✓	✓	✓				✓	
SCIT [53]			✓					
COCA [54]	✓	✓	✓					✓
SPIRE [55]			✓		✓		✓	
ITCIS-PRR [57]	✓	✓	✓				✓	
SITAR [58]		✓	✓	✓	✓	✓		
ITSI [59]	✓		✓	✓	✓	✓	✓	✓
Lazarus [25]	✓	✓	✓	✓	✓		✓	✓
TOLERANCE [60]	✓	✓	✓	✓	✓			✓

2.3. Chapter Summary

The present chapter has established the foundational context for understanding ITS, tracing their evolution from early fault-tolerant architectures to advanced solutions capable of withstanding both accidental faults and sophisticated adversarial attacks. It introduced the core concepts, including system and threat models, the importance of resilience and survivability, and the multidisciplinary challenges inherent to designing robust ITS for critical infrastructure and cyber-physical systems.

The review of related work highlighted the progression of the field, from seminal theoretical contributions to practical frameworks and architectures such as MAFTIA [38], ITUA [44], TRONE [46], CRUTIAL [47], and others. Each approach was analyzed in terms of its key mechanisms, such as replication, diversity, BFT, proactive and reactive recovery, and adaptive reconfiguration, and the trade-offs between security, performance, and operational complexity.

By consolidating these research contributions, this chapter has identified persistent challenges and open research questions, including the need for scalable, adaptive, and resource-efficient solutions that can operate in heterogeneous and evolving environments, as well as the importance of integrating findings from diverse yet complementary research fields. This overview sets the basis for the subsequent chapters, which will build upon these insights to present new methodologies and contributions aimed at advancing the state of the art in ITS design.

3. From BFT to ITS: a survey on BFT consensus.

This chapter surveys the principles and protocols of BFT, forming the cornerstone upon which the contributions of this thesis are built. BFT serves as a foundational approach for ITS, particularly through SMR, as it is designed to mask and recover from arbitrary faults, including those caused by malicious adversaries.

BFT protocols provide strong guarantees as long as the number of faulty or compromised replicas does not exceed a defined threshold. However, they focus primarily on handling faults during protocol execution and do not proactively address attacks that unfold over time, such as information leakage or the gradual compromise of replicas, which can eventually breach the threshold of a maximum of f faulty replicas.

To address such limitations, recent implementations of BFT protocols have begun to integrate supplementary mechanisms, most notably, replica rejuvenation, to mitigate the risk of stealthy, long-term attacks and progressive compromise. Consequently, modern ITSs adopt BFT as a core mechanism and augment it with a range of additional techniques and tools, including proactive recovery, diversity, dynamic reconfiguration, and IDS, in order to provide comprehensive intrusion tolerance.

The choice of BFT protocol dictates the requirements that the ITS must fulfill for correct deployment and operation, as well as the threat model it must assume. In this chapter, BFT protocols are systematically compared along several key dimensions: the distinction between deterministic and probabilistic approaches to agreement, the underlying cryptographic primitives and assumptions, the network model, and strategies for handling faulty leaders.

3.1. Background and Key Concepts

BFT-SMR protocols are a foundational component in critical distributed systems, designed to guarantee correct behavior even when some replicas experience arbitrary faults, including those caused by malicious adversaries. The core idea behind SMR is to ensure that multiple replicas of a service execute the same set of operations in the same order, providing strong consistency and fault tolerance even in adversarial settings. Well-known industrial systems, such as Google Spanner [61] and Apache ZooKeeper [62], benefit from such replication schemes.

3.1.1. Fault and Network Models

BFT-SMR protocols operate under various fault models, where up to f out of N replicas can be faulty (when $N \geq 3f + 1$ or with a trusted component integrated $N \geq 2f + 1$), and make minimal assumptions about the cause, be it software errors or deliberate compromise. Communication occurs via unreliable networks where it is assumed that messages can be dropped, delayed, duplicated, or corrupted. To maintain both safety (all correct replicas reach the same state) and liveness (all valid client requests are eventually processed), BFT-SMR protocols employ cryptographic primitives, most notably digital signatures and Message Authentication Code (MAC), under the assumption that adversaries are computationally bounded and cannot break these mechanisms.

The network assumptions underlying BFT-SMR protocols play a crucial role in determining their design and correctness guarantees:

- Synchronous networks: Every message sent between replicas is delivered within a known, fixed time bound. This strong assumption allows protocols to provide predictable liveness, but is rarely achievable in practical, large-scale distributed environments.

- Asynchronous networks: No guarantees exist regarding message delivery times; messages may be delayed arbitrarily or even lost. While this model accurately captures highly unreliable networks, it poses significant challenges for guaranteeing progress in consensus protocols.

- Partially synchronous networks: The system typically operates without delivery time bounds (asynchronously), but there are periods when messages are reliably delivered within bounded delay (synchronous intervals). This model is widely adopted by practical BFT-SMR protocols, as it reflects real-world conditions, capturing both temporary network partitions and intervals of reliable connectivity, while still enabling practical liveness guarantees.

3.1.2. Consensus Impossibility and Protocol Approaches

The Fischer-Lynch-Paterson (FLP) impossibility result [63] demonstrates that deterministic consensus cannot be guaranteed in a fully asynchronous distributed system if even a single process may fail. This foundational result implies that, in such environments, it

is impossible for deterministic protocols to always achieve agreement in the presence of faults.

To address this limitation, two main classes of consensus protocols have emerged: deterministic and probabilistic (or randomized) approaches.

Deterministic consensus protocols relax the purely asynchronous assumption by assuming that the system is eventually synchronous, that is, after some unknown but finite time, the system behaves synchronously and messages are delivered reliably within a bounded time. These protocols guarantee both agreement and liveness during periods of synchrony.

Probabilistic (randomized) consensus protocols circumvent the FLP impossibility by introducing randomization, such as cryptographic coin-tossing or leader election via verifiable random functions. These protocols can achieve consensus with high probability even under true asynchrony, but do so at the cost of potentially unbounded decision time and increased algorithmic complexity.

3.2. Preliminaries

This section introduces the key concepts, terminology, and classification principles used throughout the chapter to ensure both precision and clarity. Readers are referred to the thesis terminology page for definitions of consensus, SMR, Atomic Broadcast, Byzantine faults, trusted components, cryptographic primitives, and complexity metrics. For consistency, the following terms are adopted: **Replica/Node**—a process maintaining the state machine; **Leader/Primary**—the coordinator replica in leader-based protocols; **Backup**—a non-leader replica participating in consensus; **Byzantine Node**—a faulty or malicious replica; Client—an entity submitting service requests.

The survey structures into two main classes of BFT consensus protocols: deterministic and probabilistic (randomized), both developed as responses to the FLP impossibility [63]. Deterministic protocols guarantee consensus and liveness under synchrony, with discussion centered on their key innovations in cryptography, trust models, fault tolerance, performance, or leader recovery. Probabilistic protocols, by contrast, leverage randomness (e.g., threshold coin-tossing or verifiable random functions) to achieve consensus with high probability under full asynchrony, provided Byzantine faults remain within tolerated bounds. This structure enables clear comparison of innovations, trade-offs, and deployment considerations across both protocol classes. For full technical details, readers may consult Freitas et al. [64].

3.3. Deterministic consensus

Research on practical deterministic consensus protocols began with Castro and Liskov's seminal PBFT [11], which remains the model for subsequent work. Most follow-up protocols adopt PBFT's leader-based approach and structured message exchanges, with innovation often confined to refinements such as more efficient communication, improved leader election, checkpointing, state transfer, or replica recovery. While these optimizations enhanced robustness and performance, fundamental advances arose from protocols that redefined consensus design itself.

PBFT introduced the first practical BFT SMR system, ensuring safety and liveness so long as fewer than one-third of replicas are faulty and eventual synchrony holds. Its key contributions include a five-phase protocol, from client request through leader ordering, replica prepares, commits, and final replies, along with a view-change mechanism based on certificates for prepared and committed requests, enabling safe leader replacement without compromising progress.

Zyzzyva [65] extended this baseline with speculative execution. Replicas speculatively process requests immediately after a leader round-trip, allowing clients to confirm execution upon $3f + 1$ matching responses. This optimistic fast path achieves consensus in two steps under stable, fault-free conditions, reverting to a classical path when inconsistencies arise. Its evidence-based view-change further shortens recovery times by relying on client-provided proof of leader misbehavior.

Hardware-assisted protocols introduced the next breakthroughs. MinBFT and MinZyzzyva [66] employed the tamper-resistant USIG module, which generates unique, monotonic, and non-forgeable identifiers. This eliminates equivocation even under Byzantine faults and reduces the replica requirement from $3f+1$ to $2f+1$. MinBFT preserves PBFT's commit structure, while MinZyzzyva integrates Zyzzyva's speculative execution, both simplifying protocol communication and improving efficiency. CheapBFT [67] built on this by dynamically switching between crash fault tolerance and Byzantine tolerance through the CASH hardware module, which provides secure counters and authenticated certificates. In benign conditions, CheapBFT operates with only $f + 1$ active replicas; upon misbehavior, it executes a coordinated switch to full BFT, thereby combining efficiency with resilience.

BFT-Mencius [68] addressed scalability through a multi-leader architecture, where each replica manages consensus for a partition of operations. Its Abortable Timely Announced Broadcast (ATAB) mechanism enables safe, concurrent atomic broadcasts, balancing load

and eliminating single-leader bottlenecks. This design improves throughput, particularly under high load or across wide-area networks, while maintaining liveness through adaptive view changes.

Hybster [16] advanced parallelism using Intel SGX enclaves (TrInX) embedded at each replica. These secure counters prevent equivocation and allow consensus with only $2f + 1$ replicas. Replicas are organized into independent parallel "pillars", each running consensus for a log partition, enabling high throughput with minimal cross-pillar synchronization. Hardware-enforced attestation ensures safe recovery during view changes while balancing efficiency, cost, and resilience.

Finally, Mod-SMaRt [69] introduced a modular architecture that decouples consensus from replication logic, treating the consensus engine as a swappable component. Its VP-Consensus protocol produces cryptographically verifiable outcomes, allowing external validation and policy enforcement. By batching requests and abstracting view-change logic, Mod-SMaRt provides high throughput and seamless adaptability to advances in consensus mechanisms.

These protocols demonstrate how deterministic BFT consensus has evolved from PBFT's foundational model toward efficiency, scalability, and resilience through speculative execution, trusted hardware, multi-leader parallelism, enclave-based designs, and modular architectures. Figure 3.1 and Table 3.1 summarize their main contributions, while Table 3.2 compares the trusted components employed.

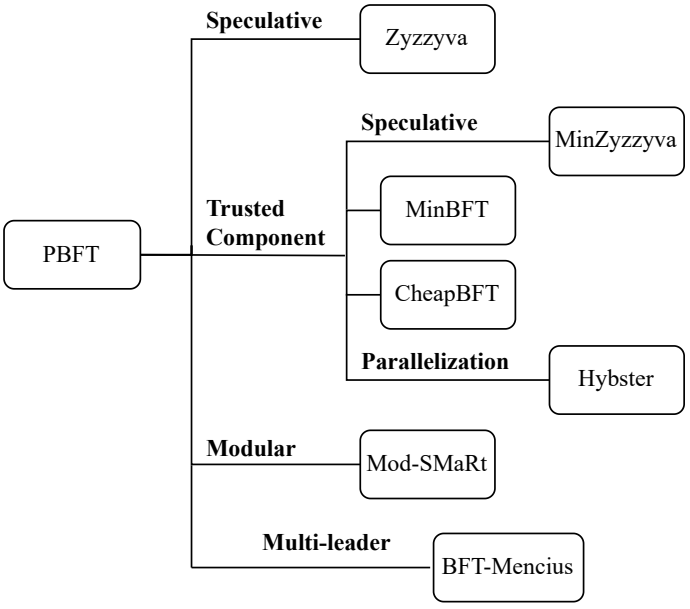

Figure 3.1: Characteristics of deterministic BFT-SMR protocols. Adapted from [64].

TABLE 3.1: Summary of key innovations from the deterministic BFT protocols.

Protocol	**Key Contributions**
PBFT [11]	First practical leader-based BFT; efficient view-change; quadratic message complexity.
Zyzzyva [65]	Speculative execution; fast optimistic path; client-driven commits; evidence-based, efficient view-change.
MinBFT / MinZyzzyva [66]	Trusted hardware (USIG) eliminates equivocation; reduces replicas to $2f+1$. MinBFT: commit-based; MinZyzzyva: speculative execution.
CheapBFT [67]	Adaptive switching between efficient Crash Fault Tolerant and BFT modes using CASH hardware; secure dynamic protocol transitions.
BFT-Mencius [68]	Multi-leader parallel consensus; ATAB primitive for concurrent, abortable atomic broadcast; improved scalability and resilience.
Hybster [16]	Parallel "pillars" with trusted SGX enclaves (TrInX); reduces replicas to $2f+1$; partitioned, efficient, attested consensus; fast, safe recovery.
Mod-SMaRt [69]	Modular, swappable consensus "black box"; cryptographically provable consensus (VP-Consensus); batch processing; generic failover.

TABLE 3.2: Comparison of Trusted Components in deterministic BFT (TTCB, MinZZ - MinZyzzyva, FPGA - Field-programmable gate array). Adapted from [64].

	Trusted Component	Trusted Subsystem	Implementation	Dedicated Hardware
MinBFT	TTCB	USIG (Counter)	Hardware	No
MinZZ	TTCB	USIG (Counter)	Hardware	No
CheapBFT	CASH	Trusted Counter	Hardware	FPGA
Hybster	TrInX	Trusted Counter	Software	Intel-SGX

3.4. Probabilistic consensus

While deterministic BFT protocols like PBFT have established robust foundations for SMR under partial synchrony, their assumptions limit applicability in fully asynchronous or highly unpredictable network environments. To overcome the FLP impossibility, probabilistic BFT protocols leverage randomness and advanced cryptographic primitives, such as threshold signatures, threshold encryption, and jointly generated coins, to achieve consensus with high probability rather than deterministically. This shift enables protocols that function under full asynchrony, leaderless designs, and flexible round structures, fundamentally rethinking consensus mechanisms.

SINTRA [70] introduced an early comprehensive framework employing a modular, leaderless architecture that integrates threshold signatures and threshold coin-tossing for secure, unbiased randomness crucial for asynchronous agreement. By stacking reliable and consistent broadcast layers with strong cryptographic authentication, SINTRA provides composable primitives that offer both theoretical rigor and practical extensibility in wide-area and adversarial networks. Its design eliminates leader bottlenecks and enhances fault tolerance by equalizing replica roles.

Seeking efficient alternatives, Correia et al. [71] proposed a protocol that forgoes costly public-key operations entirely. By using symmetric cryptography and pairwise MACs, every message is authenticated with low computational overhead. Decisions require collecting valid MACs from a quorum, ensuring security without threshold signatures or distributed randomness generation, thus delivering significant latency and throughput improvements ideal for high-performance distributed deployments.

HoneyBadgerBFT [72] marked a substantial step toward practical, fully asynchronous consensus by combining several cryptographic innovations. It utilizes threshold encryption to keep transaction batches confidential until consensus is reached, preventing censorship or premature disclosure. Its Asynchronous Common Subset (ACS) protocol coordinates reliable broadcast of proposals with parallel asynchronous binary agreements, guaranteeing that all correct nodes agree on an identical subset of values despite asynchrony or Byzantine faults. HoneyBadger also employs erasure coding to reduce communication overhead, allowing efficient scaling without sacrificing security or throughput. The protocol's leaderless, censorship-resistant design directly supports blockchain applications and other adversarial settings.

Building upon these ideas, BEAT [73] enhances flexibility through a modular architecture that allows independent tuning of encryption, broadcast, and consensus layers tailored to diverse network conditions. BEAT introduces labelled threshold encryption to uniquely tag ciphertexts, mitigating replay attacks and easing verification. It employs a threshold pseudorandom function for generating unpredictable, verifiable randomness with lower computational cost than classic threshold signatures, enhancing responsiveness during random coin generation. Additionally, BEAT leverages advanced erasure coding techniques with configurable parameters to optimize bandwidth use and latency across system scales. Multiple protocol variants (BEAT0, BEAT1, BEAT2) further optimize trade-offs between encryption overhead, broadcast efficiency, and workload distribution, such as client-side

encryption offloading, to adapt performance to deployment scenarios.

Dumbo [74] focuses on minimizing communication complexity and reducing latency bottlenecks intrinsic to HoneyBadger's ACS protocol. Instead of running asynchronous binary agreements (ABA) for every node, Dumbo employs small fixed-size committees or multi-valued validated Byzantine agreement that drastically reduces the number of necessary ABA instances per round to a small constant, often independent of the total network size. These innovations also include a provable reliable broadcast primitive that uses threshold signatures to produce concise, publicly verifiable proofs of message acceptance, ensuring scalable and secure verification. Dumbo's pervasive use of threshold cryptography streamlines authentication and generates unbiased common coins, critical for ensuring liveness and consistency in fully asynchronous adversarial contexts, resulting in markedly improved throughput and scalability.

Probabilistic consensus protocols differ from deterministic consensus in their reliance on asynchrony, probabilistic guarantees, leaderless or decentralized coordination, and the extensive use of threshold cryptographic techniques to secure randomness and authentication. These designs have advanced into practical, scalable, and censorship-resistant consensus engines. They underpin modern distributed ledgers, blockchain platforms, and large-scale fault-tolerant systems that demand operation in unpredictable and adversarial environments. The key contributions of these protocols are summarized in Table 3.3, highlighting the evolution of probabilistic BFT from foundational cryptographic models to optimized, modular solutions established for diverse real-world deployments.

Table 3.3: Summary of key innovations from the presented probabilistic BFT protocols.

Protocol	**Key Contributions**
SINTRA [70]	Modular protocol architecture; advanced cryptographic primitives with threshold signatures and threshold coin-tossing; stackable, formally specified Reliable and Consistent broadcast layers.
Byzantine-Resistant Protocol without Signature [71]	Signature-free BFT consensus using only symmetric cryptography (MACs); pairwise symmetric keys for authentication; accumulation of valid MACs for quorum-based decision making. Avoids expensive public-key operations; improved performance and scalability while ensuring robust security.
HoneyBadgerBFT [72]	Threshold encryption for confidential transaction batching; ACS protocol combining reliable broadcast and parallel ABA instances; erasure-coded reliable broadcast to reduce bandwidth.
BEAT [73]	Modular architecture for flexible customization; labelled threshold encryption to maintain confidentiality and prevent replay attacks; threshold pseudorandom function for efficient, verifiable randomness; advanced erasure coding for bandwidth optimization; variants (BEAT0, BEAT1, BEAT2) balancing encryption and broadcast trade-offs.
Dumbo [74]	ACS revamp reducing ABA instances via a small fixed-size committee and multi-valued validated Byzantine agreement; provable reliable broadcast with threshold signatures for scalable verification; pervasive threshold cryptography integrating authentication and common coin generation; near-constant round complexity, improved throughput, and scalability.

3.5. Discussion

This section compares the main characteristics of deterministic and probabilistic BFT-SMR protocols presented in the previous sections. Table 3.4 synthesizes the key differences across the two paradigms.

Table 3.4: Comparison between the different types of consensus used in BFT protocols. (Auth. - Authentication, Compl. - Complexity, Crypto. - Cryptography). Adapted from [64].

Consensus	Properties	Comm. Model	Fault Model	Auth.	Compl.
Deterministic	Safety, Liveness	Partial Synchrony	$f \leq \frac{1}{3}N$	Symmetric/ Asymmetric crypto.	$\mathcal{O}(1)$
Probabilistic	Agreement, Total Order Liveness	Asynchronous	$f \leq \frac{1}{3}N$	Threshold crypto.	$\mathcal{O}(n)$

When examining BFT protocols, deterministic and probabilistic consensus mechanisms reflect fundamentally distinct approaches to achieving agreement in the presence of Byzantine replicas. Deterministic consensus delivers absolute finality; once committed, transactions cannot be reversed, and honest nodes will never diverge. By contrast, probabilistic consensus achieves eventual consistency, where finality grows with confirmations but is never mathematically absolute.

Deterministic BFT uses structured rounds or phases (pre-prepare, prepare, commit), which are predictable but have communication complexity that grows quadratically with the number of nodes. They are considered best suited for permissioned networks where strong finality and predictable performance are critical.

Probabilistic BFT tends to achieve better scalability and lower message complexity, which is more appropriate for larger and permissionless networks. The trade-off is the absence of instant finality, since decisions only become statistically irreversible over time.

Deterministic consensus can tolerate up to $f = \lfloor \frac{(n-1)}{3} \rfloor$ Byzantine faults while maintaining strong liveness and safety guarantees, provided the network eventually delivers all messages. Safety violations in such systems are considered fatal.

Probabilistic consensus can tolerate similar numbers of faulty nodes, but both safety and liveness are guaranteed only with high probability. The risk of divergence is low and decreases exponentially with more confirmations, but it is never entirely zero.

Although not addressed in the previous sections, Deterministic BFT is often used in enterprise, consortium blockchains, and critical infrastructure where strong guarantees and auditability are essential. These systems sacrifice scalability for reliability and predictability in trusted environments.

Probabilistic BFT emphasizes scalability and decentralization, making it suitable for public blockchains and open participation. They trade strict, instant guarantees for openness and large-scale resilience.

In a general overview, each type of consensus provides the following key contributions and innovations:

- Deterministic protocols:
    1. Strong safety and liveness in synchronous and eventually synchronous networks.
    2. Reductions in message exchange complexity.
    3. Mechanisms for leader rotation and fair progress during view changes.
- Probabilistic protocols:
    1. Use of randomness and stake-weighted leader selection to improve robustness.
    2. Analyses of chain quality, fork probability, and network security assumptions.
    3. Techniques enabling thousands of participants to reach consensus without heavy message rounds.

The core distinction between deterministic and probabilistic consensus in BFT lies in the certainty and performance guarantees they offer. Deterministic protocols provide iron-clad assurances suited to controlled, permissioned settings, while probabilistic protocols enable large-scale decentralization with a small uncertainty risk. The choice is a trade-off between absolute safety under bounded network conditions and scalable openness with probabilistic guarantees, a decision that shapes the architecture, trust assumptions, and performance envelope of the entire system.

3.6. Chapter Summary

This chapter provided an overview of consensus mechanisms that constitute the foundation of ITSs, enabling them to achieve reliable agreement. The discussion distinguished between deterministic and probabilistic approaches, beginning with core concepts in BFT and SMR, and explaining how these protocols underpin resilient distributed systems capable of masking arbitrary faults, even in Byzantine environments.

The chapter explored the theoretical landscape shaped by the FLP impossibility, contrasting the guarantees and practical limitations of deterministic and probabilistic consensus. Deterministic protocols, typified by PBFT and its iterations (e.g., Zyzzyva, MinBFT, CheapBFT, BFT-Mencius, Hybster, Mod-SMaRt), were shown to provide strong safety and liveness under partial synchrony, leveraging innovations such as efficient view-changes, speculative execution, trusted hardware, resource-aware adaptivity, multi-leader architectures, parallelization, and modular consensus engines. Their predictable operation and absolute finality make them well-suited for critical, permissioned infrastructures where consistency is paramount.

In contrast, probabilistic (or randomized) BFT protocols, represented by SINTRA, the signature-free approach by Correia et al., HoneyBadgerBFT, BEAT, and Dumbo, overcome deterministic limitations by introducing cryptographic randomness and threshold techniques. These protocols achieve consensus with high probability in fully asynchronous and dynamic networks, bringing leaderless architectures, threshold encryption, modularity, scalable broadcast via erasure coding, and communication-efficient agreement to the forefront. Such innovations enable large-scale scalability and resilience against unpredictable conditions, at the expense of rapid, absolute finality.

A detailed comparison highlighted that deterministic consensus offers ”iron-clad” safety and liveness in bounded network conditions but typically incurs higher overhead and limited scalability, making it optimal for trusted, closed environments. Probabilistic consensus,

while sacrificing immediate certainty, delivers better scalability and flexibility, supporting open and decentralized deployments where resilience against asynchrony and censorship resistance is vital.

In summary, the evolution and diversity of consensus protocols surveyed in this chapter reflect the shifting requirements of ITS, from stringent reliability and predictability to large-scale, robust, and resource-efficient design. The innovations documented herein not only shaped the architecture of systems like Skynet [75] but continue to drive research at the intersection of distributed fault tolerance, cryptography, and practical security. This survey provides a foundation and an overview through which to assess protocol selection and architectural trade-offs in the design of secure, fault- and intrusion-tolerant infrastructures.

4. Skynet's Architecture

This chapter presents Skynet [75], the main contribution of this thesis, an intrusion-tolerant architecture designed to defend against increasingly sophisticated, multi-domain cyber threats. Typical BFT protocols guarantee consensus even if some nodes act maliciously, but they largely ignore vulnerabilities within the hosting nodes themselves, assuming correct execution is enough. A handful of later designs introduce periodic or reactive recovery, yet these remain insufficient against modern attack strategies.

Today's adversaries, especially APTs, target the entire software–hardware stack, exploiting attack surfaces from Software-Defined Networking (SDN) poisoning to ML model tampering to evade detection. Existing defenses address individual vectors but seldom integrate across domains, and typically rely on unrealistic assumptions about monitoring data's trustworthiness.

Skynet bridges intrusion tolerance, cybersecurity, and ML into a unified, adaptive architecture. It acts as a secure overseer, merging the roles of traditional SIEM systems and orchestrators while building on the fundamental principles of previous ITS designs. Skynet is designed to dynamically adapt to both known and unknown threats, with the aim of minimizing vulnerability windows and maximizing resilience.

Traditionally, BFT protocols have focused primarily on ensuring protocol execution and consensus, with little regard for security vulnerabilities or exploits affecting the underlying nodes. Most protocols treat node compromise as irrelevant unless it directly disrupts consensus, with only a few incorporating proactive or reactive rejuvenation strategies. However, the escalating sophistication and persistence of cyber threats have changed the landscape: adversaries now exploit the entire stack, from hardware flaws to software vulnerabilities, using multi-stage, multi-domain tactics.

Modern threats often involve techniques such as poisoning SDNs, tampering with ML models, and exploiting vulnerabilities in system design.

These complex attacks highlight the insufficiency of perimeter-focused and single-domain security strategies. Defensive solutions exist, but typically tackle isolated aspects of these challenges, often making strong assumptions about the integrity of event probes or monitoring infrastructure.

To address these issues, this thesis argues for disrupting the traditional segmentation of knowledge domains. By merging principles from ITSs, cybersecurity, and ML, systems

can be built that are capable of preemptively, proactively, and reactively addressing diverse types of threats.

Skynet embodies this vision:

- It unifies SIEM and orchestration functions on an intrusion-tolerant foundation.
- It is open-source and built for dynamic adaptation: able to respond rapidly to known and unforeseen threats, reducing the time systems remain vulnerable.

Although past ITS designs (for example, TRONE's resilient pub-sub middleware [46]) have often been dismissed for their complexity or perceived overhead, and while crash-tolerant models eclipse BFT-SMR due to resource concerns, this thesis hypothesizes that reimagining ITS through cybersecurity and ML integration leads to fundamentally stronger platforms. By enhancing classic recovery strategies with predictive ML, dynamic reconfiguration informed by risk assessment, and richer detection through endpoint and network insights, Skynet has the potential to advance the research field.

The remainder of this chapter is organized as follows. Section 4.1, describes the requirements and assumptions underlying the architecture, including both the system model and the threat model. Section 4.2 then presents the design principles and rationale, outlining the theoretical foundations, design criteria, and trade-offs that shaped Skynet. Section 4.3 provides a high-level overview of the architecture, introducing its main modules. Section 4.4, details the principal dataflows, illustrating how information and control signals traverse between components. Section 4.5 presents the integration strategy, specifying how the modules are interconnected, orchestrated, and made interoperable in practice. Evaluation of the architecture is performed indirectly through the assessments of its individual modules, presented in later chapters. Finally, Section 4.6 provides a summary and concluding remarks for the chapter.

4.1. Requirements and Assumptions

Skynet's architecture has been designed in response to the increasing sophistication of modern cyber attacks, which increasingly employ multi-domain strategies and exploit vulnerabilities across the software, hardware, network, and operational stack [75]. It is architected as a quasi–Zero Trust overseer platform that unifies BFT protocols, Trusted Execution Environments (TEEs), advanced monitoring, automated threat intelligence, risk-aware configuration, and distributed, tamper-resistant provenance across both control and execution

planes. This section details the operational system model and adversarial threat model that guided its design.

4.1.1. System Model

Skynet operates as a distributed system of n independent nodes deployed across heterogeneous environments (on-premise, cloud, and hybrid). Each node runs on commodity hardware with support for TEEs, such as Intel SGX, ARM TrustZone, or AMD SEV-SNP, used to create hardware-isolated enclaves for secure processing. Nodes support general-purpose OSs, virtualization, and standard networking stacks.

All inter-node communication is authenticated and encrypted to guarantee confidentiality and integrity. The network is modeled as asynchronous or partially synchronous, accommodating delays, message reordering, and temporary partitions without strict delivery deadlines.

Consensus and coordination within Skynet rely on BFT protocols, deployed in the Execution Plane. These protocols tolerate up to $f < \frac{n}{3}$ Byzantine nodes per replica group, maintaining safety and liveness through quorum-based agreement mechanisms independent of network timing assumptions.

The architecture features two planes: a trusted Control Plane and an untrusted Execution Plane. The Control Plane orchestrates node deployment and hosts security-aware modules for system monitoring, reconfiguration, and resilience enhancement. It may be centrally hosted, provided it is robustly secured, physically, logically, or both, and maintains trusted communication channels with any other Control Plane instances, with the goal of enhancing resilience. The Control Plane never depends on the Execution Plane for its own security, although it interacts with the Execution Plane to deploy configurations, collect telemetry, and initiate corrective or adaptive measures.

Operational correctness of the protocols does not require precisely synchronized clocks; time synchronization (e.g., via Network Time Protocol (NTP) or Precise Time Protocol (PTP)) is only needed for log correlation and forensic analysis. At each node, the minimal TCB includes cryptographic primitives for authentication and encryption, secure configuration channels, and only critical Control Plane logic executed inside TEEs (for key management, attestation, and integrity verification). The trust boundary surrounds these TEEs-protected components, while the remaining Control Plane logic operates outside the enclave on a hardened system.

The Execution Plane is entirely outside this trust boundary and, because it is directly exposed to external networks and users, is more likely to be compromised by adversaries. To address this risk, Skynet incorporates security mechanisms specifically designed to contain and mitigate the impact of such Execution Plane breaches, with the aim of enabling the overall system to remain resilient even if some nodes or services are taken over by attackers.

These assumptions establish the adversary model described in the following section.

4.1.2. Threat Model

The threat model assumes deployment in adverse environments, where adaptive, well-resourced adversaries may launch multi-domain, multi-stage attacks targeting hardware, OSs, middleware, networks, and applications. Attackers are considered capable of discovering and exploiting vulnerabilities, escalating privileges, conducting insider and supply-chain attacks, manipulating network protocols, injecting or poisoning data, and executing persistent intrusion campaigns.

The trust boundary encompasses only the enclave-protected core of the Control Plane, where critical security functions such as key management, attestation, and verifiable provenance sealing are executed within TEEs (e.g., Intel SGX, ARM TrustZone, AMD SEV-SNP). Compromise of the TEE-protected core is considered theoretically possible but with very low probability, requiring a high amount of resources and advanced technical expertise; such attacks are rational only for highly motivated and well-resourced adversaries. The remainder of the Control Plane runs on a hardened platform with strong isolation and secure, authenticated communications, while not impossible to compromise, successful attacks are considered unlikely under the assumed adversary model.

The Execution Plane, fully outside this trust boundary, is assumed to have a high probability of compromise due to direct exposure to external networks and clients. It includes the BFT protocol stack, operational workloads, virtualization and rejuvenation mechanisms, monitoring and introspection agents (e.g., Host-based Intrusion Detection System (HIDS), File Integrity Monitoring, extended Berkeley Packet Filter), log-forwarding clients, automated penetration-testing and red-team agents, and exposed service interfaces. All of these modules can potentially fall under adversarial control: BFT protocol instances may be degraded or disrupted on compromised nodes; monitoring agents and pen-testing modules may be bypassed, disabled, misled, or repurposed; log-forwarders cannot guarantee integrity until records reach the trusted enclave; and networked interfaces may be subverted

for intrusion, lateral movement, or data exfiltration. The system maintains safety and liveness as long as fewer than $n/3$ replicas per group are Byzantine and a BFT quorum of $2f + 1$ correct replicas is upheld.

Adversaries may also seek to undermine security indirectly, for example, by poisoning external vulnerability and threat intelligence feeds, misleading the ML-based risk manager, manipulating SDN controls, or corrupting time synchronization services (NTP/PTP). While such attacks can impair forensic accuracy or reconfiguration logic, they do not impact the fundamental correctness or liveness of the core protocols.

Under these assumptions, Skynet is intended to provide at the system level: confidentiality, including confidentiality under compromise, integrity, availability, and consensus correctness, provided the enclave-protected Control Plane core remains uncompromised and a strict BFT quorum is preserved in the Execution Plane. Additionally, module-specific properties are provided only by subsystems designed with special protections; for example, only the provenance subsystem ensures audit records are tamper-resistant and verifiable once sealed inside the trusted enclave, regardless of Execution Plane compromise. Other modules retain their intended security properties only within the operational limits defined by these adversary and trust model assumptions.

4.2. Design Principles and Rationale

The Skynet architecture is conceived as a direct response to the rapid advancement and increased complexity of cyber threats that span multiple technical layers. Its architecture does not foster siloed security practices, but purposefully combines theory and practical advances from ITSs, cybersecurity, and ML to potentially address multi-domain attacks and the persistent risk of systemic compromise. This section details the conceptual foundations, design priorities, and explicit architectural boundaries underlying the creation of Skynet.

4.2.1. Theoretical Foundations

At its core, Skynet fuses insights from previous research in intrusion tolerance, merging the techniques that provide pure prevention with the methods that provide resilience and survivability in adversarial environments. Foundational work in ITSs, such as proactive and reactive recovery, BFTs, and the management of diversity, shapes the system's structure and protocols. Cybersecurity principles, particularly the NIST functions of identify, protect, detect, respond, and recover, have shaped the risk management life cycle and the Skynet's approach to both threat anticipation and mitigation. ML, besides being used for

the detection of anomalous events, is also integrated to enable continuous adaptation of system configurations based on real-time analysis of vulnerabilities and exploits. Finally, the design is underpinned by a strict separation between the privileged Control Plane and the potentially untrusted Execution Plane, a strategy informed by long-standing defense-in-depth paradigms, specifically shown in the works of Verissimo et al. [76], Kreuts et al. [77], and Saltzer et al. [78].

4.2.2. Design Criteria and Trade-offs

Designing Skynet required a delicate balance between security, availability, adaptability, and overhead. The platform favors a modular, adaptable architecture that enables the seamless integration of new threat intelligence sources, forensic methods, and orchestration capabilities. Rather than pursuing the goal of absolute prevention, Skynet's structure tolerates and endures intrusions, minimizing service downtime and the risk exposure window. Operational overhead, often cited as a barrier for the adoption of intrusion-tolerant technologies, is consciously addressed through a combination of fixed and adaptive control strategies, leveraging automation where possible but anchoring key decisions in practical efficiency. While ML offers crucial automation for dynamic reconfiguration and incident response, the design recognizes that, for exceptional cases and deep incident forensics, human oversight remains essential. The system's approach to patch management further illustrates this balance: strong offline patching guarantees security but increases recovery time, while prospective online patching aims to optimize availability, albeit with known technical limitations, especially in terms of system state consistency.

4.2.3. Architectural Constraints and Assumptions

The design of Skynet is grounded in several core architectural constraints and explicit assumptions that underpin the system model and shape all subsequent design decisions.

First, a foundational assumption is a strict trust separation: the Control Plane is presumed to operate within a secured, trustworthy environment, benefiting from strong isolation, secure communication channels, hardware-based trust anchors (such as TEEs), and cryptographic protection. In contrast, the Execution Plane is considered untrusted and vulnerable to adversarial compromise; it is directly exposed to attacks and is expected to host services and protocols, such as BFT, that can operate correctly despite the presence of faulty or malicious nodes.

The model presumes the availability of essential hardware and software capabilities, specifically the presence of general-purpose OSs, virtualization support, secure storage, and network stacks capable of authenticated and encrypted communications. Where required, commodity hardware is assumed to support trusted execution and minimal TCBs for core security functions, such as key management and remote attestation.

Recognizing the risk that system probes, logging agents, or telemetry collectors may be compromised or manipulated, Skynet does not rely on any single component or data source. Instead, the architecture mandates redundancy and continuous verification, ensuring no individual misbehaving element can undermine system assurance. Data integration mechanisms are designed to handle heterogeneous, potentially noisy, or delayed information from both external and internal sources, supporting robust real-time decision making.

A key challenge in resilient system design, implementing true N-diversity in large-scale deployments, is acknowledged as a limiting factor for practical systems. Skynet addresses this by integrating automated diversity management and risk quantification, leveraging ML and live threat intelligence to provide varied and adaptive configurations where possible.

While automation is a central goal, particularly for detection and adaptive response, the architectural model explicitly allows for manual intervention. For operations where full automation is either infeasible or currently unreliable, such as exceptional incident response or deep forensic analysis, human oversight remains an integral part of the system's defensive posture, through the integration of SIEMs.

In summary, these constraints and assumptions delineate the operational and trust boundaries of Skynet, ensuring that its intrusion-tolerant approach is both robust and practically grounded.

4.3. High-Level Architectural Overview

This section provides an overview of the Skynet architecture, emphasizing the interplay between its foundational components and their roles in ensuring robust intrusion tolerance and adaptive defense. Skynet's design follows a multi-plane approach in which system responsibilities and trust boundaries are clearly separated. The Control Plane is entrusted with orchestration, monitoring, and risk management, operating within a secured environment, while the Execution Plane is deliberately assumed to be exposed and potentially compromised, running the core replicated services and application workloads. This layered architecture enables the system to flexibly and proactively respond to evolving attack vectors and operational changes. The Execution Plane is untrusted and directly exposed to

adversaries, hosting the operational workloads, protocol replicas, and supporting infrastructure. These planes interact to preserve system integrity, availability, and resilience in the face of both known and emerging threats.

Figure 4.1 illustrates the primary modules and their relationships across both planes, highlighting how coordinated functionality and defense-in-depth principles drive the overall resilience of the architecture. The following description introduces the key modules and their interconnections, providing the context for more detailed technical discussions in subsequent sections.

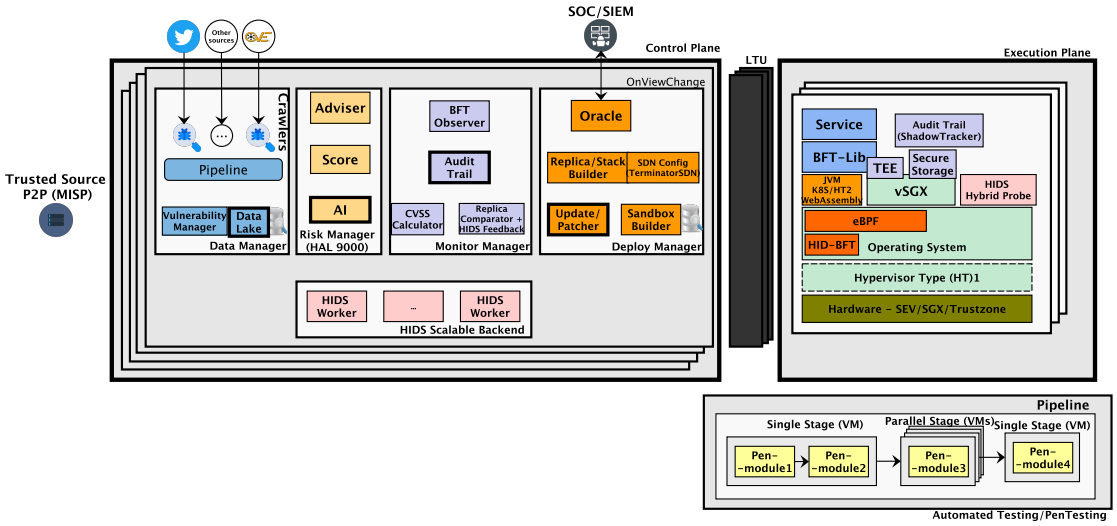

FIGURE 4.1: Skynet's architecture high-level overview. The two planes are delimited by labelled enclosing boxes and separated by the Local Trusted Unit: the trusted Control Plane (left) and the untrusted Execution Plane (right), with stacked outlines denoting replicated instances. Rectangles denote processing modules and circles denote external sources and services. Arrows indicate the direction of data flow. The same notation applies in Figures 4.2 and 4.3.

4.3.1. Control Plane Modules

Figure 4.2 expands the Control Plane from the overview in Figure 4.1, detailing the internal composition and data flows of its five modules. As the trusted and protected "brain" of the system, the Control Plane comprises the Data Manager, the Risk Manager, the Monitoring Manager, the Deploy Manager, and the HIDS Scalable Backend.

Data Manager (VExHK): The Data Manager, labelled VExHK*, module scrapes vulnerability and threat intelligence by gathering data from public sources, such as the

*Pronounced "Vek"

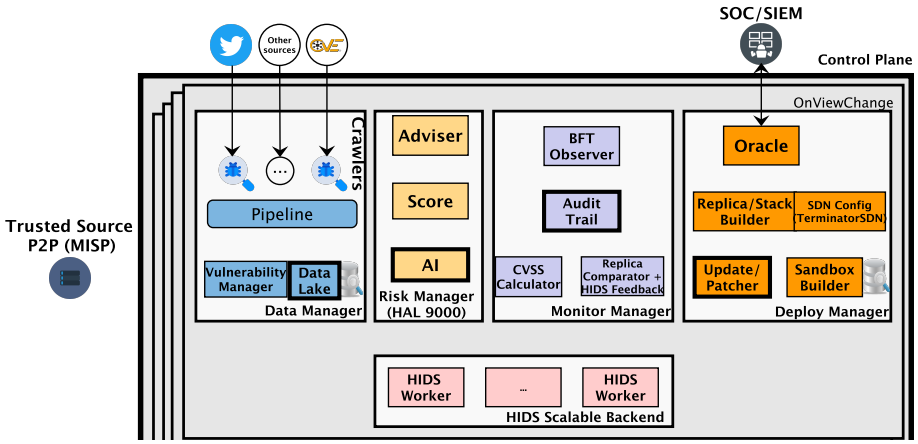

Figure 4.2: Skynet's intelligence side, integrating threat-intelligence ingestion, vulnerability-risk assessment, BFT monitoring, automated deployment, and remediation over a scalable HIDS backend.

NVD [79], CVE feeds [80], VulnDB [81], and social media (e.g., X [82]*), as well as internal sources like automated testing and penetration testing modules, and the information gathered by forensic analysis and collected through logs from exposing the configuration to a volatile environment. This information is stored centrally in a database to inform risk assessments and adaptive security measures used by other modules, specifically the Risk Manager and the HIDS Scalable Backend. In addition, the module provides feedback with useful information for threat sharing platforms, e.g., Malware Information Sharing Platform (MISP) project [83].

By gathering the informatiion locally *a priori*, i.e., as a service that periodically collects the data, allows Skynet to be prepared for possible threats and intrusions, lowering the time window to autonomously act against new and old threats, removing the barriers that current databases have against too many requests to avoid possible Denial-of-Service threats or to balance out the requests from other connections. Implemented as an open source tool and in a modular manner, it allows for further addition of sources to increase its range of collected data.

Risk Manager (HAL): The Risk Manager, labelled "HAL 9000", module acts as the system's configuration oracle, evaluating hardware and software stacks for known vulnerabilities, exploit probabilities, and threat exposure in order to guide the most secure and resilient deployments. Consistent with intrusion-tolerant architectures, which require at least $3f + 1$ replicas for BFT, HAL enforces a diversity policy that ensures no two nodes share OSs with overlapping vulnerabilities.

*Formerly Twitter, since 2023.

Beyond static assessment, HAL leverages ML techniques to cluster semantically similar CVEs. This method uncovers hidden relationships between vulnerabilities with distinct IDs that affect different platforms but have shared exploitation patterns, thus strengthening diversity-aware configuration choices.

A persistent challenge for automated ITS risk managers is the delay in CVE evaluation at the NVD, whose backlog often postpones CVSS scoring [23]. To address this, HAL integrates a CVSS score prediction model that uses Natural Language Processing (NLP) on vulnerability descriptions, generating temporary scores for newly reported CVEs still awaiting official evaluation. By combining these predictive scores with exploitability likelihood metrics such as EPSS and curated OSINT sources (e.g., ExploitDB, AlienVault OTX Pulses), enables rapid, adaptive risk assessments against both known and emerging threats.

HAL balances two complementary perspectives: (1) a security score, quantifying the risk of a single node being compromised, and (2) a resilience score, quantifying shared risks across nodes to prevent "break one, break all" situations. By jointly optimizing both, HAL is intended to ensure that the system is not only individually secured but also collectively resilient, aligning with Skynet's vision of a next-generation, self-adaptive intrusion-tolerant overseer.

Monitoring Manager: The Monitoring Manager is a cornerstone of Skynet's architecture, responsible for providing comprehensive oversight, intrusion detection, and behavioral analysis across the deployed configuration. Its main objective is to safeguard the integrity, availability, and trustworthiness of both individual replicas and the distributed consensus layer that underpins the system. By combining detection, introspection, and forensic logging, this component addresses critical blind spots left by conventional intrusion detection mechanisms and BFT protocols.

At the core of this module is the replica comparator, which leverages a trusted baseline state to compare with replicas deployed into the exposed execution environment. By periodically cross-checking the stack, the comparator can identify discrepancies that may signal compromise or Byzantine behavior. This information is further correlated with HIDS feedback, including File Integrity Monitoring and malware detection alerts, thereby enhancing the accuracy of classification and reducing false negatives. Importantly, the comparator extends its role beyond immediate detection: it supplies curated intelligence to the Risk Manager, enriching its advisory process by reporting suspected compromises and contributing to configuration reassessment. When a novel vulnerability is identified, the comparator

employs an embedded CVSS-like calculator to produce a temporary severity evaluation, which is then forwarded to the Data Manager. This shortens the response time between vulnerability discovery and countermeasure deployment, allowing Skynet to act with agility in hostile environments. The comparator itself can operate during live execution, periodically withdrawing replicas for analysis without jeopardizing the global service, or in post-mortem scenarios, where compromised nodes are inspected to detect altered files and identify potential exploit signatures useful for forensic investigation and dataset enrichment.

Beyond replica-level monitoring, the Monitoring Manager supervises the health of the consensus layer through the BFT Observer. BFT protocols are designed to mask faults by removing misbehaving replicas through view-change subprotocols, yet they do not publicize these events outside the consensus process. The BFT Observer fills this gap by detecting when view-changes are triggered, providing explicit evidence that a replica has misbehaved or may be compromised. By relaying this information beyond the protocol's internal scope, it ensures that the system can take additional corrective measures to preserve the essential $3f + 1$ invariant even in cases where faulty replicas remain active but silently excluded from consensus.

Another critical function of the Monitoring Manager is its reliance on tamper-resistant audit trails as a source of verifiable historical evidence. These logs are generated at the node level by subsystems such as ShadowTracker, which operate within the Execution Plane to protect logs against tampering and ensure resilience under adversarial conditions. While capture and replication occur within the replicas, the resulting records are securely forwarded to the Monitoring Manager in the Control Plane, where they are correlated with other telemetry streams. This enables analysts to visualize, replay, and cross-reference events, facilitating anomaly diagnosis, attack reconstruction, and accountability across the system's lifecycle.

These mechanisms are intended to position the Monitoring Manager as Skynet's "nervous system", continuously sensing, correlating, and validating information from across replicas, consensus protocols, and external logs. Through its integration with the Risk Manager and Data Manager, it has the potential to transform raw telemetry into actionable intelligence, enabling proactive reconfiguration, adaptive risk evaluation, and enriched vulnerability datasets. By embedding both introspection and forensic persistence into Skynet's operation, the Monitoring Manager extends monitoring beyond passive detection and establishes it as an active safeguard that strengthens resilience against conventional intrusions,

Byzantine misbehavior, and emerging zero-day exploits.

Deploy Manager: The Deploy Manager is responsible for creating, maintaining, and evolving the system's node set by orchestrating deployment, updates, and patching in alignment with security policies. It receives configuration directives from the Risk Manager, in particular recommendations concerning safer software stacks and network configurations, and translates them into concrete rollout procedures in the execution environment. Each deployed node is constructed from its base system upwards until the full stack is operational, applying configuration hardening along the way and embedding the threat-aware settings determined by the TerminatorSDN subsystem. This close integration ensures that SDN adaptations are incorporated as nodes are brought online or reconfigured, thereby lowering exposure to known and emerging network threats.

Beyond initial deployment, the Deploy Manager also governs continuous lifecycle operations, including update scheduling, security patch distribution, and controlled re-deployment of nodes when vulnerabilities are discovered or when the environment changes dynamically. To achieve seamless coordination across the broader infrastructure, the Deploy Manager maintains communication with external oversight layers such as SOC and SIEM tools, ensuring that deployment changes are visible to operators and align with ongoing incident response. This role is critical to sustaining operational resilience, as it enables Skynet to respond rapidly to configuration shifts recommended by the Risk Manager, while systematically distributing changes through hardened orchestration that mitigates risks of propagation from a compromised node.

HIDS Scalable Backend: The HIDS Scalable Backend provides the analytical foundation required to execute scalable HIDS across all deployed nodes. Integrated in the Control Plane, it operates within a trusted environment shielded from potential compromise in the Execution Plane. This architectural separation ensures the integrity of detection logic and correlation processes, while simultaneously redirecting resource-intensive analysis to a more computationally capable environment. By relieving nodes of heavy processing demands, the backend reduces local performance overhead and minimizes opportunities for adversarial interference with detection operations. Its analytical processes are informed by rules, signatures, and threat intelligence provided by the Data Manager, ensuring that detection is continuously updated in light of new vulnerabilities and emerging threats.

Functionally, the backend aggregates telemetry streams originating from node-side agents and subjects them to large-scale analysis. Offloading intensive computation to the

backend not only improves the efficiency of exposed nodes but also enables a broader, system-wide perspective. This global visibility allows Skynet to detect patterns that may remain covert at the level of individual nodes but become apparent when correlated across multiple instances, such as coordinated attack campaigns or recurring exploit signatures.

The backend provides its outputs directly to the Monitoring Manager, strengthening detection workflows by supplying higher-fidelity alerts and enabling more robust intrusion response. In this manner, the HIDS Scalable Backend acts as a bridge between low-level node telemetry and high-level monitoring orchestration.

By distributing detection responsibilities into a dedicated and resilient analysis tier, the HIDS Scalable Backend has the potential to enhance both the accuracy and scalability of the intrusion response capability. This design aims to promote monitoring from an isolated node-level task into a coordinated, system-wide detection fabric, with the potential to enable Skynet to dynamically adapt to individual intrusions as well as systemic, multi-node attack campaigns.

4.3.2. Local Trusted Unit (LTU)

Positioned at the boundary between the Control and Execution Planes, the LTU acts as a secure conduit for management and telemetry traffic. It employs TEEs, public-key infrastructure, and secure communication protocols (such as Transport Layer Security (TLS)) to provide strong confidentiality, integrity, and authenticity of communications, even when operating in untrusted or partially compromised environments.

The LTU serves as a gatekeeper for interactions between planes, enforcing strict isolation and access control policies. On the Control Plane side, it interfaces directly with trusted modules and exposes a minimal attack surface, only allowing well-defined, authenticated operations (such as configuration delivery, software attestation, and collection of telemetry streams). On the Execution Plane side, it mediates control and data exchanges, relaying only validated information and enforcing protocol compliance on connected replicas or agents.

To achieve a strong security posture, the LTU leverages hardware-based trust anchors (e.g., Intel SGX, ARM TrustZone, or AMD SEV-SNP) to isolate critical logic from the host OS. All cryptographic key material and attestation processes are confined within the TEE boundary, reducing the risk of compromise by privileged Execution Plane adversaries. This

design ensures that sensitive operations, such as node registration, remote attestation, and signing of provenance records, are executed in a tamper-resistant manner.

The LTU incorporates mechanisms for secure, authenticated delivery of configuration and update bundles to Execution Plane nodes, as well as rate-limiting, anomaly detection, and fail-safe controls to prevent abuse. In the event of detected compromise or misbehavior on the Execution Plane, the LTU can quarantine affected nodes by revoking channel credentials or isolating communications, supporting rapid response and containment.

Furthermore, the LTU supports integrity validation and time synchronization for logs and audit trails transmitted from the Execution Plane, ensuring that anomaly and incident records forwarded to the Control Plane can be trusted for forensic and real-time analysis. It is also responsible for multiplexing and prioritizing telemetry from multiple replicas, efficiently balancing performance and security.

By serving as a robust bridge anchored in hardware-enforced trust and cryptographic guarantees, the LTU enables secure, auditable collaboration between the Control Plane's sensitive decision logic and the Execution Plane's potentially hostile environment. Its integration helps to narrow the trust boundary, ensuring that even in the presence of active adversaries, critical management and monitoring traffic can remain confidential, authenticated, and resilient to tampering or spoofing, even in adversarial conditions.

4.3.3. Execution Plane Modules

The Execution Plane, presented in Figure 4.3, comprises the system components responsible for running core application workloads and supporting defensive infrastructure in an exposed, potentially hostile environment. Its primary modules include the Node (or Replica) and Automated Testing/PenTesting, which together provide operational services, resilient state management, and continuous self-assessment of system security.

Node (Replica): The node is conceptualized as a robust, security-oriented system component that aspires to withstand advanced attacks and maintain operational continuity even in hostile environments. The underlying approach combines various technologies and defense mechanisms, each selected for its potential to enhance intrusion tolerance, accelerate recovery, and support forensic clarity.

The design includes hardware-based TEEs, such as Intel SGX, ARM TrustZone, or AMD SEV, to safeguard critical computations and cryptographic keys even if other system layers are compromised. This form of hardware isolation is intended to preserve essential

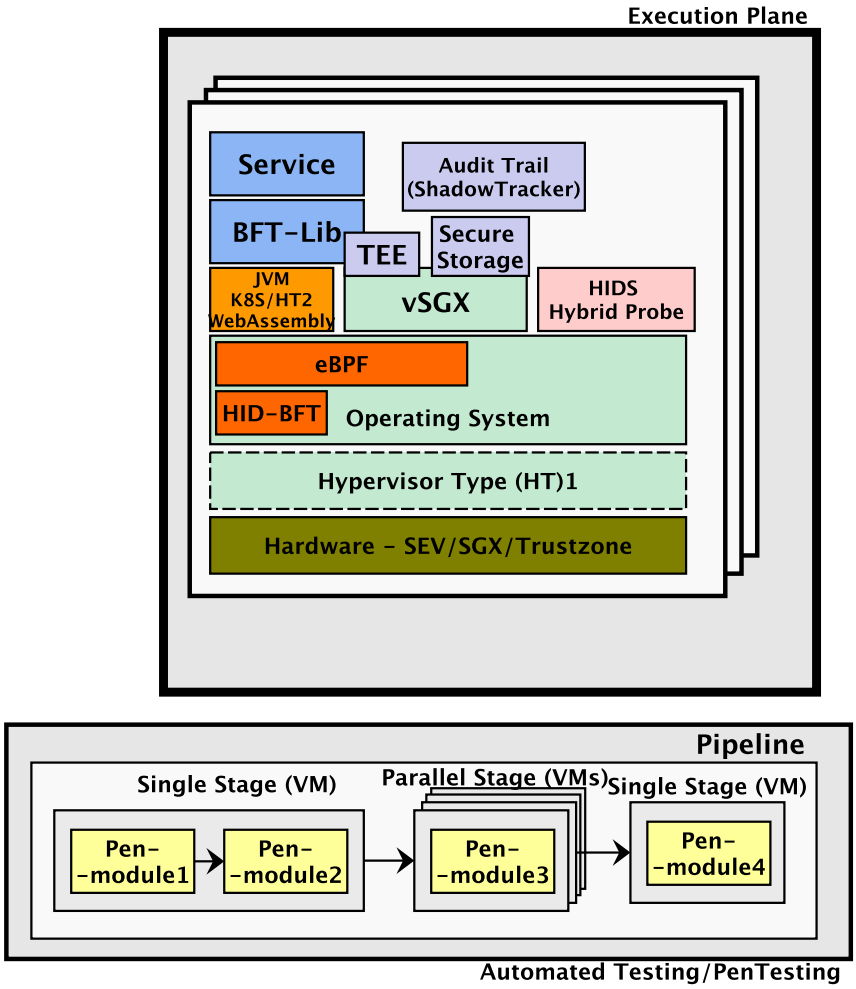

Figure 4.3: Overview of the Execution Plane.

security properties during attacks, as recognized in current technical literature [84, 85]. While such features hold promise, their practical effectiveness may vary depending on specific attack vectors and system integrations.

Layered virtualization, making use of both type 1 and type 2 hypervisors as well as technologies like vSGX, is suggested as a method for strengthening the isolation between instances and reducing the potential lateral movement of threats. These virtualization techniques could enable rapid restoration or replacement of compromised components, thereby possibly minimizing system downtime in response to security incidents. Actual recovery performance will likely depend on the environment and deployment specifics [86].

On the detection and monitoring front, advanced HIDS tools such as Wazuh [87] are intended to provide capabilities like File Integrity Monitoring, malware detection, and automated reactions such as isolation or rollback. The deployment of these monitoring

mechanisms is anticipated to improve the timeliness and effectiveness of intrusion detection and containment, although real-world performance may reflect operational and threat-model complexities.

Forensic integrity in this context is anticipated to be reinforced by secure storage, particularly leveraging Write-Once-Read-Many (WORM) strategies. The idea is that, when combined with distributed replication and cryptographic safeguards, WORM storage could substantially hinder an attacker's ability to alter or erase operational logs and forensic artifacts. Nonetheless, ensuring these protections at scale and in dynamic environments may present implementation challenges that affect their resilience in practice.

A key element of the architecture, ShadowTracker, is introduced as a distributed logging subsystem that aspires to be both tamper-resistant and tamper-evident. Through replication of logs across trusted nodes, the use of secure storage, and by embedding sensitive functions within TEEs, ShadowTracker is poised to protect audit trails and provenance information in the presence of ongoing threats. If successfully realized, this could limit adversarial efforts to obscure traces of compromise and support thorough forensic analysis, though empirical validation in demanding scenarios will be crucial for confirming its robustness.

The use of technologies such as secure OS introspection with extended Berkeley Packet Filter is considered advantageous for enabling fine-grained, real-time monitoring with limited overhead, thus contributing to more advanced detection and response capabilities. The actual security and operational benefits of these techniques will likely depend on their integration and the evolving threat and deployment landscape.

Automated Testing/PenTesting: This module is envisioned as a persistent and autonomous subsystem dedicated to strengthening the environment's resilience by subjecting running nodes to constant scrutiny through automated testing and penetration-testing tools. Leveraging widely used frameworks and utilities, such as Metasploit [88] for penetration testing, Hping3 [89] for network stress and packet manipulation, Harvester [90] for information gathering, and automated software testing platforms like Katalon [91] and TestComplete [92], this component orchestrates a broad range of security checks and attack simulations on deployed nodes.

The module's core capability lies in its ability to simulate real-world attack scenarios, probe for both known and previously undiscovered vulnerabilities, and exercise the full system stack under adversarial conditions without requiring manual intervention. Any

vulnerabilities or anomalous behaviors detected are automatically reported to the Data Manager, which then processes and disseminates this intelligence both within Skynet and to external threat-sharing platforms. This rapid, closed feedback loop enables Skynet to swiftly adapt configurations, deploy targeted patches, and update its detection heuristics, thereby significantly reducing the window of exposure to emerging threats.

In addition to vulnerability discovery, this module has the potential to contribute to proactive resilience by regularly stress-testing the system's ability to recover from discovered faults, measuring response times, and ensuring that automated remediation routines function as expected. Its integration with the rest of the architecture is designed to minimize the dependency on external threat databases by ensuring that new exploit information is rapidly surfaced, analyzed, and acted upon internally, making the environment less reliant on delayed or incomplete third-party intelligence.

While the approach promises substantial gains in threat agility and autonomous defense, its effectiveness in highly dynamic, large-scale deployments will depend on implementation choices and operational experience. However, by embedding continuous and comprehensive adversarial testing directly into the operational lifecycle of deployed nodes, Skynet aspires to maintain a defensive posture that is both adaptive and robust in the face of evolving attacks.

4.3.4. Inter-Plane Collaboration

Beyond the local deployment, Skynet instances are designed to collaborate, replicating knowledge, configurations, and threat intelligence across administrative domains for enhanced resilience and coordinated defense. Redundant Skynet deployments enable load balancing, exploit knowledge sharing, and cross-validation of security decisions.

This collaborative approach leverages self-organizing protocols capable of orchestrating multiple Skynet instances in a scalable and decentralized fashion. This federative approach promotes the dissemination of exploit signatures, recommended countermeasures, and forensic evidence among peer environments, allowing participants to leverage shared situational awareness and collaboratively developed defense strategies across the network. Through regular exchanges, instances can also assist in detecting complex, multi-domain, or coordinated attacks that might otherwise surpass the visibility of a single infrastructure.

The architecture envisages these collaborative mechanisms to not only increase fault tolerance and mitigate targeted attacks, i.e., by swiftly propagating security updates and

reconfiguration advice to all nodes, but also to support the formation of trust relationships and dynamic coalitions responsive to emerging threats. This is especially relevant for cloud, multi-datacenter, and distributed operational scenarios, where a single compromised environment can have cascading effects if not isolated and addressed promptly.

By embracing open standards for CIS and configuration interoperability, Skynet aspires to foster an ecosystem where independent administrative domains can securely and efficiently coordinate their defensive efforts without compromising local autonomy or data privacy.

Together, these modules aim to form a tightly integrated, modular, and adaptive architecture where intrusion tolerance is ingrained as a foundational operational principle—rather than retrofitted—maximizing both robustness and agility. Figure 4.1 visually summarizes these relationships and control flows within Skynet's layered defense, illustrating how collaboration across planes amplifies the system's collective security posture.

4.4. Dataflows

This section, illustrated in Figure 4.4, presents the dataflows that are essential to the operation of Skynet. It describes how information is ingested, evaluated, and propagated among specialized modules, enabling robust coordination and timely security responses. Each dataflow outlines the movement of information across core components, clarifying their interactions and highlighting the specific roles of principal architectural elements.

Four distinct process flows are defined: the (Re)Configuration flow, the Intrusion Identification flow, the Observation flow, and the Testing flow. These flows are detailed in the following subsections.

4.4.1. (Re)Configuration process flow

The (Re)Configuration process flow, illustrated in Figure 4.5, unfolds as a continuous pipeline that adapts the system to evolving vulnerabilities and threats. It begins with **(1)** the Data Manager, which aggregates vulnerability and exploit information from external sources and integrates them into the Data Lake. This knowledge is evaluated by the HAL, which estimates both resilience and security scores of the possible configurations from the available hardware and software.

The recommended configurations are then passed to **(2)** the Deploy Manager, where the Oracle component interprets these recommendations to construct or update node stacks. At this stage, the SDN settings are provisioned through the TerminatorSDN module, ensuring

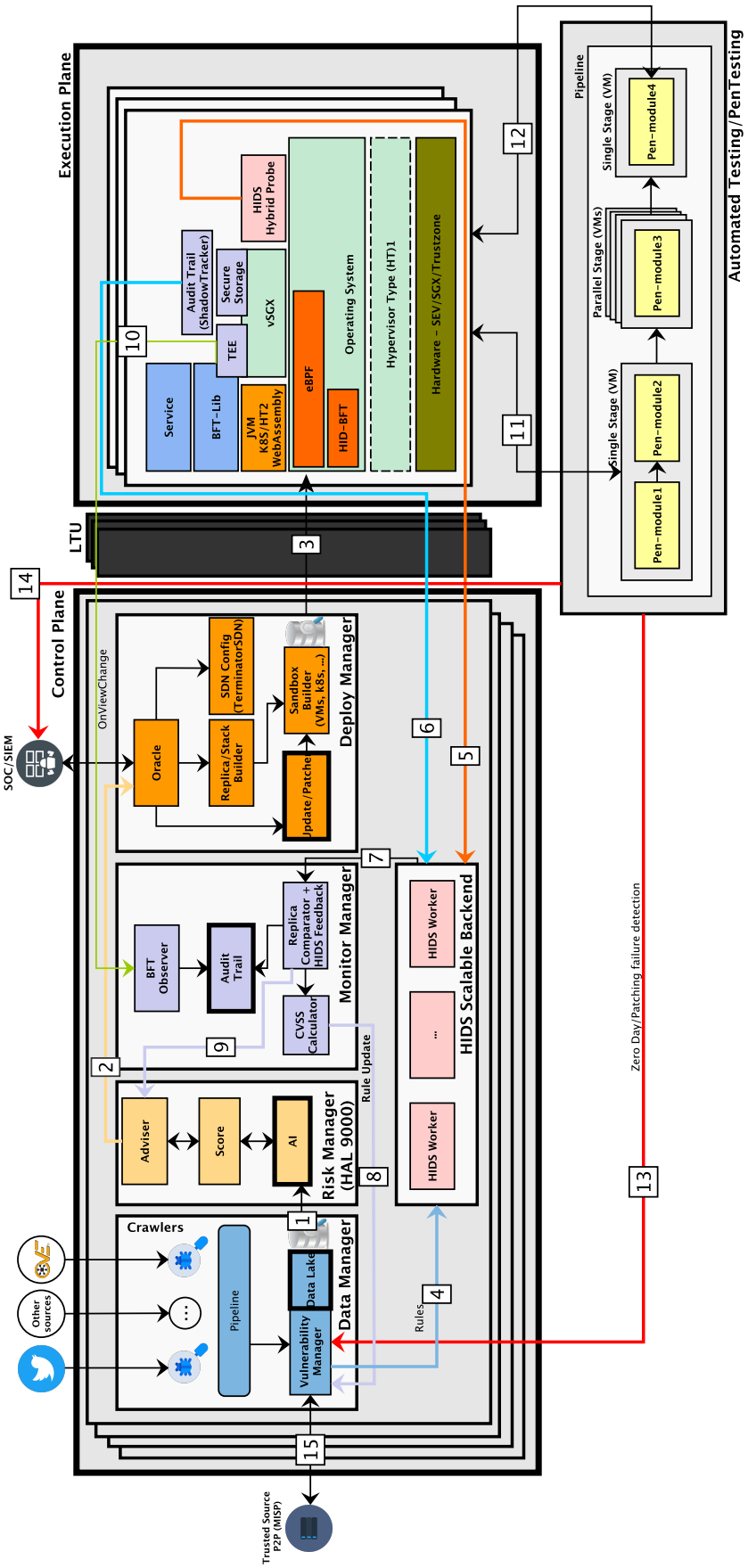

Figure 4.4: Dataflows in Skynet, showing information exchange between core modules across the control and execution planes.

that both the system and the network layers remain aligned with the prescribed resilience and security requirements.

Finally, the prepared node stacks are transmitted to **(3)** the Execution Plane, where they are deployed and instantiated. This stage operationalizes the updated configuration across the infrastructure, completing the (Re)Configuration cycle while ensuring that the architecture remains continuously adaptive and resilient to emerging security challenges.

4.4.2. Intrusion identification process flow

The Intrusion identification process flow, illustrated in Figure 4.6, continuously refines its detection capability by integrating three main inputs: knowledge maintained in the Data Manager, telemetry generated by deployed HIDS probes, and integrity logs preserved by the ShadowTracker. Together, these data sources enable the system to detect intrusions and identify the presence of malicious agents.

The process begins with **(4)** the periodic update of the detection rules managed within the HIDS Scalable Backend. In parallel, **(5)** it ingests operational data from the HIDS Workers along with **(6)** the tamper-resistant records of the ShadowTracker. The combined information and resulting feedback are subsequently relayed **(7)** to the Monitor Manager for cross-checking and system-wide correlation of events.

Potential intrusions are flagged when the active rule set enforced by the HIDS is violated or when unexpected deltas emerge during the Replica Comparator's integrity checks. When such anomalies occur, the system produces alerts together with detailed forensic evidence. These outputs are evaluated by the CVSS Calculator, and the final results are **(8)** stored in the Data Manager to update the vulnerability repository and extend the knowledge base for future detection.

4.4.3. Observation process flow

The Observation process flow, illustrated in Figure 4.7, enhances situational awareness by combining telemetry and anomaly detection from multiple sources. At its core, the process integrates three key inputs: telemetry generated by the HIDS Workers **(7)**, anomaly reports provided by the BFT Observer **(10)**, and the subsequent risk analysis performed by HAL **(9)**. This allows for continuous surveillance and adaptive response across the infrastructure.

The process begins with **(7)** the collection of telemetry and feedback from the deployed HIDS Workers, which provide insight into system and node-level activity. This information

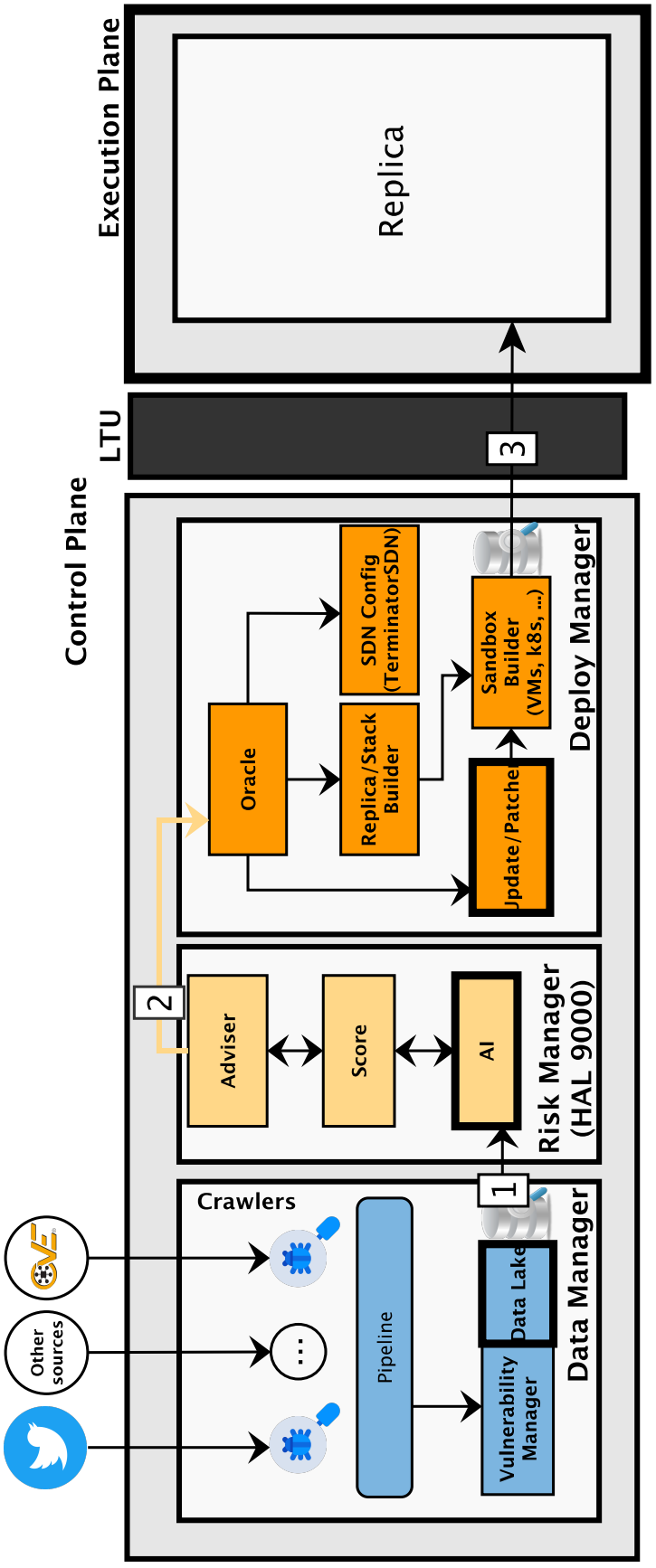

Figure 4.5: (Re)Configuration process: (1) Data Manager sends the requested data to HAL; (2) Deploy Manager receives the advised configuration from HAL to prepare for deployment; (3) the nodes are sent to the Execution Plane.

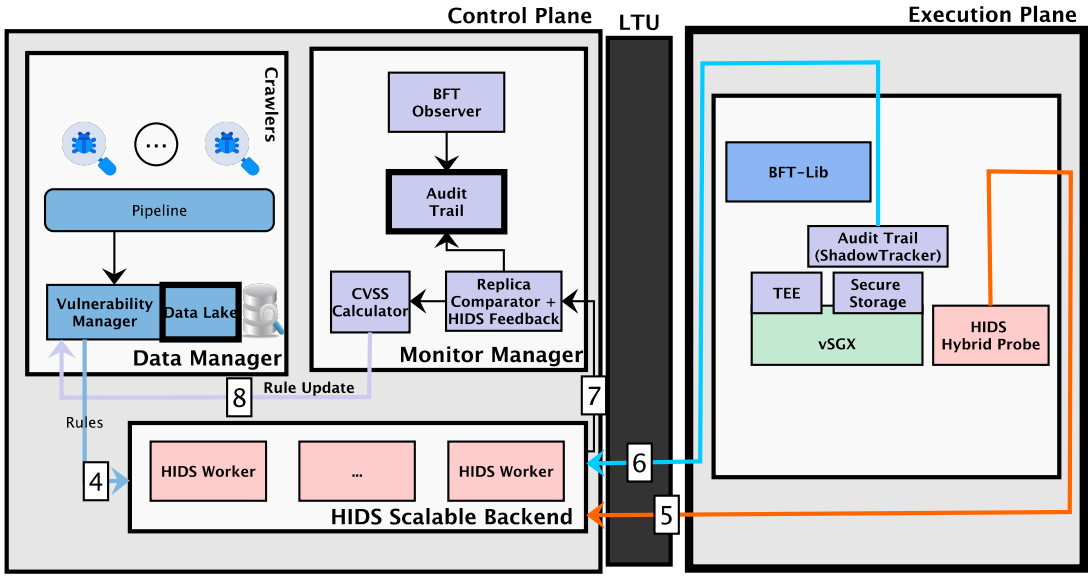

FIGURE 4.6: Intrusion identification process flow. (4) The Data Manager provides updated detection rules to the HIDS Scalable Backend; (5) telemetry from the HIDS Workers and (6) tamper-resistant logs from the ShadowTracker are processed by the Backend; (7) the Monitor Manager correlates this information; (8) the Data Manager stores the results to update the vulnerability knowledge base.

is complemented by **(10)** the BFT Observer, which identifies performance degradations and anomalies in the consensus layer. The combined results are then consumed by **(9)** HAL, which evaluates the overall system state, performs risk scoring, and advises on suitable reconfiguration strategies to preserve the security and resilience of the infrastructure.

4.4.4. Testing process flow

The Testing process flow, illustrated in Figure 4.8, systematically identifies vulnerabilities and weaknesses across deployed nodes through integrated penetration and autonomous testing, and continuous feedback. At its core, the process involves three coordinated stages: execution of testing by the Automated Testing/PenTesting modules **(11)** and **(12)**, aggregation and classification of findings within the Data Manager **(13)**, and the sharing of actionable intelligence to external platforms via the Data Manager's communication with threat sharing platforms (e.g. MISP) **(15)**.

The process begins with **(11)** and **(12)**, where dedicated Automated Test/PenTesting modules actively probe nodes using automated and penetration testing. When a zero-day vulnerability or patching failure is detected, the results are sent to **(13)** the Data Manager. Here, the findings are integrated into the repository of known vulnerabilities, enriching the

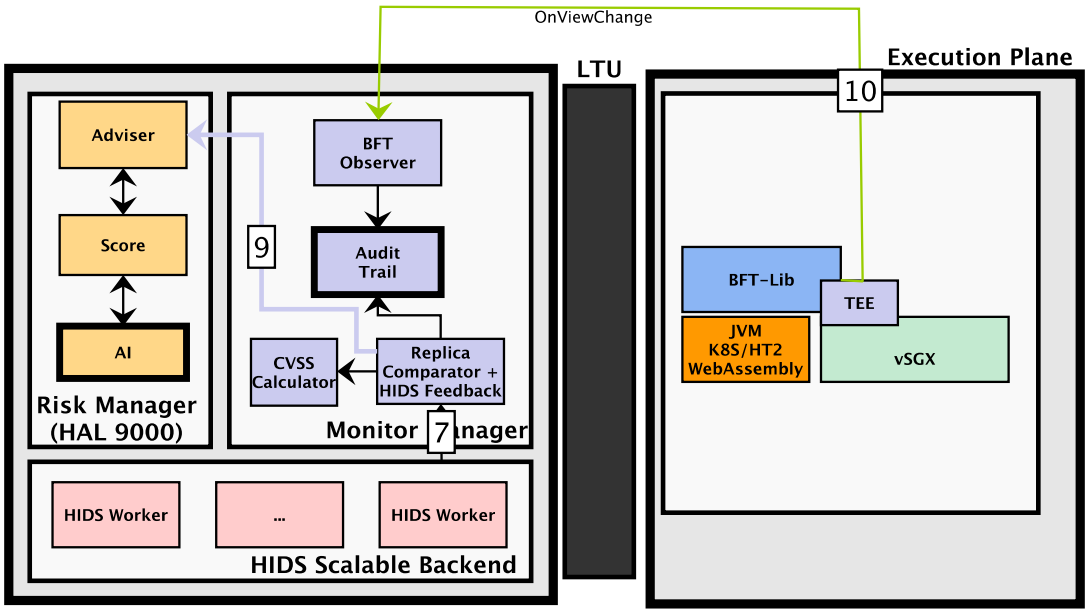

Figure 4.7: Observation process flow. (7) Telemetry and feedback are collected from the HIDS Workers; (10) anomalies and performance degradations in the BFT service are detected by the BFT Observer; (9) HAL evaluates the gathered information and advises on appropriate reconfiguration to maintain system resilience.

system's understanding of its security posture. Finally, the Data Manager **(15)** transmits the newly discovered vulnerability information to a collaborative threat intelligence platform, such as MISP, facilitating broader awareness and coordinated response within the ecosystem.

These dataflows collectively promote Skynet's adaptive and intrusion-tolerant oversight, demonstrating lines of information exchange, rapid threat detection, and coordinated recovery actions throughout the architectural landscape.

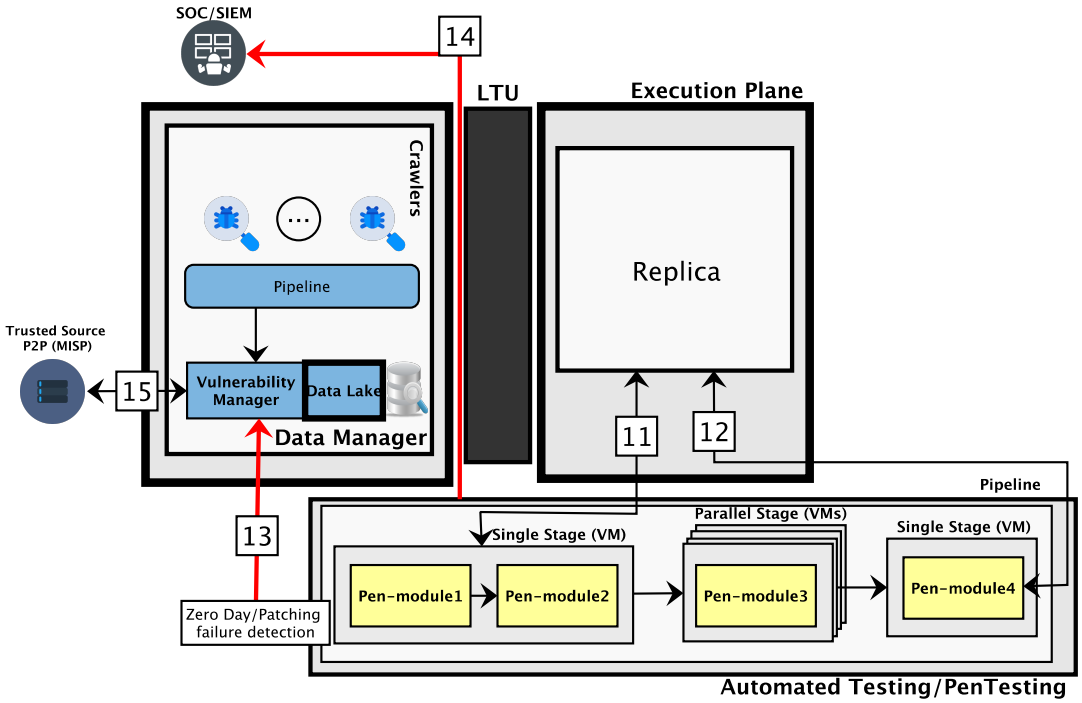

FIGURE 4.8: Testing process flow. (11) and (12) Automated Testing/PenTesting modules actively probe the nodes and identify vulnerabilities or patching failures; (13) the Data Manager aggregates findings and updates the vulnerability repository; (15) the Data Manager shares actionable intelligence with external threat sharing platforms such as MISP.

4.5. Integration Strategy

The Integration Strategy section delineates how the diverse modules and architectural elements of Skynet coalesce to form a unified, dependable system. While previous sections have laid out the individual components, their designs, and overarching dataflows, effective security and resilience depend on concrete integration mechanisms that ensure robust coordination and interoperability. This section details the architectural approach through which modules interconnect, the orchestration mechanisms that govern their collaboration, and the dependencies and interfaces that make seamless interaction possible. By clarifying these aspects, the Integration Strategy provides the blueprint for translating high-level design into a functioning, adaptable, and intrusion-tolerant system.

4.5.1. Module Interconnections

Skynet's architecture establishes module interconnections through secured and authenticated communication channels, with an enforcement of trust boundaries and defense, in-depth mechanisms designed for resilience, agility, and containment of compromise.

All traffic between the trusted Control Plane and the untrusted Execution Plane is strictly channeled through the LTU. The LTU acts as a cryptographically fortified gateway, mandating mutual attestation, public-key infrastructure, and protocol validation for every cross-plane interaction. This design explicitly prohibits any direct connections between planes, ensuring that the trust boundary is narrow, well-defined, and actively policed.

Within the Control Plane, modules such as the Data Manager, HAL, Monitoring Manager, Deploy Manager, and the HIDS Scalable Backend communicate via authenticated Application Programming Interfaces (APIs) and encrypted message buses, using mutual TLS over HTTPS and role-based access controls. This compartmentalization secures intra-plane traffic and enables comprehensive auditing.

The Data Manager aggregates and curates all internal telemetry and external threat intelligence, exposing its data via secure APIs. Modules like HAL regularly retrieve up-to-date threat and vulnerability data to guide risk assessment and adaptive reconfiguration. The Deploy Manager receives cryptographically signed configuration and patching directives, which are delivered to nodes in the Execution Plane using the LTU as the delivery orchestrator. All payloads are subject to authentication, encryption, and rate-limit protections to resist channel abuse or attack amplification.

Nodes in the Execution Plane report telemetry, including forensic logs, file integrity states, and intrusion alerts, using secured, purpose-specific agents. The LTU validates, batches, and forwards these reports upstream, ensuring that only authenticated and policy-compliant data enters the trusted environment. The HIDS Scalable Backend and Monitoring Manager then perform high-fidelity anomaly analysis and cross-correlation, with results fed back to configuration and orchestration modules.

Both control and execution modules are designed for resilience: buffered delivery, retry mechanisms, and fallback paths are employed in the event of communication disruption. The LTU actively supervises channel health, automatically revoking credentials or quarantining endpoints that exhibit anomalous or malicious behaviors. This containment strategy is key for maintaining the compositional trust model even when parts of the Execution Plane are under active attack.

These tightly enforced interconnections allow Skynet to achieve real-time situational awareness, rapid response, and robust adaptive defense, ensuring that all modules operate in cohesive synergy yet remain properly isolated in the event of compromise. Feedback loops, both automated and operator-assisted, ensure that new incidents or configuration

changes are quickly disseminated, responses coordinated, and exposures minimized, maintaining the integrity and trustworthiness of the entire platform, even under persistent, multi-stage attacks.

4.5.2. Orchestration and Coordination

Orchestration is done as a centralized, policy-driven process anchored in the Control Plane, which acts as both an intelligent coordinator and operational authority for the entire system. The Control Plane, besides relaying configurations or receiving alerts, actively drives the lifecycle, security adaptations, and integration of all subsystems following real-time risk assessments and defined resilience objectives.

At the core of the orchestration is the Risk Manager HAL, which synthesizes a diverse range of threat information collected by the Data Manager, including both external threat intelligence feeds and internally generated evidence from testing, monitoring, or incident analysis. Using these inputs, HAL determines optimal system configurations, assesses cross-domain risk, and directly issues adaptation instructions to the Deploy Manager, ensuring that software stacks and network settings remain robust and diversified in the face of evolving threats.

Orchestration is designed to be event-driven: upon detection of a vulnerability, state deviation, or suspicious behavior, the Control Plane pivots from routine operations to a targeted response mode. This may include redeployment of affected nodes, activation of additional monitoring or forensic routines, and prompt integration of threat intelligence into ongoing detection logic. All actions are executed to minimize vulnerability windows and maintain operational continuity.

Unlike isolated orchestration found in traditional setups, Skynet also extends its coordination to the inter-domain level. Multiple Skynet instances exchange vetted intelligence and configuration guidance, thus supporting federated defense, distributed mitigation, and contextual cross-validation of security strategies across larger infrastructures. This federative orchestration is underpinned by mutually authenticated protocols and a standardized schema for sharing actionable data, so collaborative defense becomes both scalable and effective across multiple domains.

Throughout, orchestration remains transparent but adaptive: policy changes, threat responses, and operational interventions are systematically logged and integrated with

external SOC/SIEM platforms, supporting both accountability and seamless operational integration with broader security ecosystems.

By enforcing tight coordination, internally and in federation with peer systems, Skynet's orchestration layer maximizes agility, resilience, and consistent enforcement of security posture, maintaining robust and adaptive defense even under adversarial pressure.

Orchestration policies and response triggers within Skynet are continuously reviewed and refined based on evolving threat intelligence, shifts in operational demands, and insights from ongoing system monitoring, ensuring that the platform's defensive posture remains agile and aligned with the latest security context.

4.5.3. Dependencies and Interfaces

Skynet's design establishes clear operational boundaries and modular interactions to facilitate secure, adaptable, and resilient integration between components. Each module depends on trusted system services such as authenticated communication channels, cryptographic verification of messages, and consistent protocols for data encoding and transmission. These shared underpinnings guarantee reliable collaboration and safeguard sensitive information as it flows throughout the architecture.

External connections are managed via integration points with industry-standard platforms, including feeds from recognized vulnerability repositories and compatibility with trusted hardware security mechanisms. Aligning with best practices, Skynet maintains connections to external security tools and monitoring suites to support comprehensive situational awareness and incident response.

To preserve system integrity, all module interactions are mediated by well-defined APIs, using secure transport mechanisms and strict access controls. Operational endpoints support essential tasks such as fetching event logs, reporting security alerts, and coordinating reconfiguration actions. These APIs are structured to facilitate updateable, future-proof integration, supporting ongoing system evolution and permitting the addition or replacement of components without compromising overall function.

Consolidating these principles, Skynet ensures its interfaces are both robust and extensible, enabling smooth collaboration between internal modules and external systems. This approach underpins the architecture's capacity to remain interoperable and responsive amid changing operational needs and emerging threats.

4.6. Chapter Summary

The chapter provides a comprehensive exposition of Skynet's architecture, an intrusion-tolerant overseer engineered to address the evolving landscape of multi-domain cyber threats. Emphasizing the growing inadequacy of siloed or perimeter-based security solutions, the narrative justifies the need for a disruptive approach that integrates intrusion tolerance, cybersecurity, and ML.

A defining aspect of the architecture is its separation of the trusted Control Plane from the untrusted Execution Plane. The Control Plane is responsible for orchestrating deployments, managing risk, monitoring system health, and adapting configurations in response to threat intelligence or anomalies. By contrast, the Execution Plane, purpose-built to run core workloads, is assumed to be at risk of compromise and is therefore equipped with layered defense mechanisms, including hardware-based isolation, forensic logging, and constant monitoring.

Each architectural module contributes distinct yet comprehensively interconnected capabilities. The Data Manager consolidates vulnerability intelligence and telemetry, supporting both risk assessment and collaborative threat sharing. HAL, the system's risk manager, leverages ML to advise on lower risk configurations and evaluate through prediction newly discovered vulnerabilities, ensuring that resilience and proactive adaptation are interlinked into the system's operational design. The Monitoring Manager extends system awareness through cross-correlation of detection signals and forensic evidence, providing an active safeguard against both conventional and Byzantine threats. The Deploy Manager oversees the lifecycle of nodes, integrating configuration recommendations and orchestrating updates to minimize exposure. A scalable backend supports HIDS at scale, aggregating analytics within the trusted environment to reveal systemic and coordinated attack campaigns.

The integration strategy described in the chapter sets out a communication and trust model in which all cross-plane interactions are funneled through a cryptographically fortified LTU. Authenticated APIs, persistent telemetry pipelines, and centralized policy-driven orchestration ensure that module interactions remain secure, resilient, and inherently auditable. Event-driven workflows and adaptive orchestration allow Skynet to respond swiftly to incidents or emerging vulnerabilities, whether they arise independently or as part of wider coordinated campaigns.

The architecture's design and rationale reflect a careful balancing act among security, resilience, operational agility, and performance overhead. Operational boundaries and clear assumptions about trust and threat models are maintained throughout. Dataflows between components are constructed to ensure timely and robust sharing of information, thereby supporting real-time decisions and minimizing windows of vulnerability.

Through this synthesis, Skynet emerges as a modular, adaptive, and open platform capable of both enduring and dynamically responding to advanced attacks. Its foundation is designed for interoperability and extendibility, supporting collaboration across domains and promoting broader ecosystem readiness. By embedding intrusion tolerance as a core operational principle, rather than as a late addition, the architecture establishes a resilient, future-ready framework poised to advance the state of security for critical distributed systems.

5. Proactive Risk Management in ITSs: Design and Evaluation of HAL

This chapter presents the design, implementation, and evaluation of HAL, a next-generation Risk Manager developed for ITSs. HAL builds on prior research by combining automation and ML-driven intelligence to provide continuous risk assessment and configuration recommendations. Its central objective is to identify and advise ITS configurations that minimize the risk exposure to malicious adversaries.

This work is guided by the hypothesis that integrating automated, ML-driven risk estimation and real-time configuration advisement into distributed ITS environments will significantly reduce risk exposure times and improve system responsiveness compared to conventional approaches. Specifically, this chapter addresses the question: "Does HAL's predictive assessment framework enable earlier and more adaptive risk mitigation in ITS environments?"

HAL is architected for deployment in large-scale, distributed ITS environments, where timely and coordinated risk management is especially critical.

Conventional risk management approaches in ITS environments primarily depend on publicly available threat intelligence sources, such as the NVD, ExploitDB, and other vulnerability databases. These systems typically assess risks through indicators like CVSS scores, which serve as the basis for prioritizing threats and determining lower-risk alternatives or system configurations.

While effective, these established approaches face limitations. Most notably, they require manual analysis and expert review before assigning risk scores to new vulnerabilities. This dependency often introduces significant delays, ranging from weeks to months, between the discovery of an exploit and the system's subsequent defensive response [93]. Such lag times undermine the responsiveness needed to address rapidly evolving threats and may leave systems unprotected during critical windows.

Moreover, traditional CVEs and their associated CVSS evaluations do not fully leverage the contextual or temporal information included in vulnerability disclosures. For example, they may fail to dynamically adjust scores in response to the release of a patch or the progression of exploit activity. To address these gaps, HAL incorporates mechanisms to

reassess CVSS scores using additional contextual details and recalculated metrics, drawing inspiration from recent advances in vulnerability scoring methodology.

A further contribution of this work is the introduction of the VExHK, an OSINT-based data aggregation module that automatically scrapes vulnerability and threat intelligence from public OSINT databases. VExHK consolidates this information into a centralized intelligence database, thereby supporting HAL with up-to-date insights.

In addition, HAL integrates a predictive CVSS scoring model to mitigate the "analysis bottleneck" that occurs when new vulnerabilities emerge but have yet to be scored manually. By employing this predictive capability, the system enables early-stage risk assessments and allows defensive measures to be deployed immediately, rather than waiting for official evaluations. In doing so, HAL provides a more agile and resilient framework for ITS security, one that adapts to uncertainty and reduces reliance on incomplete or delayed threat intelligence.

Finally, the chapter establishes a transparent and ML–enabled approach to risk management in ITS. By combining automated threat intelligence collection, predictive scoring, and adaptive configuration advisement, HAL seeks to overcome the limitations of traditional systems and contribute to the resilience of ITSs.

This module is evaluated by comparing HAL's advised risk scores and configurations to those of other state-of-the-art systems. The assessment covers resilience score (configuration effectiveness), security score (single node evaluation), accuracy of the CVSS predictor against real values, and analysis of optimal strategies for clustering descriptions of similar CVEs.

One significant challenge encountered in this work is the variable quality of vulnerability data, particularly the descriptive fields used for risk assessment and clustering. Ambiguously written vulnerability descriptions can hinder automated analysis and affect the accuracy of both clustering algorithms and predictive scoring models. Addressing these data quality issues remains a critical topic for future improvement and is discussed in subsequent sections.

The remainder of the chapter is structured as follows. Section 5.1 reviews the state-of-the-art on existing Risk Managers, CVSS score predictors, Text Document Clustering techniques, and OSINT scrapers. Section 5.2 outlines HAL's system design, including the VExHK module, its workflow, and modular architecture. Section 5.3 details the experimental setup and validation procedures. Section 5.3 analyzes the outcomes of these experiments,

and Section 5.5 summarizes the key findings of the chapter.

5.1. Review of the State of the Art

Following the introduction, this section addresses the current State of the Art in Risk Management for ITSs. It examines recent innovations in automated CVSS score prediction using ML, AI, and NLP, advances in textual clustering techniques that improve vulnerability similarity analysis, and methods for integrating open-source intelligence collection to enhance proactive defense. These advances collectively establish the development of adaptive and autonomous Risk Manager solutions such as HAL, which address the challenges posed by traditional approaches reliant on the dependency of manual vulnerability evaluations.

5.1.1. Risk Managers

Early ITS risk management approaches primarily focused on leveraging diversity strategies combined with vulnerability scoring to mitigate exposure to cyberattacks [27, 94].

A notable example is the Diversity Policy for Intrusion Tolerant Systems by Heo et al. [27], which systematically evaluates known software vulnerabilities and recommends configurations that minimize the overall risk. This approach generates all possible system configurations and selects those with the lowest aggregate risk based on the sum of CVSS scores.

The underlying architecture supporting such policies typically comprises groups of VMs categorized into cleansing and processing sets, coordinated by a central controller. Cleansing aims to clean compromised VMs through rejuvenation cycles, after which the VMs transition into the processing group to serve external requests. The controller orchestrates periodic rotation and reconfiguration of VMs to continuously refresh the system state and thwart persistent intrusions. However, the effectiveness of this strategy is fundamentally limited by its dependency on the NVD, which assigns CVSS scores only after thorough and time-consuming manual evaluation [23]. Consequently, the latency between vulnerability disclosure and formal scoring introduces a critical window of heightened risk during which ITSs cannot proactively adapt configurations to counter new threats.

Building upon these foundations, Lazarus introduced a more adaptive, BFT oriented risk management architecture that advances beyond sole reliance on static CVSS data [25].

Lazarus incorporates a dynamic scoring methodology that refines vulnerability severity assessments based on additional contextual factors such as vulnerability age, patch availability, and known exploit activity. Mathematically, for a given vulnerability v—with publication date $v.published\_date$, a patch-availability flag $v.patched \in \{0, 1\}$, and an exploitation flag $v.exploited \in \{0, 1\}$—Lazarus computes a recalculated risk score:

$$\text{score}(v) = \text{CVSS}(v) \times \text{oldness}(v) \times \text{exploited}(v) \times \text{patched}(v) \tag{5.1}$$

$$\text{oldness}(v) = \max\left(1 - 0.25 \times \frac{(now - v.published\_date)}{oldness\_threshold}, 0.75\right) \tag{5.2}$$

Where $oldness\_threshold$ is set to 365 days in the experiments; now and $v.published\_date$.

$$\text{patched}(v) = 0.5^{v.patched} \tag{5.3}$$

$$\text{exploited}(v) = 1.25^{v.exploited} \tag{5.4}$$

Aging effects decrease or increase the impact of vulnerabilities over time, while the patched and exploited parameters adjust scores to reflect real-world mitigations and threats [25]. These calculations are presented by equations 5.1 to 5.4.

Furthermore, Lazarus employs unsupervised ML, notably K-means clustering on vulnerability descriptions, to detect shared vulnerabilities across disparate OSs, revealing systemic risks overlooked by platform-specific assessments. These clusters inform the risk scoring and yield configuration recommendations that prioritize resilience by minimizing correlated vulnerabilities among replicas.

Nevertheless, Lazarus's approach incorporates inherent trade-offs. The architectural model strives to ensure that no two replicas simultaneously share unpatched vulnerabilities, preserving the BFT protocol invariant of $3f + 1$ [25]. While this enhances resilience to parallel exploitation, it may at times sacrifice absolute security by tolerating configurations with lower aggregate risk scores but potentially higher individual vulnerability risks. Additionally, the manual intervention required for optimal cluster number selection in K-means introduces delays, and the clustering method itself is sensitive to outliers, which can compromise the quality of vulnerability groupings [95, 96].

These limitations have motivated a shift toward incorporating advanced ML and automated data integration techniques within risk managers. Emerging frameworks aim to

overcome reliance on delayed, human-curated vulnerability ratings by employing NLP to predict CVSS scores or metrics directly from vulnerability descriptions [28, 97, 98]. These ML- or AI-based predictors enable near-real-time risk assessment of newly disclosed vulnerabilities, thus narrowing critical exposure windows. Concurrently, more sophisticated clustering algorithms such as OPTICS and HDBSCAN have shown promise in handling noisy and high-dimensional textual data, offering enhanced robustness against outliers and improving semantic vulnerability grouping [99, 100].

Complementing these algorithmic advances, the integration of automated OSINT scraping tools broadens the threat intelligence ecosystem, feeding risk managers [101]. By continuously harvesting vulnerability data and exploit evidence from multiple sources, including but not limited to the NVD, ExploitDB, AlienVault OTX, and Open Source Vulnerabilities (OSV), modern risk managers dynamically enrich their knowledge base. This continuous intelligence allows ITSs to preemptively adjust configurations in response to emergent, rapidly evolving threats, including quasi-zero-day exploits that historically evade timely formal analysis.

5.1.2. Machine Learning for CVSS Score Prediction

As cybersecurity threats evolve rapidly, traditional manual processes to evaluate vulnerabilities often struggle to keep pace with the influx of newly disclosed issues. In response, researchers have increasingly turned to AI and Deep Learning (DL) techniques to automate the scoring of software vulnerabilities based on their textual descriptions. These approaches aim to expedite vulnerability assessment, reduce human error, and provide timely guidance for risk mitigation.

Khazaei et al. [28] pioneered the use of ML for predicting CVSS scores directly from vulnerability descriptions. Their methodology involved extracting textual features through standard preprocessing pipelines, then applying ML algorithms such as Support Vector Machines, Random Forests, and fuzzy logic systems to train predictive models. Among these, fuzzy logic-based models achieved the highest accuracy, correctly classifying approximately 88% of evaluated vulnerabilities. This work demonstrated the feasibility of leveraging automated techniques to streamline vulnerability severity estimation.

Building on this foundation, Sahin et al. [97] extended prior efforts by incorporating advanced DL architectures, including Convolutional Neural Networks and Long short-term memory networks, to predict not only discrete severity categories but also continuous

CVSS scores. Their approach utilized word embeddings trained with the word2vec framework [102] to represent vulnerability descriptions as dense vectors, improving semantic understanding. Complemented by gradient boosting methods, their models achieved a mean error rate of about 16%, showcasing the potential of DL for more nuanced vulnerability assessment.

Elbaz et al. [103] proposed a complementary strategy focused on predicting individual CVSS vector metrics rather than aggregate scores. Using linear regression models trained on bag-of-words features extracted from vulnerability descriptions, their framework aimed to enhance the interpretability of automated scoring by decomposing assessments into constituent components. Though achieving slightly lower accuracy compared to some holistic predictors, this approach provides valuable explainability, enabling security experts to better understand underlying vulnerability characteristics.

More recently, researchers have employed transformer-based language models to further improve prediction quality. Costa et al. [98] explored the use of state-of-the-art NLP architectures such as DistilBERT, DeBERTa, and ALBERT, combined with vocabulary augmentation and preprocessing techniques like lemmatization and stemming. Experimental results favored DistilBERT, particularly when paired with a vocabulary expanded by 5,000 domain-specific terms, outperforming other models in accuracy. Nevertheless, the study focused primarily on providing interpretable insights to analysts rather than optimizing raw score predictions.

Following this tendency, Kai et al. [104] developed VultDistilBERT, a refined transformer model tailored for vulnerability severity assessment with a focus on data imbalance and sample augmentation. By strategically augmenting textual data via synonym replacement and deletion, and selecting optimal subsets of CVSS metrics as additional input, the model enhanced representation learning. Coupled with a linear classifier, VultDistilBERT demonstrated state-of-the-art performance with an accuracy exceeding 97%, marking a significant advance in automated vulnerability scoring.

5.1.3. Clustering for Vulnerability Analysis

Text document clustering is a method employed in text mining and information retrieval to organize large collections of textual data. Initially, it was explored as a strategy to improve precision and recall in retrieval systems [105, 106], facilitate the identification of nearest

neighbor documents [107], support browsing within document corpora [108], and aid in structuring search engine results [109].

This technique operates in an unsupervised learning setting, where documents are grouped into clusters based on inherent similarities without prior labeling. Notably, as pointed out by Lazarus [25], vulnerability databases can contain multiple entries with different identifiers affecting diverse software but sharing nearly identical descriptions, effectively representing the same security issue. Employing text clustering on such datasets can reduce redundancies and improve data quality by consolidating similar vulnerability reports. Several studies have evaluated various clustering techniques relevant to document clustering.

Steinbach et al. [110] conducted a comparative analysis between hierarchical agglomerative clustering methods and K-means variations, including bisecting K-means and implementations of the Unweighted Pair Group Method with Arithmetic Mean (UPGMA). Their assessment used internal metrics such as overall similarity, measuring cluster cohesion, and external metrics like entropy, which evaluates cluster purity. They found bisecting K-means particularly effective due to its balance of computational efficiency and clustering quality, making it well-suited for large-scale document collections.

Zhao et al. [111] compared agglomerative clustering algorithms with partitional clustering methods, specifically evaluating group average functions against k-way clustering. They applied the F-score, a metric introduced by Larsen et al. [112], alongside entropy measures to assess clustering quality. Their findings suggested that k-way (partitional) clustering methods generally outperform agglomerative approaches. Lower entropy values indicated more coherent clusters, and higher F-score reflected robustness against outliers.

Singh et al. [113] explored the comparative performance of K-means, Heuristic K-means, and Fuzzy C-means algorithms for clustering documents. They investigated different preprocessing strategies involving stop-word removal and stemming, as well as diverse feature representations such as term frequency, TF-IDF, and Boolean presence. Their results highlighted that while K-means is sensitive to initial centroid selection and Heuristic K-means introduces additional computational cost, Fuzzy C-means provides a more flexible clustering by assigning membership degrees rather than hard cluster assignments. Evaluation using residual sum of squares and cluster purity indicated Heuristic K-means outperforms standard K-means, though Fuzzy C-means offers robustness in handling ambiguity.

Mendonça et al. [100] examined the effectiveness of classical clustering algorithms applied to free-text data leveraging word embedding representations. The algorithms analyzed included K-means [114], Expectation-Maximization (EM) clustering with Gaussian Mixture Models (GMM) [115], Spectral Clustering [116], Mean Shift [117], and Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [118]. Using metrics such as Normalized Mutual Information and Homogeneity, they demonstrated parameter choice heavily influences clustering outcomes, with K-means emerging as the top performer under certain configurations.

More recent work by Asyaky et al. [99] focused on advancing density-based clustering algorithms, such as DBSCAN [118] and HDBSCAN [119], specifically for short text documents. By applying preprocessing techniques including lemmatization, stemming, and dimensionality reduction through document embeddings, they significantly enhanced clustering accuracy. Performance evaluations employed the Adjusted Rand Index and Adjusted Mutual Information [120], with their approach surpassing established state-of-the-art methods in this domain.

5.1.4. Automated Threat Intelligence Collection

The vast amount of OSINT data relevant to cybersecurity, encompassing vulnerabilities, exploits, and related threat information, requires automated mechanisms for continual and effective data collection. Relying solely on manual input is not only labor-intensive but may also result in incomplete coverage of critical information. To mitigate these challenges, web scraping techniques are commonly employed. Unlike web crawlers that continuously explore the Internet to locate new sites and new data, web scrapers specifically target predefined websites to extract targeted information efficiently. This automated process enables rapid acquisition of vital cybersecurity data such as Indicators of Compromises (IOCs), exploit reports, and recently disclosed vulnerabilities. The following presents an overview of OSINT scraping solutions currently established in the field.

Fernandes et al. [121] developed ScrapeIOC, a specialized web scraper focused on harvesting publicly accessible IOCs. Its primary purpose is to amass a collection of IOCs from multiple online platforms, concentrating on particular malware types and families. The collected data primarily consists of hashed samples (e.g., MD5, SHA1, SHA256), organized in an offline database according to their provenance. This repository is then integrated with publicly available sources to facilitate efficient malware hash searches. ScrapeIOC

interfaces with several platforms, including Malwares, Malshare, VxCube, ThreatCrowd, and Maltiverse. Although ScrapeIOC effectively consolidates malware hashes, its scope is limited compared to more expansive frameworks such as HAL, which capture a broader spectrum of cybersecurity intelligence, including exploits and Proof-of-Concept.

Alves et al. [122] introduced SYNAPSE, a platform designed to extract cybersecurity-related intelligence from selected Twitter* accounts maintained by individuals, research groups, and security organizations. The system aims to continuously monitor emerging threats by synthesizing and summarizing pertinent intelligence tailored for security analysts. Its pipeline incorporates stages for filtering, feature extraction, binary analysis, clustering, and IOC generation. Data acquisition is performed via scraping through Twitter's accessible API, targeting accounts deemed likely to share security-specific information. However, this focus restricts the tool's data sources to Twitter alone, thereby limiting its coverage relative to systems that aggregate intelligence from multiple platforms. Additionally, the monetization of X's API, with the discontinuation of most free access tiers and the imposition of relatively high usage fees, has rendered it increasingly difficult to utilize as an open-source solution. Many features and data access capabilities previously available without charge are now restricted to paid plans, limiting accessibility for researchers and developers relying on freely available APIs to gather public data.

Kühn et al. [123] proposed a classification framework assessing the suitability and crawlability of textual references within the NVD. Their work aimed at enhancing CVSS vector prediction by leveraging deep learning models trained on vulnerability-related texts. To support this, they engineered a tailored web scraper capable of retrieving vulnerability descriptions and related documents from external websites cited in the NVD. These external sources were categorized into groups such as version control repositories, bug trackers, mailing lists, patch notes, security advisories, third-party analyses, and blogs or social media feeds. Given the lack of standardized APIs across these sources, the scraper was tailored for targeted extraction. However, its design was confined to websites referenced within the NVD and did not extend to other publicly available intelligence repositories.

5.2. HAL Risk Manager: Architecture and Workflow

Section 5.2 presents a comprehensive exploration of the HAL Risk Manager's architecture and workflow, detailing the system's design principles and operational mechanisms that enable effective risk management in ITSs. This section is organized into subsections that

*Twitter has been officially rebranded as X since 2023.

systematically analyze the core components, starting with a high-level system overview and a formal system model that outlines the structural and functional aspects of HAL. It then addresses the threat model to characterize the adversarial environment and potential attacks the system must withstand. The section further introduces VExHK, the custom OSINT scraper integral to HAL's intelligence gathering capabilities. Subsequently, the workflow steps are described, illustrating how HAL processes vulnerability data to provide proactive and resilient configuration recommendations. Finally, the discussion on supporting modular intelligence highlights the system's flexibility in integrating diverse machine learning models and data sources, ensuring adaptability to evolving cybersecurity challenges.

5.2.1. System Overview

The architecture, illustrated in Figure 5.1, is composed of four primary components: the Vulnerability Score Predictor, the Clustering Module, the Score Reassessment, and the Configurator.

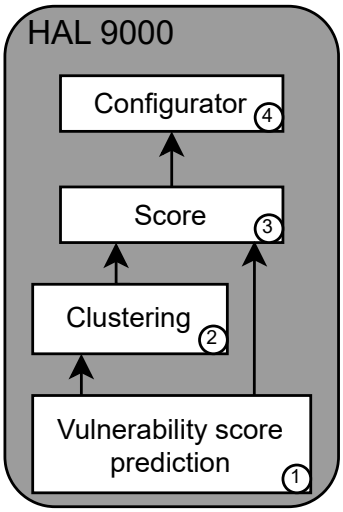

FIGURE 5.1: HAL Architecture and workflow.

Vulnerability Score Predictor: The first component is designed to estimate the severity of newly reported CVEs by predicting their CVSS base scores. Its implementation adopts a modular design, allowing different models to be interchanged as prediction engines. Depending on the selected algorithm, the model can be trained using only the textual description of a CVE, or with description and the associated CVSS metrics. After training, the model predicts the severity of previously unseen CVEs by processing their description. If the chosen algorithm directly predicts the CVSS base score, this value is used as output.

Alternatively, if the algorithm predicts individual metric values, these are subsequently combined using the official NVD equations [124] to compute the final base score.

Algorithm 1: CVSS Prediction Component Workflow

Input: Newly disclosed CVE with description d

Output: Predicted CVSS base score s

Training Phase:

Select prediction model M (trained on historical CVEs);

if *algorithm uses description only* **then**

  Train M on d;

else if *algorithm also uses metrics as well* **then**

  Train M on (d, m);

Inference Phase:

Input new CVE description d;

if M *predicts CVSS base score directly* **then**

  $s \leftarrow M(d)$;

else

  $\hat{m} \leftarrow M(d)$; `// Predicted CVSS metrics`

  $s \leftarrow$ `NVD_Calculator(`$\hat{m}$`)` ; `// Compute base score`

return s;

In its original implementation, HAL employed the algorithm introduced by Khazaei et al. [28] to perform this prediction task. Nevertheless, the system was deliberately designed with a modular interface, enabling seamless replacement of the underlying algorithm. To enforce modularity, the interaction between components was minimized: the predictor module only receives a training dataset at initialization and later accepts unseen CVEs description for inference. Both training and inference datasets are structured as CSV files that include the CVE identifier, textual description, and CVSS attributes.

Clustering Module: The second component addresses redundancy in vulnerability databases (first addressed in the work of Lazarus [25]), where different CVEs may affect distinct products but describe conceptually similar security flaws. To detect these relationships, HAL integrates an ML-based clustering algorithm that groups CVEs according to the semantic similarity of their descriptions.

The clustering procedure produces collections of related vulnerabilities, which allows subsequent steps to reason about risk across similar attack vectors. To maintain robustness,

the selected ML algorithm should be capable of automated operation, minimizing human intervention and reducing potential bias during the execution workflow. Additionally, the algorithm must remain resilient to outliers to prevent poor clustering performance when handling new or noisy data.

Figure 5.2 depicts an example of the clustering process, where the Elbow method is traditionally used to infer the number of clusters. Algorithms known for strong performance in handling outliers, such as HDBSCAN [119] or OPTICS [125]—can be employed, though the architecture is not restricted to them. Similar to the predictor, the clustering module follows a modular design, supporting extensions with supervised, semi-supervised, or DL approaches.

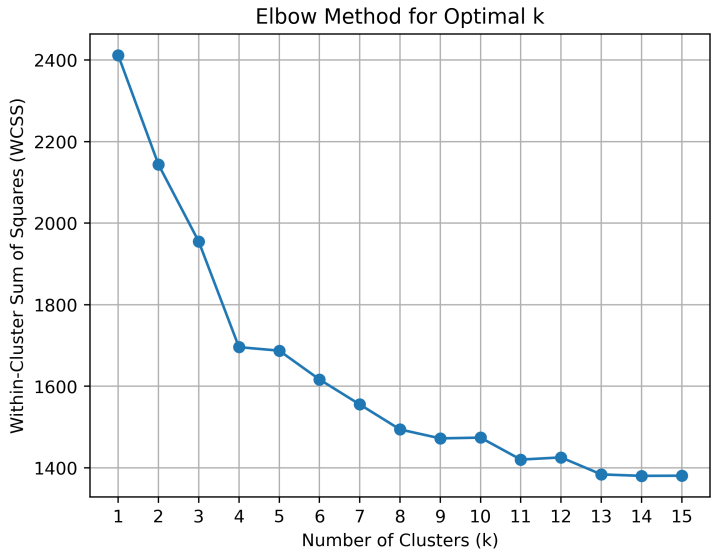

FIGURE 5.2: Example of a graph where visually extracting k can be challenging, to cluster the CVE descriptions by their similarity. Here k denotes the number of clusters, distinct from the "K" in K-means. According to the Pelt [126] method, the best k is 12. Adapted from [101].

Score Reassessment Engine: The third component improves the scoring methodology proposed by Lazarus. Lazarus's model refines CVSS values by incorporating additional vulnerability attributes, such as age, patch availability, and evidence of exploitation *. However, the original formulation assumes that once a patch is made public, exploitation risk diminishes, even though real-world incidents (e.g., ProxyShell and Log4Shell) demonstrate otherwise, as attackers frequently exploit CVE long after patches exist [127], i.e., the availability does not directly translate into its immediate deployment.

*A vulnerability is considered exploited when there are reports of its occurrence or when code is provided to exploit the vulnerability.

HAL extends this approach by integrating the EPSS [26], which estimates the probability that a vulnerability will be exploited in the following 30 days, and by explicitly considering patch adoption uncertainty. Formally, HAL recalculates the adjusted score as shown in equation 5.6, where both patched and unpatched scenarios are weighted by exploitation likelihood.

To further refine assessment, the current work also incorporates curated threat intelligence from AlienVault Open Threat eXchange (OTX) Pulses, which provide collective evidence of CVE exploitation in the wild across multiple campaigns. The extended formulation penalizes vulnerabilities present in numerous OTX pulses, signaling higher exploitation potential. This refinement is summarized in equation 5.7, where the log-transformed count of related pulses contributes to the adjusted risk score.

$$\text{score}_{np}(v) = \text{CVSS}(v) \times \text{oldness}(v) \times \text{exploited}(v) \tag{5.5}$$

$$\text{hal\_score}(v) = \text{score}(v) \times (1 - \text{EPSS}(v)) + \text{score\_}np(v) \times \text{EPSS}(v) \tag{5.6}$$

where $\text{score}_{np}(v)$ denotes the score computed without the patch adjustment (equivalently, with $\text{patched}(v) = 1$), representing the worst case in which an available patch has not been applied.

As an example, CVE-2017-11882 originally carries a CVSS score of 7.8 (classified as High). Using Lazarus's method, it would be downgraded to 3.65 (Low) due to patch availability, which fails to capture the true threat level. HAL's earlier methodology, enhanced with EPSS, reassessed the score to 7.2 (High) by accounting for delayed patch adoption and exploitation probability (97.99%). With the newly proposed pulse-based extension, the reassessment raises the score to 8.9 (borderline Critical), correctly capturing its persistent and widespread exploitation risk.

Equation 5.1 together with Equations 5.6–5.7 formalize the progression from Lazarus's scoring adjustments to the proposed HAL extensions:

$$\text{hal\_score}(v) = \min\left(10, \text{score}(v) \times \big(1 - \text{EPSS}(v)\big) + \text{score}_{np}(v) \times \text{EPSS}(v) + \log(\#\text{related\_pulses})\right) \tag{5.7}$$

The configurator evaluates a candidate configuration $config$, composed of nodes n_i, where $V(n_i)$ is the set of vulnerabilities present on node n_i and $V(n_i, n_j)$ is the set of

vulnerabilities shared between nodes n_i and n_j. The security and resilience risks of a configuration are then defined as:

$$\text{security\_risk}(config) = \sum_{n_i \in config} \sum_{v \in V(n_i)} \text{hal\_score}(v) \tag{5.8}$$

$$\text{resilience\_risk}(config) = \sum_{n_i, n_j \in config} \sum_{v \in V(n_i, n_j)} \text{hal\_score}(v) \tag{5.9}$$

Configurator: The final component consolidates the outputs from the previous modules to recommend secure and resilient ITS configurations. Its purpose is to minimize two risk functions:

- *security_risk(config)* – representing the cumulative impact of all vulnerabilities across nodes in the configuration, assessed using the predicted and reassessed CVSS values.
- *resilience_risk(config)* – quantifying systemic risk by identifying shared or semantically clustered vulnerabilities across pairs of nodes, thereby capturing "break one, break all" attack scenarios.

5.2.2. System Model

The HAL Risk Manager is deployed within an ITS environment designed to ensure both availability and security in the presence of successful intrusions. The ITS relies on replicated service nodes distributed across heterogeneous platforms and OSs, aiming to reduce the likelihood of shared vulnerabilities. These replicas are periodically rejuvenated, diversified, or replaced through techniques such as OS diversity, replica rejuvenation, and N-version programming, thereby limiting fault propagation and sustaining system liveness.

HAL is assumed to operate exclusively within the control plane of an ITS. This plane is assumed to be isolated and protected, providing a secure execution environment shielded from direct exposure to adversaries. The control plane hosts HAL and ensures trusted access to the monitoring, scoring, and configuration processes that guide the deployment of the replicas in the execution plane.

Conceptually, HAL is structured into four modules, namely, the Vulnerability Score Predictor, Clustering Module, Score Reassessment Engine, and Configurator, whose functionality was detailed in the system overview. In the context of the system model, these modules are treated as abstract functional units that collectively enable risk management within the ITS framework. Their role is to process vulnerability intelligence, reassess risk,

and recommend resilient configurations, while remaining confined to the protected control plane of the architecture.

HAL functions as a continuously informed risk manager by interfacing with external vulnerability and exploit intelligence sources, including the NVD, ExploitDB, AlienVault OTX, and the OSV project. An automated crawler periodically fetches, normalizes, and stores this data in a local repository, ensuring a current risk knowledge base without relying on real-time external queries.

5.2.3. Threat Model

The threat model for HAL considers adversaries who seek to undermine the Risk Manager or interfere with its decision-making. While the control plane hosting HAL is assumed to be isolated from direct exposure, no system is entirely immune to compromise, and sufficiently motivated attackers could attempt to target it. The design therefore does not claim absolute protection, but rather operates under the assumption that such attacks fall outside the adversaries' cost–benefit threshold, where the effort required outweighs the likely reward.

From an adversarial perspective, the most valuable avenues of attack against HAL would include attempts to gain unauthorized access to the control plane, tamper with risk assessments, or disrupt the integrity of configuration recommendations. A direct compromise of HAL's execution environment would allow an attacker to distort vulnerability models and mislead the ITS configuration process, potentially weakening resilience. However, it is assumed that the control plane employs sufficiently strong isolation, hardening, and auditing mechanisms such that the difficulty of compromising it remains prohibitively high compared to attacking the exposed execution plane replicas.

Instead, adversaries are expected to operate within the execution plane, where service replicas run on diverse platforms and are inherently exposed to public networks and known vulnerabilities. HAL's effectiveness stems from the fact that undermining the replicas is a lower-cost and higher-reward strategy for attackers than directly attempting to subvert the control plane or interfere with its internal data flows. Nevertheless, indirect attempts to mislead HAL, such as exploiting delays or omissions in public vulnerability feeds, are considered within the realm of possibility, although the system assumes that intentional large-scale poisoning of trusted OSINT providers is unlikely.

In summary, HAL's threat model assumes that while the control plane could, in principle, be a target, its isolation and protection render it an unattractive objective when weighed against easier opportunities in the execution plane. Thus, the primary expectation is that adversaries exploit vulnerabilities in exposed replicas, while direct attacks against the Risk Manager or its control-plane environment are treated as out of scope due to their unfavorable risk–reward balance.

5.2.4. Integration with the VExHK Module

A significant enhancement introduced in the extended HAL framework is the VExHK module [101], which serves as a collector of OSINT feeds. Rather than querying external sources during runtime, VExHK maintains a continuously updated local repository of vulnerabilities and exploitation data. This design minimizes execution overhead for HAL while ensuring that predictions and risk assessments are always based on the most recent available intelligence.

The module operates as a scheduled collector that retrieves data from diverse sources, including the NVD, ExploitDB, AlienVault OTX, and the OSV project. For sources that expose APIs (e.g., NVD and OTX), VExHK issues rate-limited requests authenticated via provider keys, whereas for others with scraping restrictions (such as ExploitDB), supported tools like Searchsploit are wrapped and automated. Collected entries are normalized into a common data model and stored in a local key–value database, allowing HAL to query them efficiently in subsequent processing steps.

By decoupling online retrieval from risk assessment execution, VExHK increases both reliability and scalability. Rate limiting and unavailability of external sources no longer directly impact HAL since new intelligence is asynchronously incorporated into the local repository. Moreover, the module preprocesses the raw records to remove redundant fields and standardize identifiers, thus improving input quality for ML prediction, clustering, and score reassessment.

The VExHK module acts as HAL's intelligence backbone: it filters, consolidates, and delivers curated vulnerability information in near real time. This integration ensures that HAL can continuously reassess risks using timely and comprehensive threat data, while remaining shielded from the inconsistencies or delays of external OSINT providers.

5.2.5. End-to-End Workflow with VExHK Integration

With the integration of the VExHK module, HAL's workflow expands from a purely predictive and evaluative loop into a continuous intelligence-driven pipeline. The process begins with the collection of vulnerability and exploit data from multiple open-source intelligence feeds. VExHK continuously aggregates, normalizes, and stores the information locally. This design ensures that HAL operates on an always-available, up-to-date knowledge base while remaining isolated from API rate limits, missing records, or inconsistencies that may affect the source repositories.

Once new or previously unscored vulnerabilities are detected, the system triggers its predictive component, which uses trained ML models (currently based on Random Forests) to assign temporary CVSS scores. These predictions are immediately informed by the contextual enrichment provided by VExHK, which consolidates cross-database feeds. The collected intelligence streamlines the prediction phase and ensures that early scoring captures both technical and situational threat factors from the outset.

The resulting knowledge base is then processed by the clustering module, which leverages sentence embeddings in combination with non-parametric algorithms such as OPTICS to identify families of semantically related vulnerabilities. Through clustering, HAL surfaces common weaknesses that may affect heterogeneous software and OSs across nodes, thereby uncovering latent "break one, break all" risks that would remain hidden if CVEs were considered in isolation.

Following prediction and clustering, HAL enters the reassessment stage. Here, provisional scores are dynamically adapted by accounting for vulnerability age, patch availability, and evidence of exploitation in the field. The EPSS contributes a probabilistic likelihood of near-term exploitation, while VExHK supplements this with multi-source corroboration and curated context, such as exploit Proof-of-Concept sightings and pulse frequency across threat intelligence platforms. This recalibration results in more realistic scores, particularly in scenarios where vulnerabilities remain highly exploitable long after patches are technically available.

Finally, HAL aggregates the reassessed vulnerability information to evaluate all candidate ITS configurations. The Configurator component jointly analyzes two key risk perspectives: the cumulative *security risk*, reflecting the overall vulnerability exposure of each node, and the *resilience risk*, which highlights systemic weaknesses shared between nodes or

across clustered vulnerabilities. Each configuration is scored with respect to both metrics, enabling HAL to recommend the deployment that best balances security and resilience.

This end-to-end workflow, presented in Figure 5.3, from intelligence collection to configuration advisement, widens HAL's scope from a predictive scoring engine into a comprehensive risk management pipeline. By embedding VExHK as its intelligence backbone, HAL combines timely vulnerability detection and risk reassessment into a decision-support system capable of advising secure and resilient ITS deployments under evolving and adversarial conditions.

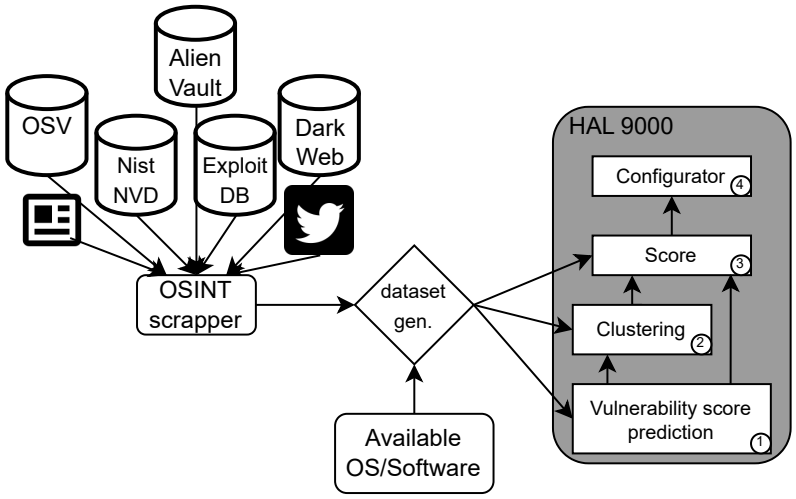

FIGURE 5.3: HAL Architecture and workflow, along with the integration of the OSINT scraper. Within HAL's architecture, the data execution path is enumerated from one to four to showcase the execution flow. (The Twitter logo is now officially rebranded as X since 2023.) Adapted from [101].

5.3. Experimental Design and Results

The experimental evaluation of HAL was conducted to substantiate the effectiveness of the extended framework and to determine its practicality for deployment in real ITS environments.

To verify these expectations, a series of controlled experiments was designed to cover complementary aspects of HAL. First, the accuracy of CVSS score prediction for previously unrated vulnerabilities was assessed using ML and AI models. Second, the robustness of the clustering process was examined by comparing different data preprocessing techniques and clustering algorithms, with the goal of reliably grouping semantically related CVE. Third, a comparative experiment was conducted to evaluate security and resilience assessments across all risk manager implementations, namely Heo, Lazarus, and HAL. This stage

focused on analyzing how each approach balanced configuration security and fault-tolerant resilience under the same dataset conditions. Fourth, the performance and capability of VExHK, the automated OSINT scraper, were measured. This experiment assessed execution time, resource overhead, and data acquisition limitations.

The results presented in this section highlight the comparative performance of HAL against prior approaches such as Lazarus and Heo. Beyond confirming improvements in prediction and clustering accuracy, the evaluation demonstrates HAL's ability to generate secure and resilient system configurations while reducing dependence on delayed external scoring processes. The remainder of this section details the dataset preparation, and presents the obtained results.

5.3.1. Dataset Preparation

A dataset containing information on CVEs, and exploits related to OSs and software was required for the experiments. However, no public dataset provided complete coverage with the desired characteristics, namely a consistent mapping between OS families, their installed software, and associated vulnerabilities.

To construct the dataset, a predefined list of OSs and software packages was first assembled, representing deployment environments for ITSs. This list was then used to query the database managed by VExHK, ensuring that all relevant vulnerabilities were retrieved for each OS or software entry.

The retrieved entries included fields such as the CVE identifier, publication date, last modification date, description, exploitability, and CVSS metrics (v2 and, where available, v3.0 or v3.1) [128]. By restricting the dataset to vulnerabilities recognized in NVD and ExploitDB, comparability with prior risk managers, specifically Lazarus and Heo, was maintained, while also ensuring a clean baseline to evaluate HAL's extended workflow.

The resulting dataset included more than 38,000 vulnerabilities spanning multiple OS families and versions, including Debian, Ubuntu, Windows Server, Solaris, BSD, Fedora, CentOS, and OpenSUSE. Table 5.1 and Table 5.2 list the considered OSs and the respective number of associated CVEs included in the dataset. This breadth of coverage enables systematic evaluation of security and resilience trade-offs across heterogeneous system configurations while remaining grounded on real-world data sources.

Table 5.1: List of considered OSs and respective amount of CVEs (Part 1). Adapted from [129].

OS	#CVEs
Debian 7	3,923
Windows 10	1,514
FreeBSD 11	221
Debian 8	3,923
Ubuntu 16.04	2,035
Solaris 10	359
OpenBSD 6.0	69
Centos 7	8
Fedora 30	835
Solaris 11	359
Ubuntu 14.04	2,035
Ubuntu 16.04	2,035

Table 5.2: List of considered OSs and respective amount of CVEs (Part 2). Adapted from [129].

OS	#CVEs
Debian 6	3,923
Ubuntu 17.04	2,035
Fedora 16	835
Ubuntu 12.04	2,035
Ubuntu 22.04	2,035
Debian 10	3,923
Ubuntu 10.04	2,035
Fedora 38	835
Windows Server 2012	1,105
Fedora 24	835
Centos 8	8
OpenSuse 42.1	1,298

5.3.2. CVSS score prediction experiment

This experiment evaluated the capability of state-of-the-art ML and DL approaches to predict CVSS scores for vulnerabilities using the constructed dataset. Three representative methods from the literature were selected and implemented for comparison. Each approach employed a distinct prediction strategy as described in Section 5.1.

The first method, proposed by Khazaei et al. [28], applies traditional ML algorithms trained with CVE descriptions and previously assigned CVSS scores to directly predict a vulnerability's score. The second method, introduced by Costa et al. [98], uses DL techniques combined with NLP to predict the individual CVSS vector metrics from vulnerability descriptions; these are subsequently used to compute the final CVSS score according to the v3.1 specification. The third method, VulDistilBERT [104], employs a distilled version of BERT for text encoding, trained with CVE descriptions and severity levels, to estimate the severity of new vulnerabilities.

The summary of results obtained for the three implementations is presented in Table 5.3. In all three cases the predicted output was discrete, corresponding either to categorical severity levels or integer values of CVSS metrics. To enable direct quantitative comparison, the discrete predictions were translated into continuous scores, and the Root Mean Square Error was computed on the reconstructed scores. Alongside this, inference execution time was also measured for each method to capture computational cost.

TABLE 5.3: Analysis of different state-of-the-art AI implementations for CVSS prediction. Root Mean Square Error (RMSE) is calculated by translating the predicted discrete data into continuous data. Adapted from [101]

Model	**Accuracy (%)**	**RMSE**	**Time Execution (s)**	**Classifiers**
Khazaei et al. [28]	99%	0.66	132.85	Random Forest
Costa et al. [98]	87%	1.55	1432.14	Linear
VulDistilBERT [104]	90%	3.63	195.49	Linear

5.3.3. Clustering Evaluation

This experiment assessed alternative text preprocessing strategies and clustering algorithms for their effect on risk calculation within HAL's configuration analysis process. Two preprocessing methods were considered: bag-of-words and sentence embeddings. Each preprocessing approach was combined with a set of twelve clustering algorithms, including K-means, bisecting K-means, spectral clustering, mean-shift, Gaussian mixtures, BIRCH, agglomerative clustering, affinity propagation, Ward hierarchical, DBSCAN, HDBSCAN, and OPTICS.

For each combination, clusters of CVE descriptions were generated, and the resulting groupings were integrated into HAL's scoring component. The impact was then measured by computing the aggregate system risk score based on the clustered vulnerabilities. Figure 5.4 presents the comparative results, showing aggregate risk scores obtained when applying the different combinations of preprocessing methods and clustering algorithms.

5.3.4. Comparative Risk Assessment

This experiment compared the configuration recommendations produced by three Risk Manager implementations: Heo, Lazarus, and HAL. Each system was executed on the same dataset of OSs and vulnerabilities, and the resulting configurations were evaluated according to two criteria: security level and resilience level.

The security level was measured as the sum of the CVSS scores of all CVEs present in the advised configuration. The resilience level was measured as the product of the reassessed CVSS scores of shared CVEs and their respective EPSS probabilities, considering both explicitly shared and clustered vulnerabilities.

Figures 5.5 and 5.6 present the results of this assessment. Figure 5.5 shows the security score values for each Risk Manager across the evaluation period, while Figure 5.6 shows the corresponding resilience score values derived from the same experimental setup.

5.3.5. OSINT Scraper Performance

This experiment measured the execution performance of the developed OSINT scraper when retrieving vulnerability and exploit data from multiple intelligence sources. Four databases were included in the evaluation: the NVD, ExploitDB, AlienVault OTX, and the OSV database.

The scraper was executed to collect initial datasets from each source, and the execution time, number of retrieved entries, and access requirements were recorded. For sources requiring authentication, API keys were used to comply with rate limits. For databases with downloadable snapshots, such as ExploitDB and OSV, the scraper was configured to import data from the available export archives.

Table 5.4 summarizes the benchmark results, reporting whether each database was scrapable, the total execution time in seconds, the API key requirement, and the total number of retrieved entries.

Table 5.4: Scraper benchmarks. Adapted from [101].

Database	Scrapable	Time (s)	API Key	Entries
NVD	Yes	390.29	Yes	277,152
ExploitDB	Yes*	3.42	No	46,575
AlienVault OTX	Yes	99,774	Yes	277,152
OSV	Yes*	235.44	No	255,743

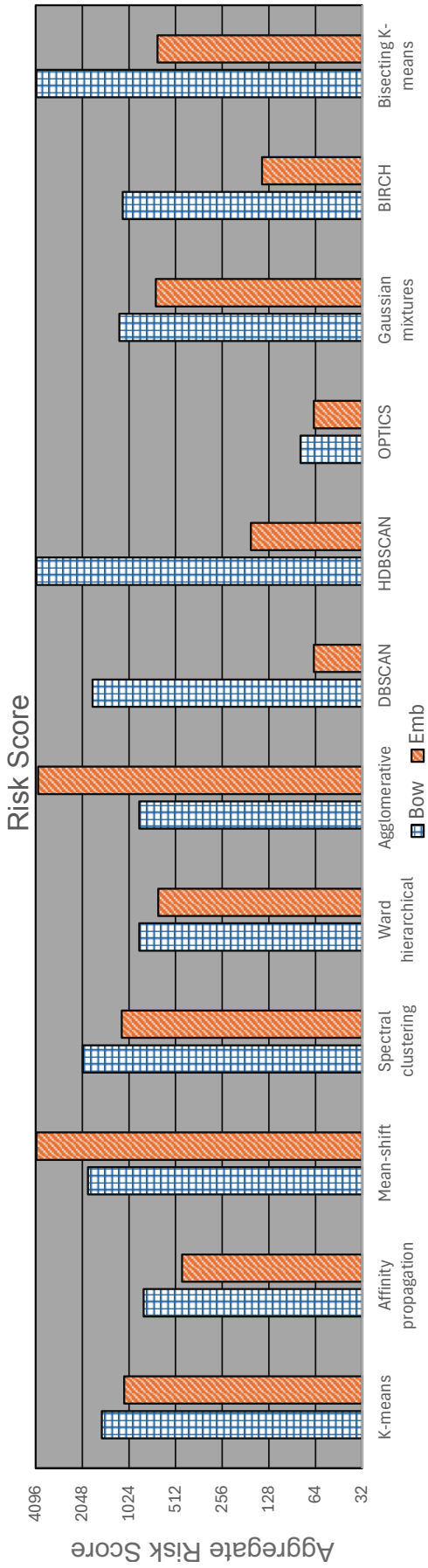

FIGURE 5.4: Aggregate risk scores for various clustering algorithms. OPTICS with sentence embeddings achieves the best performance.

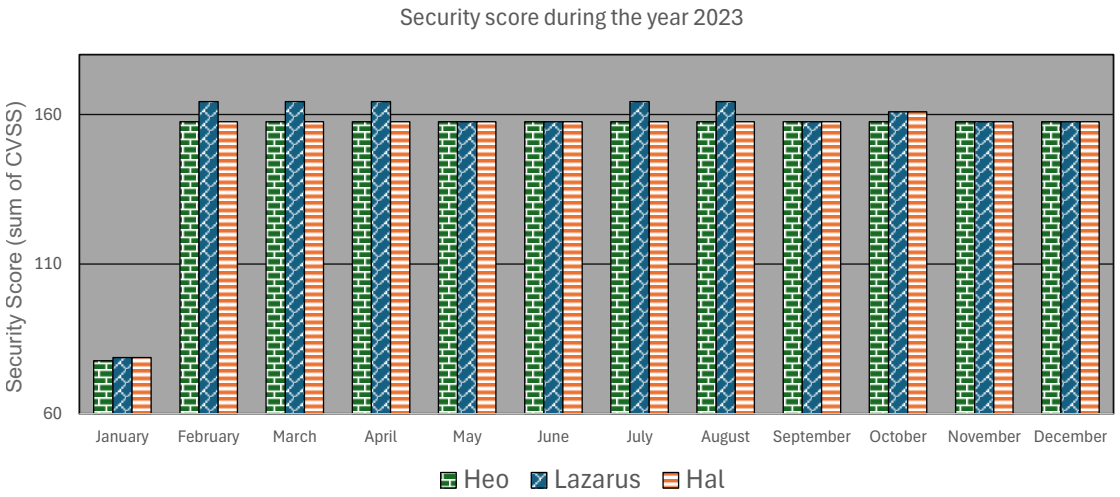

FIGURE 5.5: Evaluation of the different Risk Managers, considering the level of security, i.e., the sum of the CVSS score of each CVE present in the advised configuration (lower is better). Adapted from [101].

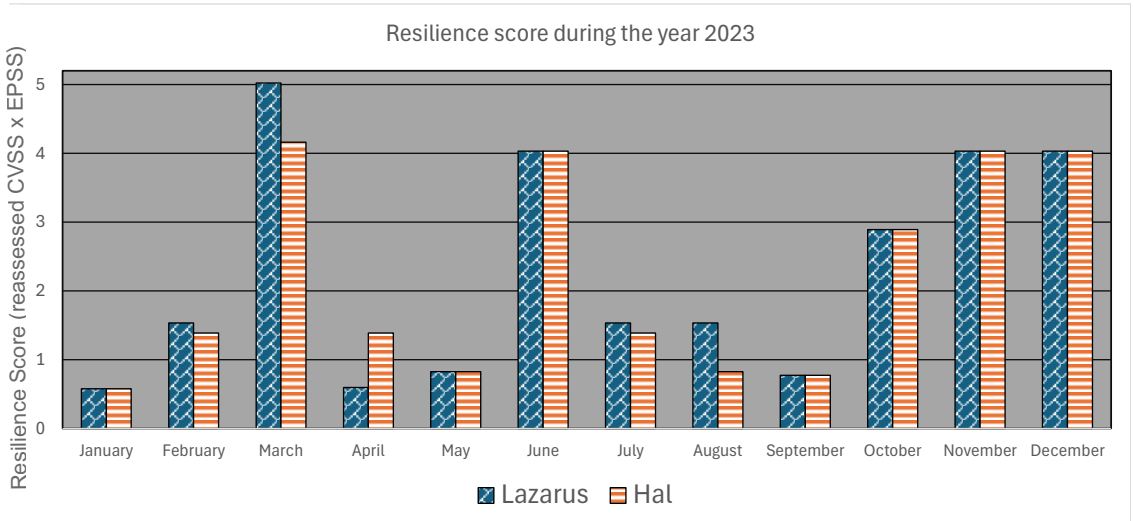

FIGURE 5.6: Evaluation of the different Risk Managers, considering the level of resilience, i.e., the multiplication between the reassessed CVSS score of the common CVEs and respective EPSS score (by shared CVE or by clustering) present in the advised configuration (lower is better). Adapted from [101].

5.4. Discussion

This section, interprets the results obtained from the experiments presented in the previous section. Rather than simply presenting performance metrics, this discussion examines what the observed results reveal about the effectiveness of HAL's workflow and its integration of ML and OSINT-driven intelligence. In doing so, it highlights how the experiments validate the proposed architecture, identify the key advantages and innovations demonstrated by the results, and reflect on the remaining limitations and challenges. By combining empirical evidence with critical analysis, this section provides a balanced assessment of HAL's potential as a next-generation Risk Manager and outlines directions for future improvement.

5.4.1. Insights from the CVSS Score Prediction Experiment

The CVSS prediction experiment (Table 5.3) shows that different strategies for vulnerability scoring produce distinct trade-offs. Khazaei et al.'s method achieved the highest accuracy (99%) and the lowest Root Mean Square Error (0.66), while also offering relatively fast execution. This combination makes it suitable for integration into HAL's real-time scoring workflow, where rapid and reasonably precise predictions are more valuable than detailed interpretability. In contrast, Costa et al.'s DL approach incorporating NLP, although conceptually more granular by predicting vector metrics individually, incurred high computational overhead (over 1400 seconds per execution) and a higher Root Mean Square Error, which limits its practicality for an online system such as HAL. VulDistilBERT provided competitive accuracy at 90% but suffered from high Root Mean Square Error and difficulty mapping discrete severity levels into continuous CVSS scores, reducing the risk interpretation. Taken together, the results justify HAL's adoption of the Khazaei method as the default score predictor: it balances predictive reliability with execution speed, an essential property for a Risk Manager that operates under strict time constraints. Nonetheless, incorporating insights from Costa's or VulDistilBERT's methods as complementary checks could enhance explainability, even if not used directly in HAL's operational pipeline.

5.4.2. Insights from the Clustering Evaluation

The clustering evaluation (Figure 5.4) highlights the importance of text representation and algorithm selection for grouping vulnerabilities with similar risk characteristics. The

results demonstrate that sentence embeddings outperform the bag-of-words approach, capturing nuanced semantic relationships that traditional frequency-based features could not. Among the clustering algorithms, density-based methods such as OPTICS and DBSCAN yielded the lowest aggregate risk scores and proved more robust to outliers than K-means and its variants. This indicates that HAL benefits significantly from adopting embeddings with OPTICS, since both scalability and insensitivity to noise are critical for continuously evolving vulnerability datasets. Compared to Lazarus, which relied on manual tuning with K-means, HAL achieves automated and reliable clustering. This improvement addresses one of the major drawbacks of prior systems, execution delays due to expert oversight, and supports HAL's design objective of minimizing human bias in adaptive risk calculation.

5.4.3. Insights from the Comparative Risk Assessment

The comparative assessment of HAL against Heo and Lazarus (Figures 5.5 and 5.6) illustrates how different design methods balance security and resilience. Heo produced configurations with generally lower cumulative CVSS scores, reflecting a bias towards minimizing direct vulnerability impact, but did not consider resilience across replicated nodes, leaving the system exposed to parallel compromise scenarios. Lazarus, by contrast, achieved stronger resilience by penalizing shared vulnerabilities, but in doing so sometimes accepted higher overall risk exposure. HAL outperformed by combining both objectives: it maintained security levels close to or better than Heo in several months while also improving resilience scores relative to Lazarus during critical intervals such as February, March, and July. This validates HAL's design of tunable prioritization, where policies can dynamically emphasize risk minimization or configuration resilience depending on mission requirements. Occasional cases where HAL produced configurations that were more secure but less resilient (e.g., April) highlight that trade-offs still exist; however, the system is explicitly designed to allow operator-driven tuning of such trade-offs, whereas competitors are locked into one approach. These results confirm HAL's operational versatility for ITS deployments where both resilience against parallel failures and reduction of total exposure matter.

5.4.4. Insights from the OSINT Scraper Performance Evaluation

The OSINT scraper benchmarks (Table 5.4) reveal the feasibility and limitations of integrating diverse threat intelligence sources into HAL's local database. While ExploitDB and OSV proved lightweight to retrieve due to downloadable archives, AlienVault OTX

presented the largest operational challenge because of strict API rate limits and the lack of bulk retrieval options. These constraints led to long execution times for the initial data collection (almost 100,000 seconds). Nonetheless, once bootstrapped, incremental updates require only small periodic queries, which makes production deployment viable. The NVD and OSV provided comprehensive coverage (more than 250,000 entries each), establishing a solid baseline for HAL's scoring. By automating ingestion into a local database, HAL reduces dependency on single authoritative feeds and strengthens resilience to external service outages. Compared with prior managers that drew almost exclusively from NVD and ExploitDB, HAL demonstrates a clear expansion in intelligence scope, which improves the timeliness of insights and reduces the vulnerability window before official scoring. These results dim the scraper as a complementary tool for HAL's automated execution: while score prediction accelerates initial scoring, OSINT integration ensures continuous, real-world relevance.

5.4.5. Limitations and Future Work

Despite these innovations, several limitations remain that define the scope of future research. One important avenue is the further validation and automatic weighting of intelligence sources within the VExHK module, including the incorporation of anomaly detection and misinformation filtering to counter the risk of unreliable open-source intelligence. Another direction involves the deeper integration of data obtained from penetration tests, active vulnerability scans, and potentially dark web intelligence feeds, which would extend HAL's situational awareness beyond standard OSINT databases. The computational overhead and overall resilience of the framework also require evaluation when exposed to adversarial conditions, particularly scenarios that involve intentionally poisoned or manipulated data streams. Finally, the system must be subjected to operational deployment at scale to assess the impact of its recommendations in practical settings, with special attention given to its suitability for real-time and mission-critical intrusion-tolerant systems where performance, reliability, and security are equally critical factors. Despite leveraging multiple sources of OSINT, HAL's current reliance on publicly accessible databases such as NVD, ExploitDB, AlienVault OTX, and OSV introduces challenges in terms of data completeness, timeliness, and integrity. Vulnerability assessment and scoring processes are inherently delayed due to manual validation and scoring bottlenecks within primary repositories. For example, the current backlog at the NVD increases the time before newly

reported vulnerabilities are processed and scored, impacting risk calculations for emerging threats.

Further, while the integration of ML- or AI-based CVSS prediction enables rapid assessment of unknown or zero-day vulnerabilities, it cannot fully substitute expert human review. Nuances and contextual subtleties of certain vulnerabilities may be misrepresented by automated models, resulting in imperfect interim scores. These predictions are intended as temporary measures and always require post hoc refinement once the evaluation becomes available.

HAL's effectiveness may be compromised by adversarial activities targeting its local intelligence repositories or poisoning its machine learning models. Even though a secured control plane is assumed, the occurrence of successful attacks cannot be entirely ruled out. Input injection attacks or the ingestion of misleading OSINT could degrade its risk assessment, introducing new security concerns for the risk management process itself.

The scraper tool, though automated, remains subject to limitations imposed by third-party data source rate limits, inconsistent support for batch queries, and architectural constraints (such as those imposed by ExploitDB and AlienVault OTX APIs). These factors can create data gaps, increase latency, and pose maintenance challenges when scaling to larger OSINT landscapes.

For future work, ongoing research aims to integrate expanded data sources, particularly from penetration testing tools, automated test suites, and additional real-time cybersecurity platforms, including dark web monitoring. Enhanced intelligence coverage will allow for improved responsiveness and a more holistic view of the threat landscape, but requires robust vetting mechanisms to counteract possible misinformation and ensure reliability.

To address the issue of data authenticity, future versions of HAL will embed validation and provenance-tracking mechanisms. This includes cross-referencing threat intelligence, employing ML for anomaly detection, and weighting information from verified sources higher within risk calculations. Such innovations seek to minimize the influence of false reports, fake CVE claims [130], and information purposely injected by malicious entities to mislead automated systems.

Real-world deployments in diverse infrastructure environments are also considered, enabling empirical study of HAL's operation overhead and adaptive capabilities under adversarial conditions. This will create opportunities to examine practical trade-offs between

risk assessment accuracy and system scalability, as well as refine algorithms to balance resilience and security when making configuration recommendations for ITSs.

Finally, further research will pursue continuous improvement in automated vulnerability clustering and scoring, aiming to reduce execution time, minimize human intervention, and maximize the system's ability to accurately recommend secure and resilient configurations in rapidly evolving threat scenarios.

5.5. Chapter Summary

This chapter presented the design, implementation, and evaluation of HAL, a next-generation Risk Manager tailored for proactive risk assessment and adaptive configuration in ITSs. Building upon prior research, HAL integrates automated ML-driven prediction of vulnerability scores with advanced clustering techniques and real-time data ingestion via the VExHK scrape module. This combination enables continuous, dynamic evaluation of system risk that addresses key limitations of traditional approaches relying on delayed, manual vulnerability scoring.

The architecture comprises four main components: vulnerability score prediction, clustering, score reassessment, and configuration optimization, functioning together to recommend resilient and secure ITS configurations. The score reassessment module enhances existing methodologies by incorporating factors such as vulnerability age, patch availability, exploit likelihood from the EPSS, and empirical intelligence from AlienVault OTX pulses. The clustering module leverages state-of-the-art ML algorithms and sentence embeddings to identify semantically similar vulnerabilities, reducing redundancy and improving threat correlation across heterogeneous systems.

Experimental results demonstrated HAL's capability to generate configurations that balance security and resilience compared to existing Risk Managers such as Lazarus and Heo. The predictive components, notably based on the approach by Khazaei et al. [28], provide accurate and timely vulnerability scoring with minimal computational overhead, enabling responsiveness to newly disclosed threats. Additionally, the VExHK scraper successfully aggregates and normalizes intelligence from multiple OSINT sources, enhancing the currency and completeness of the vulnerability database.

Despite these advancements, challenges remain concerning data quality, timeliness, and the potential impact of adversarial attempts to manipulate inputs or ML models. Future work aims to extend HAL's capabilities by integrating additional intelligence sources, improving data validation and anomaly detection, and refining the balance between security

and resilience in configuration decisions. Planned real-world deployments will further assess operational effectiveness and scalability under adversarial conditions.

Through HAL, the research field of ITS risk management advances by harnessing automation, machine learning, and comprehensive intelligence integration to provide a more agile, informed, and robust defense posture against evolving cybersecurity threats.

6. Security Orchestration for EVs: A Collaborative Approach within ITS Research

The present chapter situates EVSOAR within the broader scope of intrusion-tolerant orchestration research, highlighting how the project's architectural, experimental, and collaborative outcomes extend and ground the thesis's core technical contributions.

During the course of the doctoral research, a collaboration was established with the research group RC3 at King Abdullah University of Science and Technology (KAUST), focusing on the development of a novel SOAR architecture tailored for EV cybersecurity through the EVOLVE platform [30].

This chapter presents the joint effort resulting in the EVSOAR project, which leverages the EV CSN as a unique edge computing infrastructure to address the increasing cybersecurity challenges posed by the expanding attack surface of modern connected and autonomous vehicles. The collaboration aimed to overcome limitations in existing VSOCs, which predominantly rely on shared wireless networks such as 4G, 5G, and WiFi. These wireless infrastructures suffer from latency, bandwidth bottlenecks, and interference due to their shared nature, adversely affecting the timely deployment of security updates and the responsiveness of defense mechanisms.

The EVSOAR architecture repurposes the physical and network infrastructure of the EV CSN to create a distributed, scalable, and responsive security orchestration system. By utilizing direct physical connections between vehicles and charging stations through dedicated PLC channels, computation-intensive tasks such as real-time log analysis and FL-based and ML-based IDSs are deployed within or closer to the vehicle. This edge-oriented design reduces reliance on cellular networks, enhances communication reliability, and fosters cross-vendor collaboration through shared threat intelligence.

Experimental evaluations demonstrated improvements in latency and network stability, with EVSOAR achieving a minimum 41% improvement over existing 5G-based VSOCs for small payload transmissions. Moreover, the system's FL approach permitted privacy-preserving collaborative model training across multiple OEMs while maintaining detection accuracy comparable to traditional ML techniques.

This collaboration provided valuable insights into SOAR and IDS that complement the

doctoral research focused on the design of orchestration architectures for ITS. ITS constitute a foundational cyber resilience paradigm characterized by architectures and protocols that guarantee system availability, integrity, and correct operation despite the presence of intrusions or faults. While ITS research traditionally emphasizes fault-tolerant consensus mechanisms, proactive-reactive recovery strategies, and replica diversity for attack mitigation, integrating cybersecurity orchestration concepts represents a frontier for enhancing ITSs adaptability and responsiveness.

The EVSOAR collaboration concretely illustrates how SOAR principles can be embedded within these systems to enable automated detection and mitigation actions at multiple distributed levels, a concept that aligns with the vision of a secure overseer orchestrating ITS components to dynamically respond to threats.

The technical synergies between EVSOAR and the broader ITS orchestration architecture, specifically Skynet, are visible in the architecture presented in Figure 4.1 and described in Chapter 4. EVSOAR applies SOAR techniques and methods that can be reapplied to ITS to augment capabilities for monitoring and reacting to security-relevant events. EVSOAR's use of FL and edge-based detection highlights innovative approaches to managing distributed data privacy concerns and computational constraints—issues equally pertinent to ITS deployments spanning diverse critical infrastructures while maintaining team and stakeholder privacy.

The experience gained in architecting EVSOAR's hierarchical SOAR agents, from in-vehicle modules to edge charging stations and centralized management, provides insights into integrating SOAR capabilities into ITSs for policy enforcement and automated response workflows. Furthermore, EVSOAR's evaluation under emulated network conditions underscores the importance of robust communication infrastructures and adaptive control mechanisms to maintain intrusion tolerance in dynamic, resource-constrained environments.

This collaboration also underlines challenges that resonate with ITS research goals, specifically, balancing privacy with cross-entity collaboration and providing effective and tangible ways to address intrusions. The EVSOAR project's approach to leveraging trusted platform infrastructures, cryptographic assurances, and privacy-preserving FL techniques provides pathways to address these challenges, which are relevant to the ITS context where fault boundaries and trust assumptions are inherently constrained.

The KAUST collaboration on EVSOAR exemplifies an interdisciplinary research endeavor enriching the doctoral investigation into intrusion-tolerant orchestration architectures. It extends the theoretical and systemic foundations of ITS by grounding them in a tangible, vehicular cybersecurity use case marked by unique architectural and operational characteristics. This synergy not only validates the applicability of orchestration concepts in emerging cybersecurity domains but also informs ongoing enhancements toward designing adaptive, scalable, and privacy-conscious ITS. The collaborative experience and outcomes thus constitute an integral component of the doctoral research narrative, contributing situational and practical relevance to the Skynet architecture.

The remainder of this chapter is organized as follows: Section 6.1 reviews the state of the art in VSOCs and intrusion detection systems; Section 6.2 details the EVSOAR architecture including its threat model and key applications; Section 6.3 discusses the experimental evaluation comparing EVSOAR with state-of-the-art SOCs; Section 6.4 presents the connection to Skynet; Section 6.5 explores the technical and research synergies; Section 6.6 provides a chapter summary, concluding it.

6.1. Background and State of the Art: VSOC

The automotive industry has witnessed rapid advancements with the integration of modern EVs, autonomous driving, software-defined architectures, and vehicle-to-everything (V2X) communication. This technological progress has resulted in highly connected vehicles, but also expanded the attack surface for cyber threats. Traditional in-vehicle security mechanisms, including firewalls, gateways, and IDSs, struggle to adapt to the unique challenges posed by heterogeneous protocols, real-time processing constraints, and limited computational resources inherent to vehicles [129].

To address these limitations, SOCs, originally developed for IT environments, have evolved into VSOC. VSOCs are designed to continuously monitor vehicle behavior, detect anomalies and intrusions, and respond to cybersecurity threats across both in-vehicle and off-vehicle domains. Central to VSOCs operations are SIEM and SOAR tools, which aggregate data, automate threat detection, and coordinate responses among vehicle manufacturers (OEMs), suppliers, and infrastructure providers.

Despite these advances, state-of-the-art VSOC solutions rely on wireless infrastructure such as 4G, 5G, and WiFi to communicate with the vehicle. These shared networks frequently suffer from congestion, limited bandwidth, and increased latency, leading to delays in applying security updates and transmitting critical threat information. As the number

of connected vehicles and other wireless devices increases, scalability and performance challenges become more pronounced, and conventional wireless-based VSOCs struggle to meet the evolving demands of automotive cybersecurity.

Recent research highlights additional complexities introduced by modern in-vehicle communication networks and advanced Electronic Control Unit (ECUs). While these technologies offer sophisticated capabilities, they also heighten vehicle vulnerability to cyberattacks [131]. Legacy security tools such as IDSs, firewalls, and gateways are often insufficient for addressing these new risks. Although SOCs play a crucial role by centralizing data analysis and orchestrating incident response [132], their adaptation to the automotive context must confront multiple constraints, including stringent timing, multi-protocol integration, and limited available processing power within vehicles [133].

Moreover, the local scope of traditional IDS approaches limits system-wide visibility, necessitating integrative frameworks like SIEM and SOAR, which can synchronize threat intelligence and responses across manufacturers, suppliers, and infrastructure partners. Various academic studies have addressed the evolution of SOCs into VSOCs, describing foundational legal, technical, and architectural requirements and implementing prototypes in real-world contexts [134, 135, 136, 137, 138, 139]. The majority of these solutions remain dependent on wireless communication, which leads to resource competition and scalability concerns as the volume of vehicle-generated data grows.

EVSOAR proposes a distinct approach by exploiting wired connectivity through the EV CSN. This architecture promotes stability, predictable resource allocation, and quality of service, all while supporting seamless integration of advanced SOAR functionalities [129]. Compared to shared wireless network infrastructures, the EV SOCs enabled by the EVs CSNs deliver superior scalability and reliability through direct connections, allowing security services to expand naturally alongside the growth of charging stations. This one-to-one mapping between vehicle and charging infrastructure significantly simplifies deployment and delivers robust capacity to the automotive cybersecurity ecosystem.

6.2. EVSOAR Architecture, Dataflow and Innovations

The EVSOAR architecture, depicted in Figure 6.1, consists of three primary elements: the vehicle-installed SOAR Agent, the Edge-SOAR components located at each charging point, and the Central SOAR system hosted in the cloud. Detailed specifications of the foundational layers required for deploying EVSOAR are referred to in the EVOLVE publication [30].

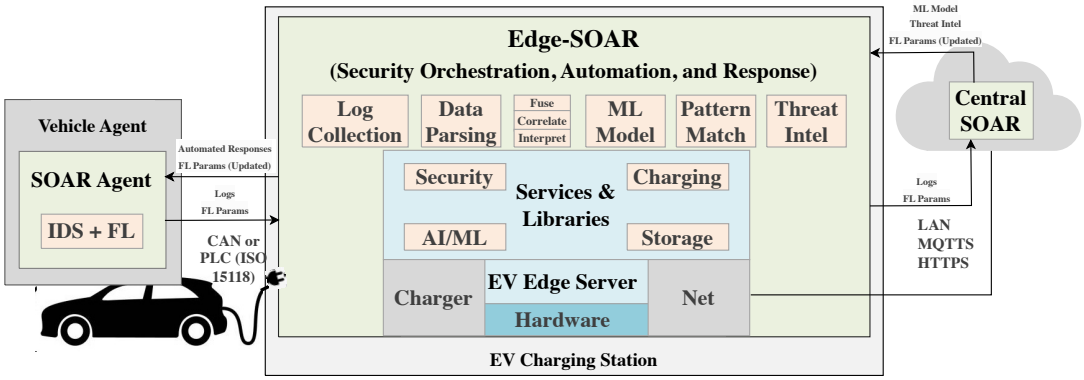

FIGURE 6.1: EVSOAR architecture

The **SOAR Agent**, within the vehicle, enforces policies defined by the Central SOAR concerning the collection and handling of logs, events, and alert triggers throughout the vehicle's operation. It also allows communication with OEMs by transmitting vehicle-generated logs and events, notifying them of detected malfunctions, and managing requests for security patches or updates based on log analyses. Depending on the vehicle design, essential data and alerts can reside in components such as the Event Data Recorder, Central Gateway ECU, Telematics Control Unit (TCU), ECU internal memory, or the Onboard Diagnostics System.

Installed at EV charging stations, the Edge-SOAR nodes create a secured physical connection to the SOAR Agent via the charging cable standards (CCS Type-2 or CHAdeMO 3.0), employing secure communication channels such as TLS or VPN for data exchange. These Edge-SOAR units are interconnected through the EV CSN, enabling collaborative threat intelligence sharing and orchestrating coordinated responses within defined geographic zones. For wider area coverage, communication between stations and broader networks relies on WAN or internet connections, ensuring system resilience via a distributed architecture.

When a vehicle plugs into a charging station, the Edge-SOAR leverages the SOAR Agent connection to retrieve logs, alerts, and events through the charging cable. This cable is presumed to carry a dedicated software communication channel, either integrated alongside the power lines or utilizing PLC technology [140]. Collected logs may be subjected to analysis by an IDS, which scans for intrusion patterns or forwards data to the Central SOAR for comprehensive examination. Upon detection of confirmed intrusions, the IDS can initiate immediate automated actions, if in the presence of a known threat, or notify the central entity to evaluate and update response strategies. Moreover, the Edge-SOAR

incorporates rule-based mechanisms to autonomously address malicious events, intrusions, or malfunctions reflected in the logs.

The **Central SOAR**, typically operated by a dedicated third-party security team, maintains connections with all Edge-SOAR nodes deployed across participating charging stations. It aggregates and preprocesses incoming data, presenting it for detailed analysis and investigation by security analysts. The central SOAR's functions include detecting security incidents from received logs and events, deploying automated countermeasures through Edge-SOAR units, supporting IDS model training with centralized ML on gathered data, or aggregating FL parameters supplied by SOAR Agents to improve detection models. Additionally, it maintains bidirectional communication channels with OEM servers, facilitating the dissemination of newly identified attack signatures and the reception of patches or mitigation updates. These SOAR connections extend to software suppliers responsible for vehicle systems, recognizing that vehicles may operate software from multiple vendors.

Figure 6.2 presents the overall communication sequence: a vehicle connects to a charging station, which consults the cloud for the latest threat intelligence, performs threat assessments, relays notifications back to the vehicle, which may then request security fixes, and finally uploads logs back to the cloud for continued monitoring and analysis.

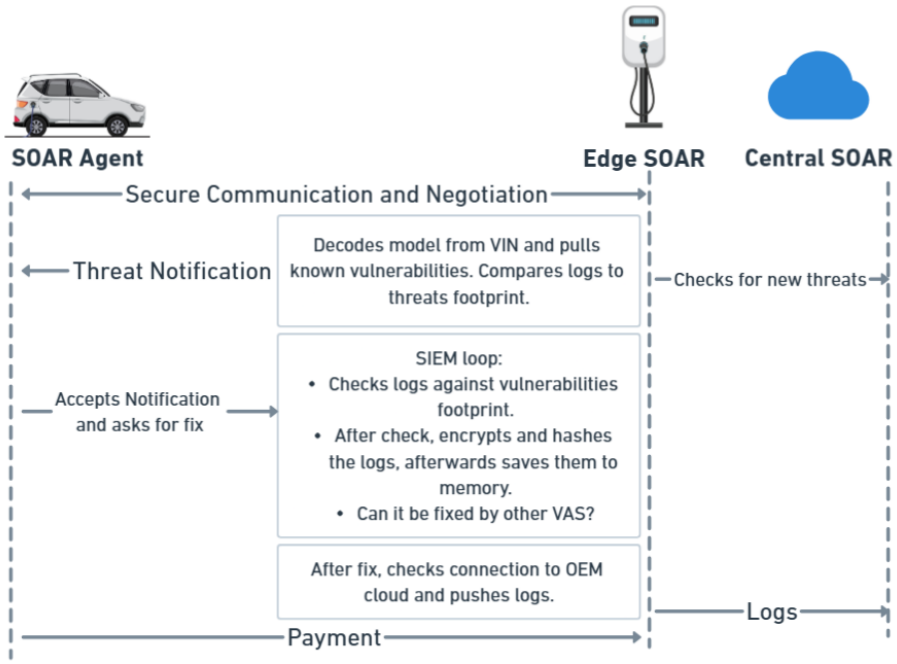

FIGURE 6.2: EVSOAR communication flow

The innovations of the EVSOAR architecture stem from its novel integration of EV charging station infrastructure as an edge computing platform, fundamentally transforming how SOAR are managed within the vehicular ecosystem. Unlike traditional wireless-dependent approaches, EVSOARleverages the physical link between the vehicle and the charging station, enabling secure, low-latency bidirectional communications. This unique connection bypasses many of the stability, congestion, and bandwidth limitations of 4G, 5G, and WiFi networks, delivering reliable data transmission essential for security-critical functions.

A central innovation is the hybrid design of intrusion detection. Heavy computational tasks, such as model training, are performed centrally in the cloud, after which updated models are deployed to charging stations for fast, local assessment. This ensures immediate detection and response capabilities close to the vehicle. At the same time, EVSOAR addresses privacy concerns using an alternative FL approach. Sensitive log analysis is conducted entirely on the vehicle, with only model parameters, i.e., raw data is kept within the vehicle, shared for aggregation and global threat model improvement. This balance between privacy preservation and collaborative intelligence is particularly impactful for data governed by strict privacy requirements, such as infotainment and telematics, while more traditional ML can be applied where privacy is less sensitive.

Another hallmark of EVSOAR is its automation of mitigation and adaptive defense mechanisms. Automated responses are triggered at the network edge, allowing swift isolation of compromised components, remedial software patching, and rollback actions on detected threats without the delay of round-trip communication to a remote data center. The system's adaptive logic enables scalable, reliable, and context-aware protection, all while minimizing exposure and operational risk.

By shifting detection and mitigation closer to where attacks can occur, exploiting the growing EV charging infrastructure, and integrating real-time, privacy-preserving analytics, EVSOAR offers a shift from legacy VSOC frameworks. Its architecture is inherently scalable (scales with the CSN), robust to network variability, able to facilitate secure cross-vendor intelligence sharing, and optimized for the stringent requirements of modern vehicle cybersecurity.

Furthermore, the choice of privacy-preserving FL methods is motivated by technical concerns and by growing regulatory requirements, such as GDPR and automotive-specific

cybersecurity standards [141, 142, 143], which increasingly govern the management and sharing of vehicle data.

6.3. Experimental Evaluation and Results

The experimental evaluation of EVSOAR demonstrates advances in performance, reliability, and security effectiveness compared to state-of-the-art vehicle security operation architectures. Through a comprehensive set of emulated network scenarios and real-world-inspired use cases, SOAR's use of EV CSN as edge security platforms was systematically validated against both wireless-based (VSOC-4G/5G, RSU-WiFi) and alternative wired solutions.

The EVSOAR prototype was implemented in Golang with the ELK stack providing the SOAR platform's monitoring and threat detection functionality. Testing was carried out within the Emulab environment, enabling multiple dedicated physical machines and the precise emulation of realistic network characteristics, such as bandwidth, latency, and Packet Loss Ratios.

The experiments evaluated six network types: VSOC-4G, VSOC-5G, Road Side Unit-WiFi, EVSOAR-PLC10M, EVSOAR-PLC100M, and EVSOAR-PLC1G, capturing both wireless and wired communication conditions. The setup of each network is presented in Table 6.1.

Table 6.1: Distinct link configurations.

Type of connection	**Bandwidth (Mbps)**	**Latency (ms)**	**PLR (%)**
VSOC-4G	30	36	0.2
VSOC-5G	100	17	0.2
RSU-WiFi	27	100	17
EVSOAR-PLC10M	10	2	0
EVSOAR-PLC100M	100	2	0
EVSOAR-PLC1G	1000	2	0

Benchmarking included direct comparison to SOC4AS and XL-SOC as representative SOAR-enabled vehicle security systems. Through controlled variation of protocol parameters, key performance measures, throughput, network stability, and median round-trip time (RTT) were quantified. These are critical for understanding responsiveness and reliability in real-time vehicular security scenarios, particularly relating to data exchange between vehicles and the security operations center. The protocol provided the basis for deep functional comparisons, including practical threat detection, response flows, and nuanced privacy-accuracy trade-off analyses between ML and FL IDS.

The results, presented in Figure 6.3 and 6.4, demonstrate that wired communication via the EV CSN yields lower latency and higher throughput than wireless alternatives. For small payloads, EVSOAR realized a minimum 41% performance improvement over VSOC-5G, despite its comparable theoretical bandwidth to EVSOAR-PLC1G, underscoring the practical limitations of shared and contended wireless media. The PLC1G setup delivered low and consistent upload times around 10 ms, attributable to its dedicated 1 Gbps bandwidth, whereas wireless configurations suffered from variable and often degraded performance, particularly under high network load. For very large payloads, VSOC-5G can outperform EVSOAR, but the advantage remains minimal and offset by shared network constraints.

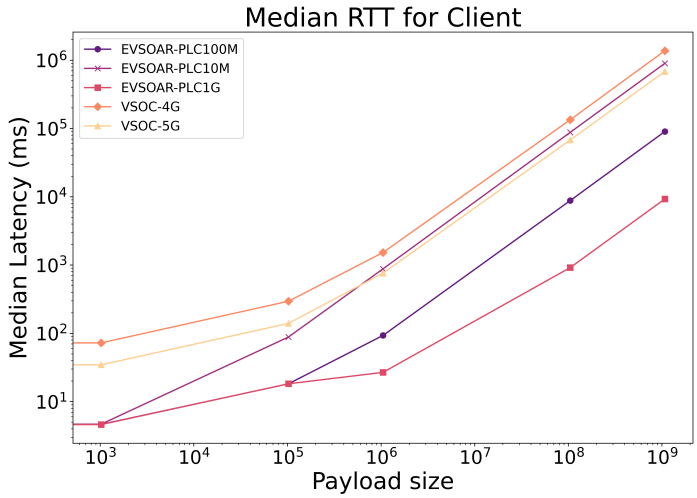

FIGURE 6.3: Median Client RTT

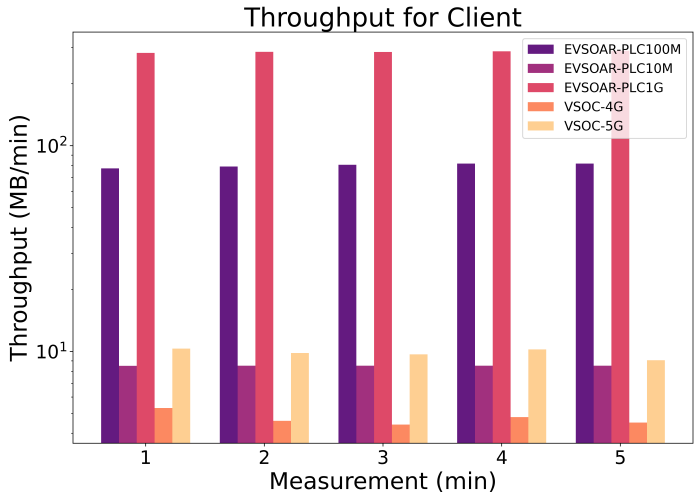

FIGURE 6.4: Throughput for Client

Stability assessments, illustrated in Figure 6.5, showed more consistency in packet delivery times for wired links; EVSOAR's transmission times typically varied by only a few milliseconds, in comparison with wireless links, where delays fluctuated (from hundreds of milliseconds to several seconds), a function of environmental interference and dynamic conditions. This stability is especially crucial for security-sensitive and time-critical actions, as wireless unpredictability can directly impede the execution and timeliness of defense mechanisms. The adoption of the EV CSN thus addresses a reliability bottleneck for vehicle cybersecurity.

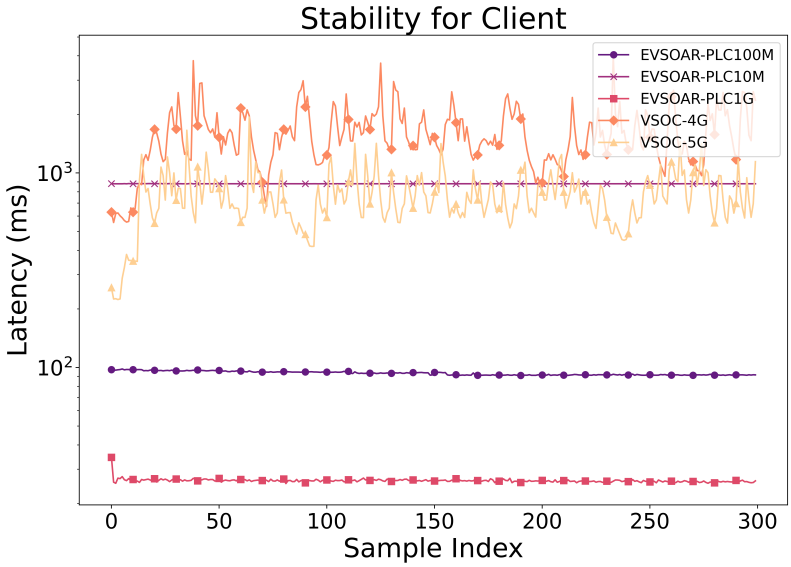

FIGURE 6.5: Network stability

Security effectiveness was further tested by evaluating ML-IDS and FL-IDS approaches in practical deployment scenarios, presented in Table 6.2. Centralized ML-IDS had higher overall accuracy (97%) and recall for normal and compromised data, enabled by full access to vehicle-generated logs. However, this comes at the cost of greater privacy exposure and larger volumes of sensitive data being handled centrally. The FL-IDS, using only abstracted model parameters, achieved strong privacy guarantees and, in multi-OEM configurations, outperformed the ML-IDS in recall for compromised data (87% vs. 82%, a 5% gain). Although the FL-IDS showed lower overall accuracy, it surpassed in minority-class detection, and achieved acceptable trade-offs, i.e., notably lower data overhead and false positives, especially suited for domains like telematics or infotainment.

Further assessing FL performance, the data shows that training with pooled, multi-OEMs datasets boosts detection rates for compromised data (from 75% with single-OEM to 87%

Table 6.2: Combined Results Across ML, FL, and FL Mix

Setup	Class	Recall	Support	Data Size	Model
ML	0	1.00	2,443,833	171 MB (Logs)	XGBoost
	1	0.82	510,341		
	Acc.	0.97			
FL IDS Single OEM	0	0.93	301,818	844 KB (Params)	Feed Forward Neural Net.
	1	0.75	45,527		
	Acc.	0.91			
FL IDS Multi OEM	0	0.89	301,818	844 KB (Params)	Feed Forward Neural Net.
	1	0.87	45,527		
	Acc.	0.89			

with multi-OEM pooling), demonstrating the value of federated, cross-vendor intelligence sharing. However, this also raises challenges around inter-organizational privacy, where even shared model parameters could leak sensitive patterns. Careful mechanisms and trust frameworks are needed for safe collaborative deployments.

Taken together, the experimental data shows that EVSOAR's architecture, leveraging the bandwidth, locality, and stability of EV charging infrastructure, not only addresses longstanding performance and scalability bottlenecks of wireless vehicle network security but also lays the groundwork for advanced, privacy-preserving, and collaborative intrusion detection.

6.4. Connection to Skynet

The EVSOAR architecture provides actionable knowledge for advancing the capabilities of Skynet by demonstrating how distributed SOAR systems can collaborate efficiently, provide security, and share cyberthreat information in complex, dynamic environments. Integrating the experience and core design principles from EVSOAR—namely, distributed security orchestration, hierarchical and modular architecture, adaptive and automated response, FL with collaborative intelligence, privacy-awareness, and hybrid rule-based/ML-driven IDS—into Skynet can enhance its defensive capabilities while opening new research directions for large-scale, distributed cyber defense through robust SOAR integration at distinct levels of the infrastructure.

The deployment of SOAR capabilities at the edge, exemplified by the Edge SOARs in EVSOAR, demonstrates the effects of localized, real-time detection and response, reducing incident response latency and minimizing the impact of network disruptions or attacks. Integrating such SOAR modules within Skynet's modular architecture allows for granular monitoring, automated orchestration, and context-aware decision-making distributed throughout the infrastructure. This model supports the core Skynet goal of adaptive,

intrusion-tolerant oversight by enabling dynamic reconfiguration and automated defense rooted in field data.

A key insight from EVSOAR is the value of harnessing localized knowledge. The ability of Edge SOARs to monitor, correlate, and share insights within its operational domain can be repurposed in Skynet to create a scalable federation of security overseers. With multiple Skynet instances deployed across diverse environments, the lessons from EVSOAR enable these instances to collaborate in near-real time, pooling observations, validating threat intelligence, and reinforcing cross-location defense strategies. This collaborative approach directly addresses Skynet's research objective of self-organizing, mutually supportive security infrastructures.

Several EVSOAR features are especially relevant for integration or redesign within Skynet:

- FL and distributed IDS: The mechanisms developed in EVSOAR for federated model training and privacy-preserving detection can be upscaled to enhance Skynet's ML engines, supporting global analytics without compromising local privacy.
- Reliable edge-to-cloud communication: The architectural solutions for resilient, wired communication using EV charging infrastructure can be generalized to other domains, supporting Skynet's need for robust data exchange across diverse networks.
- Automated response workflows: EVSOAR's incorporation of rule-based and data-driven automated responses, such as patch deployment, threat isolation, and system rollback, provides concrete mechanisms for Skynet to deliver rapid, automated mitigation at both local and global levels.
- Privacy-preserving analytics: By processing sensitive data locally and sharing only model updates or insights, EVSOAR's privacy model can shape Skynet's approach to data minimization and regulatory compliance.
- Performance and scalability validation: Lessons from EVSOAR's experimental evaluation, such as documented latency reductions and throughput improvements from edge orchestration, can inform Skynet's own deployment strategies and highlight the operational value of architectural choices.

EVSOAR demonstrates the feasibility and value of integrating SOAR automation for large-scale, cyber-physical systems. Drawing from these lessons, Skynet can innovate by

enabling a network of collaborating security overseers, each with strong local autonomy, but also benefiting from federated knowledge and adaptive orchestration. This direction supports the evolution toward resilient, distributed, and intelligent cyber defense ecosystems.

6.5. Technical and Research Synergies

Adapting selected elements and insights from EVSOAR's edge-driven security orchestration into Skynet's intrusion-tolerant oversight fosters new opportunities to address complex and evolving technical challenges across distributed cyber defense. This selective incorporation enables Skynet to benefit from practical lessons learned and proven architectural strategies, enhancing its capacity to respond to emerging threats while refining its resilient security mechanisms. At the core of the synergy is the shared objective to combine automation and intelligence across the cyber-physical stack, enabling both platforms to transcend the limitations of isolated security management and reactive, siloed responses. By bridging the gap between localized, context-aware detection and response, exemplified by EVSOAR's real-time operations at EV charging infrastructure, and Skynet's broad, federated-readiness, the combined approach enables adaptive defense strategies that extend across edge and cloud domains.

An advantage of this collaboration is the capability to integrate security automation across diverse operational environments. Through joint orchestration protocols and federated analytic engines, it becomes possible to coordinate incident response and policy enforcement on a higher scale, while still maintaining the implications of local context. With this approach, the integration supports seamless handoff of alerts, actions, and forensic data between distributed components, fostering not only technical resilience but also organizational agility in response to evolving adversarial tactics.

The use of FL and privacy-preserving analytics by both architectures enables a unified research direction focused on mechanisms that protect sensitive data at its source and share only high-level knowledge. This approach also allows detection models to adapt to the specific characteristics and threat profiles of individual domains. This joint effort enables advanced ML techniques to operate at scale, yet with measured trust and without diluting privacy or exposing critical information.

Through this collaboration, both architectures share the same challenges in distributed cyber-physical defense: reconciling local autonomy with collective intelligence, maintaining privacy without sacrificing accuracy, and building trust without centralized dependency. By

aligning their core technical strengths and integrating their research efforts, EVSOAR and Skynet highlight a direction for building resilient, adaptive, and integrated cyber defense ecosystems capable of addressing the challenges posed by increasingly sophisticated and large-scale adversaries.

Several open challenges remain, including: (i) establishing robust trust frameworks for secure cross-organizational FL; (ii) managing residual risks of model inversion or information leakage during cross-vendor collaboration; and (iii) dynamically adapting orchestration in the presence of evolving attack surfaces and adversarial tactics.

6.6. Chapter Summary

This chapter presents EVSOAR—a novel SOAR architecture that leverages the EV CSN as a secure, scalable edge computing platform for modern vehicle cybersecurity. The chapter establishes the motivation for EVSOAR through a review of the evolving automotive attack surface, the limitations of conventional VSOCs, and the bottlenecks imposed by wireless communication infrastructures.

The EVSOAR architecture is detailed, describing its distributed design with in-vehicle SOAR agents, charging station Edge-SOAR modules, and a centralized cloud SOAR system. The system is engineered for real-time log collection, privacy-preserving FL and ML based intrusion detection, and secure, automated response workflows, all maintained by direct, physically secured vehicle-to-infrastructure connectivity.

Extensive experimental evaluation demonstrates that EVSOAR improves reliability, latency, and scalability when compared to leading 4G/5G VSOC alternatives, with a minimum 41% improvement in small-payload data exchange. FL models enable collaborative, privacy-enhancing cross-manufacturer threat intelligence sharing, balancing detection accuracy and data protection.

The chapter also analyzes the technical and conceptual affinities between the EVSOAR design and broader ITS research, particularly Skynet, showing how advancements in edge SOAR, adaptive automation, and privacy-aware analytics inform next-generation intrusion-tolerant orchestration. EVSOAR's results and collaborative development underpin practical, research, and regulatory progress—especially as new compliance requirements like GDPR and UNECE WP.29 increasingly govern connected vehicle security and data management.

In summary, the chapter integrates practical architecture, experimental validation, and

cross-domain research to demonstrate how edge-enabled security orchestration can significantly strengthen the cyber-resilience of connected automotive ecosystems and serve as a blueprint for adaptive ITSs.

7. LegionITS: A Federated ITS Architecture for Scalable, Privacy-Preserving CTI Sharing

The escalating sophistication and prevalence of cyberattacks have stretched traditional defense mechanisms to their limits, often leaving isolated organizations unable to keep pace with rapidly evolving threats. While defenses such as SIEM and SOAR platforms play vital roles in threat mitigation, they frequently fall short against novel and unknown attack vectors. The need for collaborative, resilient, and privacy-preserving frameworks has therefore become paramount, substantially increasing interest in systems that enhance cybersecurity through secure information sharing.

ITS represent a shift from mitigation to tolerance in cybersecurity, focusing on resilience and maintaining core operations even when compromised, thereby providing graceful degradation of service. However, even ITS architectures can become vulnerable to sophisticated adversaries and unforeseen, especially coordinated or novel, attacks when deployed in isolation. To address these limitations, recent research proposes that organizations should implement controlled sharing of CTI and defensive insights through federated or collaborative frameworks, which seek to protect each entity's autonomy and sensitive information by employing privacy-preserving and cryptographic mechanisms.

This chapter explores the foundational concepts, architectural designs, and operational mechanisms of federated ITS frameworks, specifically examining the LegionITS system as a model for secure, scalable, and privacy-aware cyber defense. The discussion encompasses integration challenges, privacy-preserving technologies, and practical case studies, such as responses to zero-day vulnerabilities, ultimately highlighting the imperative for collective resilience in contemporary cybersecurity.

LegionITS is a novel federated architecture designed to advance cyber resilience by enabling secure, privacy-preserving, and collaborative information sharing among independent organizations, specifically tailored for deployed ITSs. Built on the premise that the defensive capability of an isolated entity can eventually become overwhelmed by today's rapidly evolving threat landscape, LegionITS empowers entities to jointly detect, analyze, and respond to cyber threats while preserving data autonomy and confidentiality. The system integrates key concepts from ITSs, FL, and advanced privacy-enhancing technologies,

including but not limited to MHE, DP, and ZKPs, to facilitate actionable CTI and CIS exchange without disclosing sensitive operational details.

By leveraging a distributed ledger for verifiable event recording and modular interfaces for interoperability with platforms such as MISP, SIEM, and SOAR, LegionITS enables scalable, trust-based collaboration across both intra- and inter-organizational boundaries. Through practical mechanisms for secure CTI and CIS sharing, automated incident orchestration, and resilience-building analytics, LegionITS is designed as a model for next-generation collective cyber defense frameworks, providing a robust foundation for situational awareness and agile, privacy-preserving responses to emerging threats.

The remainder of this chapter is organized as follows: Section 6.1 reviews the limitations of existing SIEM, SOAR, and SOC systems and introduces the concepts of ITSs, collaborative defense, and federated cybersecurity architectures. Section 7.2 surveys related work in privacy-preserving CTI sharing and FL frameworks, highlighting existing trade-offs between privacy, utility, and scalability. Section 7.3 defines the system and threat model, including stakeholder roles, trust assumptions, and operational context for decentralized environments. Section 7.4 details the architecture of the proposed federated ITS—LegionITS—covering its core components, data flow, privacy-preserving technologies, and governance mechanisms. Section 7.5 presents operational scenarios and a case study addressing practical intelligence sharing and zero-day containment. Section 7.6 analyzes the privacy-utility trade-off in federated learning through experimental results. Section 7.7 evaluates the system's flexibility and scalability under various extension and integration scenarios. Section 7.8 discusses the design's broader implications and systematically addresses the posed research questions, and Section 7.9 concludes with a summary of the chapter, highlighting the main findings, contributions, and practical implications of LegionITS.

7.1. Background and Motivation

Modern enterprise cybersecurity platforms, including SIEM, SOAR, and SOC, play critical roles in managing threats through log collection, event correlation, and automated response workflows. While effective against known threats, these systems are limited when faced with sophisticated, targeted, or zero-day attacks that evade detection by relying predominantly on predefined rules, signatures, and historical patterns. Moreover, the exponential growth and complexity of security data can overwhelm security analysts, escalating alert fatigue and increasing the risk that subtle attack indicators go unnoticed. Isolated deployments

also lack the broad context of threats experienced across different organizations, creating exploitable visibility gaps for adversaries.

In response to these challenges, collaborative cyber defense through CIS has emerged as a crucial strategy. Sharing CTI across organizations enhances collective situational awareness, enabling faster detection and mitigation of emerging threats. Established standards such as Structured Threat Information eXpression (STIX) and Trusted Automated eXchange of Indicator Information (TAXII) facilitate interoperability by defining common formats and protocols for CTI sharing. Despite these advances, concerns around privacy, confidentiality, and regulatory compliance create significant barriers to open exchange. Organizations may hesitate to share sensitive operational data due to legal risks, reputational concerns, or competitive disadvantages.

Addressing these challenges requires a combination of technical and organizational measures. Privacy-preserving techniques—such as data anonymization, minimization, DP, and advanced cryptographic methods including but not limited to HE and ZKP—are necessary to safeguard shared data while preserving its utility. Complementing these technical solutions, governance frameworks that incorporate accountability, trust management, and policy enforcement are critical to ensuring secure and compliant information sharing. Frameworks like the NIST Cybersecurity Framework provides guidance for managing risk, establishing secure information sharing policies, and fostering trust among collaborating parties.

Federated architectures contribute to a robust approach that blends these technical and governance aspects to enable secure, scalable, and privacy-conscious cooperation. Within a federated model, each participating entity maintains control over its own security infrastructure, sharing only vetted or aggregated intelligence in compliance with agreed policies, using standardized protocols and secure communication channels. This promotes interoperability across heterogeneous environments and enforces confidentiality and integrity through cryptographic guarantees and policy controls. By balancing local autonomy with global threat intelligence, federated architectures establish the technical and organizational foundations necessary for resilient, adaptive collective defense capable of scaling to meet an evolving threat landscape.

7.2. Related Work

Several research efforts have addressed various components relevant to this architecture, including federated architectures, ITSs, CTI, and CIS. However, there is still a gap in frameworks that conceptually integrate these domains into a unified architectural vision. This section reviews seminal contributions that have informed and shaped the design principles of the proposed architecture, LegionITS.

In the area of privacy-preserving CTI sharing, several approaches have been proposed. Freudiger et al. [144] introduced the SIC framework, which leverages selective collaboration via Private Set Intersection (PSI), allowing entities to share intelligence without disclosing unnecessary details. Similarly, Van de Kamp et al. [145] developed a cryptographic method enabling sharing of IOCs and intrusion sightings while safeguarding sensitive information from central aggregators. Complementing these, DNSBloom [146] employs Bloom filters to detect malicious Domain Name System activity while minimizing privacy risks.

Other works focus on more advanced cryptographic techniques. For example, some frameworks utilize HE to compute over encrypted data [147], albeit relying on centralized infrastructures, which pose single points of failure. In contrast, Trocoso-Pastoriza et al. [148] advanced a fully distributed CTI-sharing platform that fuses MHE with FL and DP to collaboratively train predictive models without exposing raw data.

Research in CIS emphasizes privacy principles and trust-enhancement mechanisms. Fisk et al. [149] articulated foundational ideals such as minimal data disclosure and continuous progress in sharing. In contrast, Vakilinia et al. [150] applied blind signatures and ZKP to protect participant anonymity and enable accountability. Similarly, Fuentes et al. [151] proposed PRACIS, a privacy-focused analytics platform based on the STIX standard, combining HE and format-preserving encryption to thwart linkability under server compromise scenarios. Broader perspectives, such as CYBEX-P [152], integrate trusted computing and advanced cryptographic constructs to anonymize both data content and user identities.

The concept of federation itself has been explored in distributed systems research. Early foundational work by Heimbigner and McLeod [153] introduced federation as a means to integrate heterogeneous autonomous databases through a shared schema and federated dictionary, supporting dynamic participation without centralized control. Building on these principles, Wijegunaratne and Fernandez [154, 155] outlined an event-driven federated architecture tailored to enterprise environments, highlighting the importance of dependency separation and controlled interface connection to maintain autonomy across domains.

Advancements in FL methodologies also contribute to this domain. Rizk et al. [156] presented a decentralized FL topology that combines graph-based structures with privacy guarantees via DP and cryptographic masking, demonstrating performance comparable to non-private baselines under rigorous mathematical assumptions.

Focusing on ITS, Veríssimo et al. [9] advocated a paradigm shift from pure prevention to assuming compromise and prioritizing resilience and containment, an outlook further developed in subsequent studies [25, 60, 75, 157]. These advances include adaptive recovery mechanisms, architectural reconfiguration, and automated response capabilities. Modern ITS frameworks increasingly integrate external intelligence sources, such as OSINT feeds and social media data, to continuously enrich detection and mitigation efforts.

Moreover, ITSs can serve dual roles, not only consuming CTI but also producing actionable intelligence that feeds autonomous control planes, enhancing automation and diminishing the period of attacker presence.

Collectively, these contributions form the conceptual foundation for *Legion*: a federated ITS architecture that facilitates privacy-preserving threat intelligence sharing and coordinated automated defenses across distributed organizational networks. The design emphasizes autonomy, resilience, and scalability, incorporating key architectural patterns and security mechanisms suited for the evolving cybersecurity landscape.

7.3. System and Threat Model

LegionITS models its system and threat paradigm around the concept of a federation of organizations, each acting in multiple collaborative roles to ensure resilient, intrusion-tolerant cybersecurity. Every stakeholder within this federation is simultaneously a data provider, a data processor, and a data consumer. Data providers in the system include intelligence agencies, cyber defense units, and internal platforms such as MISP instances or OSINT aggregators that collect and generate CTI for the federation. The responsibility of data providers is not limited to sharing intelligence; they must also enforce strict privacy and confidentiality requirements through access control policies, anonymization techniques, encryption schemes, data minimization strategies, and policy-driven sharing agreements. The diversity among data providers means that while some operate honestly and follow defined protocols, others, described as semi-honest, comply with processes but attempt to infer additional sensitive information from shared data. There is also the risk of malicious stakeholders who may deliberately submit falsified or privacy-violating intelligence with the intention of degrading system reliability or undermining trust in the shared intelligence.

Stakeholders also act as data processors, transforming, correlating, and enriching CTI collected within their domains. Their responsibilities can include analyzing internal telemetry, taking part in distributed learning workflows, and contributing to federated inference tasks. The system accommodates honest processors who faithfully execute their duties, semi-honest actors who follow the protocol but may try to extract sensitive information, and malicious processors who might tamper with data flows, inject adversarial examples, or attempt to disrupt analysis and inference processes. The data consumer role is fulfilled by human analysts, automated detection platforms, and orchestration frameworks that query the intelligence repositories for operational monitoring or automated response. Data consumers are typically honest or semi-honest, with no ability to modify upstream data, but may attempt to extract additional insights from aggregated intelligence.

LegionITS assumes a decentralized and partially synchronous environment for federated communication. Interactions between organizations occur over a network that can be unreliable, introducing delays, message losses, duplication, or reordering. This communication model is robust against both benign faults and adversarial interference. The architecture envisions each organization as a participant with sufficient computational capacity—either through in-house resources or outsourcing—to support both local analysis and collaborative processing in the federated intelligence space. This operational assumption enables a win-win collaboration, where each participating organization can act as both a consumer and provider in a privacy-preserving manner, strengthening overall resilience against advanced threats. The federated system supports multiple autonomous organizations, each operating one or more ITS deployments at varying levels of internal granularity, such as departments, teams, or operational units. This arrangement ensures compartmentalization within the organization and supports the formation of intra-organizational federations based on trust boundaries and policy domains. Each local ITS instance manages the collection, processing, and response to CTI for its domain, while also enabling secure sharing and collaboration under policy controls that maintain data sovereignty and confidentiality.

Federated collaboration is extended beyond organizational boundaries through policy-driven interactions that define what can be shared and under which conditions, ensuring compliance with privacy, legal, and strategic requirements. These cross-domain operations, such as collaborative anomaly detection, privacy-preserving aggregation, and joint analysis, rely on distributed protocols resilient to network faults and partial participation, without

needing centralized coordination. Communication occurs over partially synchronous, unreliable networks and is abstracted from any inherent trust in the communication layer. Integrations with existing infrastructures like CTI platforms, logging systems, and response tools enable seamless federation, supported by standardized representations and exchange protocols. Behavioral assumptions from the threat model inform the system's operation, allowing a mix of cooperative and adversarial participants, but always with a focus on maintaining operational correctness, confidentiality, and resilience throughout the federated environment.

7.4. Architecture of a Federated ITS: LegionITS

The architecture of LegionITS, illustrated in Figure 7.1, exemplifies the design of a federated threat intelligence sharing system, envisaged to operate through a two-level federation structure that potentially optimizes threat intelligence sharing both within organizations (intra-organization) and among distinct organizations (inter-organization). This dual-layered approach enables compartmentalization and facilitates secure and efficient collaboration, fostering a cybersecurity ecosystem that balances autonomy with collective defense capabilities.

LegionITS is designed to integrate with modern ITSs, such as Skynet, and leverage SIEM platforms to enhance cyber defense operations. The architecture delineates roles between CTI producers and consumers, intended to support intelligence-driven security workflows. It incorporates components such as IDS with Web Application Firewalls (WAF) at Layer 7, which could provide advanced functionalities including in-depth malware inspection, binary analysis, and attribution of attacks to known threat actors like APTs. Complementarily, HIDS are expected to collect granular telemetry on system-level intrusions to improve local threat detection accuracy. User Access Management (UAM) modules are designed to contribute by monitoring authentication logs and access patterns for anomalies suggestive of adversarial behaviors, aligned with frameworks such as MITRE ATT&CK.

At the data collection and processing levels, LegionITS is anticipated to gather telemetry from various system components, including hypervisors, VMs, OSs, and firewalls. This telemetry would undergo privacy-preserving preprocessing—such as anonymization, pseudonymization, and encryption—to uphold confidentiality and support compliance with regulations like GDPR. The SIEM pipeline is expected to transform raw telemetry into actionable intelligence, with access tightly controlled through role-based permissions and

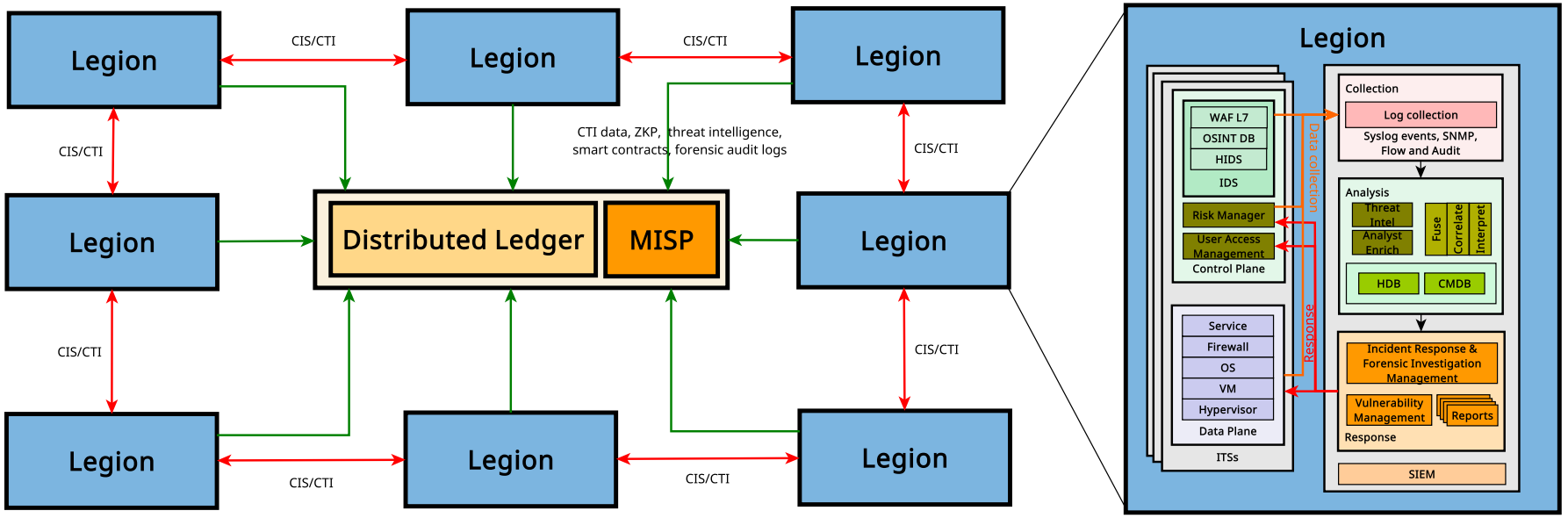

FIGURE 7.1: LegionITS's architecture, illustrating both internal and inter-organization sharing.

data retention policies enforced to manage privacy and lifecycle constraints. Enriched intelligence, leveraging databases like Hardware Databases (HDB) and Configuration Management Databases (CMDB), would inform targeted mitigation strategies by contextualizing vulnerabilities, dependencies, and attack indicators.

Key to the architecture are advanced privacy-preserving technologies that are designed for implementation. LegionITS proposes the use of MHE to perform computations directly on encrypted data, which potentially safeguards sensitive inputs during processes such as FL. DP techniques are introduced to add calibrated noise to data or models to defend against inference attacks while maintaining privacy guarantees—albeit with a trade-off in analytic accuracy. Additionally, ZKP are intended to facilitate verification of security claims (including compliance and exposure status) without revealing sensitive underlying information. Collectively, these cryptographic methods are expected to ensure secure intelligence sharing and trustworthy joint analytics, preserving confidentiality and fostering trust.

To coordinate and govern collaborative activities, LegionITS incorporates a distributed ledger, exemplified by solutions such as RepuCoin [158], which is designed to log cryptographic hashes of CTI data, ZKP verifications, encrypted intelligence, and audit trails. This ledger aims to ensure data integrity, immutability, and transparency, while supporting smart contracts that automate alerts and response triggers. Governance frameworks embedded within the architecture are meant to enforce fine-grained access controls and sharing policies, thereby facilitating compliance and accountability within the federation.

LegionITS supports envisioned operational scenarios: primarily, inter-organizational sharing empowers entities to exchange threat intelligence securely, leveraging cryptographic safeguards and anonymization to mitigate leakage of sensitive details. Secondly, intra-organizational sharing allows diverse ITSs within an organization to collaborate, maintaining strict compartmentalization to limit lateral threat propagation. Lastly, the architecture anticipates enabling sanitized intelligence to be shared publicly via the distributed ledger, expanding accessibility of actionable insights while protecting source anonymity and mitigating risks of exploitation.

Together, these components build a conceptual, robust, and scalable framework with the potential to enable effective, privacy-conscious threat intelligence exchange and coordinated cyber defense across diverse organizational environments. While LegionITS incorporates

state-of-the-art security mechanisms and compliance considerations, it also aspires to foster proactive resilience through collaborative technologies and governance structures.

7.5. Operational Scenarios and Use Cases

This section presents the concrete operational scenarios and typical use cases supported by the LegionITS architecture, highlighting how its federated, intrusion-tolerant approach promotes secure information sharing, adaptive defense, and collaborative cyber resilience in practice.

The LegionITS architecture supports a diverse range of operational scenarios and use cases that collectively demonstrate its potential to enhance cyber resilience for organizations operating both independently and as part of a broader federation. One of the core scenarios envisioned is internal intelligence sharing among multiple ITS clusters within a single organization. Organizations can deploy several ITS instances to address different operational domains, teams, or departments, each maintaining its own operational autonomy and security policies. LegionITS enables internal ITS clusters to share telemetry, threat intelligence, and security insights securely. This internal information exchange is governed by cryptographic protections and policy-controlled data flows, ensuring that no sensitive information is leaked across domains while reinforcing the security posture of the entire organization. Anonymization, data minimization, and encrypted communications are integral to this scenario, fostering robust detection and response without undermining confidentiality requirements.

Another use case supported by LegionITS is secure collaboration and sharing of CTI with trusted external partners. In this scenario, multiple independent organizations—such as enterprises, governmental bodies, or critical infrastructure operators—form a federated trust domain to strengthen their defenses against evolving cyber threats. Each participant contributes and consumes CTI while maintaining granular control over what is shared and how it is disseminated. LegionITS leverages advanced privacy-preserving technologies, including MHE, DP, ZKP, and encrypted anonymized intelligence, to ensure that only sanitized and policy-compliant information is exchanged. Automated negotiation of privacy and access control policies further enables dynamic and risk-adaptive collaboration while preventing direct or indirect exposure of sensitive operational details.

In practice, operationalization is augmented by integration with SIEM and SOAR platforms. This allows intelligence sharing, detection, and mitigation to be orchestrated via automated playbooks and policy-driven workflows. By integrating with a wide spectrum

of data sources—including HIDSs, network telemetry, and external intelligence feeds—LegionITS ensures that organizations benefit from comprehensive threat visibility. ML and advanced analytics modules allow for continuous monitoring, behavioral anomaly detection, and rapid contextualization of emerging threats. These capabilities are further strengthened by integration with attack surface management and digital risk protection tools, which extend the system's defensive horizon beyond the organizational perimeter.

The CVE-2024-0132 case study is an illustrative example that highlights LegionITS's practical strengths in addressing APTs and zero-day vulnerabilities, which traditional cybersecurity defenses often struggle to manage effectively.

This vulnerability involves a time-of-check to time-of-use (TOCTOU) race condition in the NVIDIA Container Toolkit, widely used in GPU-accelerated container environments both on-premises and in the cloud. This flaw allows a malicious container to bypass standard isolation mechanisms, potentially gaining unauthorized access to the host system's filesystem. Exploiting such a zero-day vulnerability can lead to severe consequences, including arbitrary code execution, privilege escalation, data tampering, denial of service, and exposure of sensitive information. Because this type of attack exploits previously unknown weaknesses, conventional signature-based detection methods—which rely on known attack patterns—are insufficient to recognize or mitigate these threats in real-time.

LegionITS mitigates this by leveraging internal segmentation and collaborative analytics mechanisms. Each organization's infrastructure is partitioned into multiple independent ITS instances, each managing specific operational domains or services with enforced segmentation policies. This architectural segmentation strictly isolates different ITS clusters, meaning that even if one instance is compromised by an attacker via exploiting CVE-2024-0132, the infection cannot freely propagate laterally within the organization or across federated participants. Such lateral movement prevention is critical for containing the attack's impact and confining it to the initially affected segment.

Moreover, LegionITS employs encrypted information flows secured by advanced cryptographic techniques like MHE and ZKP. These ensure that intelligence shared between ITS instances or organizations remains confidential and tamper-resistant, even under adversarial conditions. This encryption means that attackers who compromise a particular LegionITS node cannot access or manipulate intelligence data in transit or at rest to escalate privileges or deceive other federation members.

To maintain data and intelligence trustworthiness over time, LegionITS incorporates automated revocation mechanisms. These capabilities allow the federation to detect compromised or falsified intelligence contributions and efficiently revoke or sanitize them, preventing polluted or malicious data from degrading the collective knowledge base. This preserves the integrity of CTI being exchanged and ensures that subsequent detection and mitigation actions remain reliable.

As such, this case study exemplifies how LegionITS's integration of federation-based collaboration, cryptographic safeguards, and adaptive automation enhances cyber resilience both for individual organizations and interconnected systems. Through controlled and privacy-respecting intelligence exchange, LegionITS promotes proactive defenses and rapid, trustworthy responses against evolving threats like zero-day exploits and APTs. It creates a robust cybersecurity ecosystem that balances operational autonomy, data confidentiality, and collective situational awareness, addressing the complexities of modern, sophisticated threat landscapes in real-world deployments.

7.6. Privacy-Utility Trade-off in Federated Learning

LegionITS incorporates FL as a core mechanism to enable collaborative cybersecurity analytics across autonomous organizations while preserving data privacy and confidentiality. This section explores the critical trade-offs between privacy guarantees and model utility by examining the integration of DP within the FL framework.

The inclusion of the FL with DP experiment is to empirically demonstrate and quantify the privacy-utility trade-offs associated with applying privacy guarantees to federated cybersecurity analytics. While FL enables collaborative model training without sharing raw data, the addition of DP protects against sophisticated inference attacks that could potentially extract sensitive information from shared model updates. Providing experimental results on this integration validates that LegionITS can safeguard privacy without rendering the detection models impractically inaccurate.

In a federated environment, FL allows each local ITS instance to process its telemetry and compute model updates or gradients based on sensitive cybersecurity data without exposing raw data externally. These encrypted updates are securely aggregated using advanced cryptographic techniques such as MHE, maintaining confidentiality during both transmission and collective model training.

To further strengthen privacy, DP introduces controlled statistical noise to the model updates, mitigating the risk of adversarial inference attacks aimed at extracting sensitive

information about individual data points from shared parameters. LegionITS adopts DP with a privacy budget parameter $\epsilon = 1.64$ and $\delta = 10^{-5}$, which provides a moderate level of privacy protection. Lower values of ϵ denote stronger privacy but introduce more noise, potentially degrading model accuracy, whereas higher values prioritize utility over privacy. This choice of balances effective privacy safeguards with practical utility, mitigating risks such as record linkage or membership inference while maintaining sufficient model performance for operational use.

Experimental evaluation using an anomaly-detection dataset, presented in Table 7.1, demonstrates this inherent privacy-utility trade-off. The non-DP FL model achieves high accuracy (typically above 97%), rapid convergence, and robust detection performance. By contrast, the DP-enabled FL model experiences a manageable accuracy reduction—approximately 10–15% across training rounds—while still maintaining strong predictive capabilities. The F1 score and recall metrics of the FL model remain reliable and improve steadily over time, validating the practical feasibility of deploying privacy-preserving FL within LegionITS for real-world intrusion detection and threat attribution scenarios.

Round	Setting	Accuracy	F1 Score	Recall
1	Non-DP	0.9737	0.9610	0.9651
	DP	0.8211	0.7546	0.8253
2	Non-DP	0.9951	0.9934	0.9929
	DP	0.8773	0.8279	0.8942
3	Non-DP	0.9837	0.9947	0.9948
	DP	0.8811	0.8338	0.8955

Table 7.1: Federated Evaluation Metrics: Non-DP vs DP

The architecture supports fine-tuning of DP parameters, such as noise multiplier and clipping norms, enabling flexible adaptation of privacy guarantees to suit varying operational requirements and threat models. This balance ensures that LegionITS can enforce formal privacy safeguards without sacrificing the overall effectiveness of timely and accurate cybersecurity defenses.

The integration of DP-enhanced FL within LegionITS exemplifies a step towards bridging privacy with utility in federated cybersecurity analytics, enabling autonomous organizations to collaboratively improve detection accuracy while preserving the confidentiality of their sensitive operational data. This approach fosters secure, scalable, and privacy-preserving collaboration for robust collective cyber defense in a constantly evolving threat landscape.

7.7. Evaluation of Architecture Flexibility and Scalability

This section presents an evaluation of the LegionITS architecture using the Scenario-Based Architecture Analysis Method [159] to assess how well the system addresses the complex demands of modern cybersecurity operations and privacy scenarios. This is a structured technique for evaluating software architectures by identifying key usage scenarios, examining how these scenarios engage various architectural elements, and assessing the system's capability to meet both functional and non-functional requirements under real-world conditions. This method uncovers architectural strengths and weaknesses, reveals potential conflicts, and highlights areas for improvement, thereby informing design decisions and future enhancements.

By applying Scenario-Based Architecture Analysis Method with diverse and realistic scenarios, the evaluation provides a comprehensive and systematic validation of LegionITS's architectural design against its intended use cases and evolving operational needs. The assessment underscores LegionITS's modular structure, robust security features, scalability, and adaptability, demonstrating its effectiveness while also identifying possible challenges in complex, distributed federated environments.

The key scenarios analyzed include integrating a new internal threat intelligence source, sharing anonymized threat intelligence with trusted external partners, detecting and responding to zero-day vulnerabilities using FL, replacing the default distributed ledger, extending support for emerging privacy-preserving technologies, scaling to accommodate a tenfold increase in participating organizations, managing large-scale security incidents, and integrating with external threat intelligence platforms.

This evaluation highlights LegionITS's capacity to maintain organizational autonomy alongside fostering collective defense, ensuring resilient, efficient, and privacy-aware collaboration among multiple cybersecurity stakeholders.

Integrating a new internal threat intelligence source: Integrating new internal threat intelligence inputs, such as additional HIDS, can bolster an organization's cybersecurity framework. The design of the LegionITS architecture emphasizes modularity in data collection and seamless compatibility with SIEM platforms to simplify the integration of new data sources.

When deploying a new HIDS, developers create specific connectors or adapters to link with existing telemetry gathering systems, thereby guaranteeing smooth data transfer and

compatibility. Once operational, the collected telemetry undergoes privacy-centric processing steps, including anonymization and pseudonymization, aligned with the organization's privacy policies and regulatory mandates.

The processed data is subsequently standardized and aligned with established schemas such as STIX and TAXII. This enables effective incorporation into analytics workflows and threat detection pipelines without impacting established baseline models or detection processes. Integration often requires thorough validation of data formats, parser updates, and comprehensive testing to prevent inconsistencies or negative performance impacts.

From a deployment standpoint, integrations are managed via careful configuration tuning and phased rollout plans, which support controlled implementation and continual assessment of effects on system performance and threat detection efficacy. Automated surveillance systems monitor the health and throughput of these new integration points, flagging any irregularities.

In scenarios where telemetry volume grows substantially, system resources and load balancing mechanisms are adjusted accordingly to uphold optimal performance and system stability. Importantly, this approach typically avoids modifications to the fundamental architectural components, aside from ensuring telemetry data formats conform to existing standards and accommodating any increased processing needs.

Due to LegionITS's modular and scalable architecture, organizations can swiftly adopt new detection technologies and integrate new intelligence sources without disrupting operational continuity or compromising system reliability.

Sharing anonymized threat intelligence with trusted external partners: Facilitating the secure and privacy-conscious exchange of threat intelligence with trusted external partners is a necessary component for effective federated cybersecurity collaboration. The LegionITS framework employs a dual-layer federation model that distinctly separates internal and external data sharing processes. This architecture promotes the empowerment of organizations with detailed control over the scope and recipients of shared data.

To safeguard sensitive information, the system integrates advanced cryptographic techniques such as MHE, DP, and ZKP. These technologies ensure that intelligence disseminated outside the trusted boundary remains both anonymized and encrypted, effectively preventing unintended data exposure while enabling the sharing of actionable threat indicators, attacker tactics, and vulnerability details.

The onboarding of a new trusted partner involves a series of well-coordinated steps. Threat intelligence is exchanged through standardized protocols and data formats supported by instances of the MISP, promoting automated, structured, and interoperable information flow. Privacy configurations are customized according to the trust level and specific security requirements of each partnership, including options to limit data granularity, enhance anonymization of sensitive attributes, and enforce rigorous access controls.

Operationally, incorporating a new partner is designed to be effortless and minimally disruptive. Typically, this involves adjustments within existing MISP configurations and privacy modules, without requiring fundamental changes to the core system design. Continuous monitoring and auditing tools are implemented to oversee data exchanges, ensuring compliance with agreed sharing policies and maintaining governance visibility.

Through this approach, LegionITS supports flexible, policy-driven collaboration with a broad spectrum of external parties while upholding stringent security and privacy standards.

Using FL to Detect and Address Zero-Day Vulnerabilities: Rapid detection and mitigation of zero-day vulnerabilities are pivotal in modern threat intelligence, where the window for countermeasures might be minimal. The LegionITS architecture applies FL combined with privacy-enhancing analytics to enable multiple organizations to collaboratively build detection models without sharing sensitive raw data. This collaborative approach allows each participant to contribute their local intelligence—such as telemetry feeds, behavioral signals, and incident data—while safeguarding proprietary and regulated information.

Operationally, individual ITS instances locally analyze telemetry and generate encrypted model gradients or updates. These updates are securely combined into a unified global model using advanced cryptographic safeguards, notably MHE, preserving confidentiality during transmission and aggregation. Furthermore, DP techniques can be employed to mask individual contributions further, reducing the potential for data leakage or inference attacks.

LegionITS's architecture integrates analytics, risk management, and privacy functions specifically tailored for such cooperative model training. Automation modules orchestrate workflow coordination, authenticate participants, and enforce regulatory and policy compliance. The system supports flexible federation management, allowing dynamic addition or

removal of organizations in response to changes in readiness, threat landscape, or resource constraints.

Throughout the federation lifecycle, extensive monitoring and validation mechanisms verify model convergence, identify irregularities, and maintain collaborative integrity. Upon reaching a stable state, the global detection model is swiftly disseminated to all members, enabling near real-time detection and response to emergent zero-day threats. This capability exemplifies LegionITS's strength in supporting efficient, privacy-conscious collaboration for threat intelligence sharing and mitigation across complex multi-organizational environments.

Replacing the default distributed ledger: LegionITS's design exhibits significant flexibility, especially when there is a need to switch its default distributed ledger to an alternative blockchain platform. The distributed ledger provides a critical function in the system by securely storing cryptographic hashes related to threat intelligence data, audit records, and smart contracts, thereby assuring data integrity, transparency, and resistance to tampering.

When LegionITS's interaction with the ledger is architected through a clearly defined abstraction layer (referenced in the Section 7.4), migrating to a new blockchain primarily involves updating the integration interfaces (APIs) and confirming compatibility with the new blockchain's protocols. This modular design approach limits the scope of changes solely to the integration layer, greatly reducing the risk of widespread system disruption.

Conversely, should the ledger interface be directly bound to specific features or data formats of the existing blockchain, transitioning may require broader updates encompassing several components, introducing greater migration complexity and potential risks. During such transitions, it is possible to securely export historical data—including cryptographic hashes and audit logs—from the incumbent ledger and import these into the new blockchain environment. Stringent validation processes ensure that all records remain complete and unaltered.

This scenario highlights the critical importance of modular interfaces and architectural abstraction for facilitating foundational technology upgrades with minimal systemic impact. Moreover, thorough planning and execution of migration procedures are essential to preserve data accuracy, accessibility, and regulatory compliance, maintaining audit readiness throughout the lifecycle.

Extending support for emerging privacy-preserving technologies: As cryptographic standards progress, organizations utilizing LegionITS may face the need to incorporate novel privacy-enhancing technologies, including post-quantum cryptography, into their security infrastructures. Such integration involves more than merely swapping cryptographic algorithms; it demands comprehensive updates to data exchange protocols, revisions to key management frameworks, and adaptations to privacy workflows to accommodate new operational requirements introduced by advanced cryptographic techniques.

LegionITS's cryptographic architecture is purposefully modular to minimize disruptions during these upgrades. This modularity enables targeted adjustments limited to cryptographic components without requiring a full overhaul of the entire system. However, advanced cryptographic methods, including systems engineered with resistance to quantum attacks, often bring considerable resource demands due to larger key dimensions and computational complexity. These factors can impact system latency and throughput, particularly when real-time threat intelligence sharing is a necessity.

Adoption across federated environments introduces further challenges as multiple organizations must synchronize their transition to new standards. Achieving seamless interoperability may require gradual rollouts, temporary maintenance of legacy cryptosystems, or protocol negotiation mechanisms enabling dynamic selection of cryptographic methods based on the capabilities of counterpart systems.

On the operational front, incorporating new cryptographic primitives requires rigorous validation to maintain enduring privacy and security guarantees. This encompasses exhaustive testing of data flows, verification of protocol correctness, and reevaluation of threat models to identify newly introduced vulnerabilities. Additionally, documentation updates, personnel training, and revisions of compliance and audit policies are essential steps to align with evolving cryptographic frameworks.

By adopting a flexible and modular cryptographic environment, LegionITS promotes the integration of emerging privacy-preserving innovations while managing the technical and organizational complexities inherent to these transitions.

Scaling to accommodate a tenfold increase in participating organizations: Expanding LegionITS's capacity to support a tenfold increase in participating entities — such as during nationwide or industry-wide rollouts — introduces a host of architectural and

operational complexities. The federation management layer must adapt to oversee a substantially larger and more diverse membership, each potentially guided by distinct policies, network environments, and regulatory demands. This growth amplifies the processing burden on data aggregation systems, which must efficiently handle and correlate significantly increased volumes of telemetry, alerts, and threat intelligence in near real-time.

To address potential bottlenecks, the system's data flows may require restructuring to promote greater parallelism, while aggregation points could benefit from techniques such as sharding or horizontal scaling. The underlying distributed ledger, which guarantees data integrity and transparency, must also be upgraded to accommodate heightened transaction rates and expanded audit logs. This might involve enhancements to consensus algorithms, enlarging block capacities, or deploying additional ledger nodes to maintain performance and fault tolerance.

The federation management tools and interfaces need to be enhanced to provide operators with advanced oversight capabilities, streamlined onboarding functions, adaptable policy enforcement, and in-depth monitoring across the enlarged federation. FL models underpinning collaborative analytic and detection efforts could confront limitations related to increased communication overhead, prolonged training durations, and synchronization challenges. Approaches such as hierarchical federation structures, asynchronous or decentralized learning protocols, and model compression can help alleviate these issues.

Effectively scaling the system demands forward-looking planning — optimizing storage solutions to handle voluminous datasets, implementing intelligent load balancing, and reinforcing federation management frameworks to support dynamic participant changes. Continuous performance assessments, rigorous stress testing, and architectural refinements will be pivotal to ensuring LegionITS sustains robust performance, reliability, and security as it scales to meet the needs of an expanded federation.

Managing large-scale security incidents: In the event of a large-scale security incident, such as a coordinated cyberattack affecting multiple organizations within a federation, LegionITS is designed to facilitate swift and coordinated responses while ensuring the integrity and confidentiality of data. Its architecture supports live threat intelligence feeds, delivering timely updates on IOCs, attack vectors, and recommended countermeasures to all impacted members.

Automated alerting and correlation systems help map the scale and progression of the attack, enabling security teams to prioritize and align their response actions effectively. Decision-making processes at the federation level are dynamically adapted to promote information sharing and collective defense, with privacy-preserving analytic techniques and cryptographic safeguards preserving the confidentiality of sensitive information.

LegionITS enables rapid dissemination of remediation strategies, patch distribution, and forensic evidence, leveraging a distributed ledger to log events in an immutable and transparent manner. Operational features include incident response playbooks and automated workflows designed to be triggered for specific attack scenarios, encompassing actions such as isolating compromised systems, revoking affected credentials, and implementing urgent patching or configuration changes.

Throughout the incident lifecycle, continuous monitoring and detailed forensic logging are maintained to support post-incident reviews and compliance reporting. This scenario tests LegionITS's ability to function effectively under high-pressure conditions, demanding seamless coordination, robust automation, and expedited action across the federation. The system balances the urgency of response with strict adherence to privacy and security requirements, ensuring collaborative defense does not compromise organizational confidentiality or regulatory compliance.

Integrating with external threat intelligence platforms: LegionITS enhances its threat detection capabilities by incorporating external intelligence sources such as commercial feeds, industry Information Sharing and Analysis Centers (ISACs), and public CTI repositories. Its architecture includes a secure integration layer designed to ingest a wide range of external data streams, normalizing and mapping them into internal schemas to enable seamless analysis and correlation with local telemetry.

To ensure security, every external feed is rigorously validated, filtered, and sanitized before inclusion in the intelligence pipeline. Privacy and data protection are strictly maintained, with sensitive or proprietary details being redacted or pseudonymized in accordance with organizational policies and legal requirements. The system allows configurable trust settings, enabling tailored integration strategies aligned with the reliability and relevance of each external source.

Interoperability is supported through adherence to standardized data formats and APIs, such as STIX and TAXII, enabling automated and real-time exchange of threat indicators,

signatures, and contextual information. The integration framework also facilitates dynamic onboarding and removal of data sources, allowing organizations to quickly adapt to emerging threats or engage with new intelligence providers. This approach ensures LegionITS maintains access to current and comprehensive threat intelligence, bolstering proactive defense and situational awareness.

7.8. Discussion

The LegionITS architecture presents an effort to integrate federated principles into ITSs, aiming to balance individual organizational autonomy with the potential benefits of collective cyber defense. Recognizing the complexity and evolving nature of modern cybersecurity threats, particularly in multi-stakeholder environments, LegionITS proposes a modular and privacy-conscious framework designed to support secure and scalable information sharing among heterogeneous ITS instances.

From a design perspective, the layered federation model—encompassing both intra- and inter-organizational domains—is a purposeful choice to contain risk through operational segmentation while enabling collaboration. This approach acknowledges the practical realities of organizational compartmentalization and heterogeneous security policies, seeking to enable controlled information exchange without compromising local defenses or data sovereignty. Key architectural elements such as privacy-preserving cryptographic techniques (including MHE, DP, and ZKP) and distributed ledger technologies are incorporated to enhance trust, data integrity, and compliance with regulatory requirements. These components are intended not as universal solutions but as foundational tools that require careful integration and ongoing evaluation.

Operationally, managing information flow within and between organizations involves delicate trade-offs. LegionITS's emphasis on cryptographically protected communication channels, fine-grained access controls, and data sanitization pipelines reflects an understanding of the dual imperatives to maximize threat intelligence utility while preserving privacy and reducing attack surfaces. Nonetheless, practical implementation challenges remain, including coordinating diverse detection capabilities, reconciling differences in data granularity, and ensuring robust mechanisms to detect and mitigate false or malicious intelligence—areas that require continued research and refinement.

Considering the type of shared data, LegionITS focuses on exchanging IOCs and associated contextual metadata that can be processed to minimize disclosure risks. Employing privacy-preserving transformations and anonymization techniques, the design seeks to

strike a balance where shared information retains meaningful operational value without revealing unnecessary internal details. This approach reflects awareness of the vulnerabilities introduced by overly detailed data sharing and attempts to balance the trade-off between transparency and confidentiality inherent in collaborative cybersecurity. Highlighting the benefits, newly acquired intelligence through the federation has the potential to enhance an entity's situational awareness and improve detection accuracy—particularly invaluable when confronting novel or zero-day threats. By correlating external indicators with internal telemetry, organizations can prioritize response and adapt defenses. At the systemic level, LegionITS aspires to facilitate a continuous learning dynamic where accumulated shared knowledge incrementally raises the collective resilience of interconnected ITS. Nonetheless, realizing these benefits depends on the reliable validation, revocation, and integration of shared intelligence, which are challenging tasks given adversarial tactics and evolving threat landscapes.

LegionITS's modular and extensible design creates opportunities to incorporate advances in cryptographic methods, adaptive federation governance, and automated intelligence quality assurance. The architecture provides a conceptual foundation, but practical deployment will require iterative refinement informed by empirical evaluations, stakeholder feedback, and emerging threat trends. There is a necessity for flexible and scalable orchestration frameworks capable of managing diverse participants, enforcing dynamic policies, and addressing computational resource demands. Additionally, advancing standardized interoperability and encouraging broader community collaboration are expected to be essential factors for the sustained effectiveness and adoption of such systems.

LegionITS proposes a thoughtfully constructed solution framework that seeks to advance federated collaboration within intrusion-tolerant cybersecurity ecosystems. By transparently addressing challenges related to federation integration, secure information sharing, privacy-respecting data types, and mutually reinforcing benefits, LegionITS contributes towards resilient, privacy-aware collective defense in increasingly complex operational environments.

7.9. Chapter Summary

The chapter presents LegionITS, a novel federated architecture designed to enhance cyber resilience through secure, privacy-preserving collaboration among autonomous organizations ITSs. It addresses the limitations of conventional cybersecurity platforms by enabling

scalable, policy-driven sharing of threat intelligence safeguarded by advanced cryptographic techniques such as MHE, DP, and ZKP.

LegionITS balances organizational autonomy with collective defense, employing a dual-layer federation model for intra- and inter-organizational information exchange that upholds data sovereignty and confidentiality. Practical integration with existing SIEM, SOAR, and open threat intelligence platforms supports automated detection, analysis, and response workflows, including handling zero-day vulnerabilities and advanced persistent threats.

The architecture's modular design facilitates extensibility, scalability, and adaptability to emerging privacy technologies and expanding federation memberships. Experimental evaluation of FL within this framework demonstrates a manageable trade-off between privacy guarantees and detection accuracy. Overall, LegionITS offers a foundation for next-generation collaborative cybersecurity, promoting resilient, privacy-aware collective defense across heterogeneous environments while addressing operational, privacy, and trust challenges inherent to federated intelligence sharing.

8. Conclusion

The thesis demonstrates that integrating concepts from cybersecurity, AI, and distributed systems enables the design of a more advanced architecture, Skynet, for resilient and adaptive defense against evolving cyber threats. It advances the state of the art in ITSs and privacy-preserving collaborative cybersecurity through an exploration of BFT protocols, multidisciplinary architectural principles, ML-driven risk analysis, AI-powered predictive actions, dynamic orchestration, and federated threat intelligence sharing.

At its core, this work embraces a fundamental paradigm shift in cybersecurity—from constructing impenetrable perimeters to accepting that breaches, including APTs, are inevitable. Consequently, the primary objective becomes ensuring operation under compromise, employing redundancy, diversity, and dynamism to guarantee continuous functionality and security even amid sustained attacks.

Overall, this thesis confirms that a cross-disciplinary, integrated approach leveraging distributed systems, AI, and cybersecurity fosters resilient, self-healing, and scalable ITSs. The designs provided by Skynet, HAL, EVSOAR, and LegionITS offer blueprints for operational deployments across critical infrastructure, cloud services, and automotive, supporting both centralized and federated cyber defense models.

8.1. Contributions

This section summarises the primary contributions of this thesis, each introduced in Section 1.2 and disseminated through peer-reviewed publications.

- **Chapter 3 — From BFT to ITS:** an in-depth taxonomy of deterministic and probabilistic BFT protocols, addressing their trade-offs and deployment implications. This comparative analysis informs the design choices underlying Skynet's architecture.
- **Chapter 4 — Skynet:** a modular, multi-plane architecture explicitly separating a trusted control plane from an untrusted execution plane, leveraging hardware-protected environments such as TEEs to safeguard critical functions. Its key innovations are:
    1. automated data collection and enrichment via the VExHK module;
    2. risk-aware, ML-driven decision-making in the HAL risk manager;
    3. adaptive, policy-driven orchestration and dynamic configuration;

4. integrated detection, monitoring, and forensic logging subsystems;
5. a closed feedback loop spanning planes and federated interfaces enabling collective defense.

- **Chapter 5 — HAL:** reduces exposure windows by automating vulnerability data ingestion, leveraging ML for predictive CVSS scoring, and applying unsupervised clustering to identify related vulnerabilities, enabling more timely configuration recommendations than traditional risk-assessment approaches.
- **Chapter 6 — EVSOAR:** applies SOAR principles to EV security, exploiting the CSN for reliable, low-latency, privacy-preserving orchestration and adopting FL at the edge, showcasing how SOAR can be deployed in constrained environments while extending Skynet's defense capability.
- **Chapter 7 — LegionITS:** a federated ITS enabling privacy-preserving sharing of CTI and CIS across organisations, integrating advanced cryptographic techniques with FL to balance organisational autonomy, data confidentiality, and collective resilience. Empirical evaluation confirms that, even under the privacy constraints inherent in FL, LegionITS provides analytics without compromising data privacy, making it suitable for large-scale, decentralised deployments.

8.2. Future Research Directions

While this dissertation addresses several key challenges in dynamic and adaptive intrusion tolerance, it also opens avenues for further investigation. Despite the advances above, challenges remain:

1. the reliability and completeness of public threat intelligence sources can delay detection and response;
2. the robustness of ML-based risk managers against adversarial manipulation or poisoned data is an ongoing concern;
3. there are critical operational trade-offs between automation and human oversight in adaptive, mission-critical systems;
4. the need to withstand evolving adversarial tactics, including attacks targeting learning and decision-making processes;

5. the complexity of deploying privacy-preserving collaborative frameworks at scale while maintaining performance and compliance.

This dissertation reframes intrusion tolerance as a principle of sustained, trustworthy operation under compromise, through an enduring alignment of distributed-systems rigor with cybersecurity and AI-driven adaptability. The proposed architectures and experimental results pave the way for expanded research, broader deployment, and enhanced assurances in next-generation cyber defense—a dynamic, adaptive, and multidisciplinary strategy anchored in BFT, powered by AI, and enabled by federated privacy-preserving collaboration.

References

[1] Michael J Fischer. The consensus problem in unreliable distributed systems (a brief survey). In *International Conference on Fundamentals of Computation Theory*, pages 127–140. Springer, 1983. doi:10.1007/3-540-12689-9_99.

[2] Paul Bachmann. *Zahlentheorie: Die analytische zahlentheorie*, volume 2. Teubner, 1894. doi:10.1007/978-3-322-85524-4_4.

[3] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography: Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006. Proceedings 3*, pages 265–284. Springer, 2006. doi:10.1007/11681878_14.

[4] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. Pmlr, 2017.

[5] Alysson Neves Bessani, Paulo Sousa, Miguel Correia, Nuno Ferreira Neves, and Paulo Verissimo. The CRUTIAL way of critical infrastructure protection. *IEEE Security & Privacy*, 6(6):44–51, 2008. doi:10.1109/msp.2008.158.

[6] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *forty-first annual ACM symposium on Theory of computing*, pages 169–178, 2009. doi:10.1145/1536414.1536440.

[7] Leslie Lamport. The implementation of reliable distributed multiprocess systems. *Computer Networks (1976)*, 2(2):95–114, 1978. doi:10.1016/0376-5075(78)90045-4.

[8] Yves Deswarte, Laurent Blain, and Jean-Charles Fabre. Intrusion tolerance in distributed computing systems. In *Proceedings. 1991 IEEE Computer Society Symposium on Research in Security and Privacy*, pages 110–110. IEEE Computer Society, 1991. doi:10.1109/risp.1991.130780.

[9] Paulo Esteves Verissimo, Nuno Ferreira Neves, and Miguel Pupo Correia. Intrusion-Tolerant Architectures: Concepts and Design. *Edited by G. Goos, J. Hartmanis, and J. van Leeuwen*, page 3, 2003. doi:10.1007/3-540-45177-3_1.

[10] Dan McWhorter. APT1: Exposing one of China's cyber espionage units. *Mandiant. com*, 18, 2013.

[11] Miguel Castro and Barbara Liskov. Practical Byzantine Fault Tolerance. In *Third Symposium on Operating Systems Design and Implementation*, OSDI '99, page 173–186, USA, 1999. USENIX Association. doi:10.5555/296806.296824.

[12] Ralph Langner. Stuxnet: Dissecting a cyberwarfare weapon. *IEEE security & privacy*, 9(3):49–51, 2011. doi:10.1109/msp.2011.67.

[13] Gary Adkins. Red teaming the red team: Utilizing cyber espionage to combat terrorism. *Journal of Strategic Security*, 6(3):1–9, 2013. doi:10.5038/1944-0472.6.3s.1.

[14] Caitlin Kenny. The Equifax data breach and the resulting legal recourse. *Brook. J. Corp. Fin. & Com. L.*, 13:215, 2018.

[15] Miguel Correia, Nuno Ferreira Neves, Lau Cheuk Lung, and Paulo Veríssimo. WORM-IT–a wormhole-based intrusion-tolerant group communication system. *Journal of Systems and Software*, 80(2):178–197, 2007. doi:10.1016/j.jss.2006.03.034.

[16] Johannes Behl, Tobias Distler, and Rüdiger Kapitza. Hybrids on Steroids: SGX-Based High Performance BFT. In *Twelfth European Conference on Computer Systems*, EuroSys '17, pages 222–237, New York, NY, USA, 2017. Association for Computing Machinery. doi:10.1145/3064176.3064213.

[17] Michael Ben-Or, Boaz Kelmer, and Tal Rabin. Asynchronous secure computations with optimal resilience. In *Thirteenth annual ACM symposium on Principles of distributed computing*, pages 183–192, 1994. doi:10.1145/197917.198088.

[18] The Wall Street Journal. Cyber Companies Stress AI as Core Future Technology. *The Wall Street Journal*, 2024. URL: https://www.wsj.com/articles/cyber-companies-stress-ai-as-core-future-technology-6944ae93.

[19] Financial Times. European Defense Companies See Boost in Demand Amid Geopolitical Tensions. *Financial Times*, 2024. URL: https://www.ft.com/content/f6119e99-04d9-4d37-8d06-e226aeecf88a.

[20] Lawrence A. Gordon and Martin P. Loeb and Lei Zhou. Integrating cost–benefit analysis into the NIST Cybersecurity Framework via the Gordon–Loeb Model. *Journal of Cybersecurity*, 6(1), 2020. doi:10.1093/cybsec/tyaa005.

[21] Rafael R Obelheiro, Alysson Neves Bessani, Lau Cheuk Lung, and Miguel Correia. How practical are intrusion-tolerant distributed systems? 2006.

[22] Rajat Verma, Vipin Gautam, Chandra Prakash Yadav, Ishu Gupta, and Ashutosh Kumar Singh. A survey on data leakage detection and prevention. In *International Conference on Innovative Computing & Communications (ICICC)*, 2020. doi:10.55248/gengpi.5.0324.0701.

[23] Jukka Ruohonen. A look at the time delays in CVSS vulnerability scoring. *Applied Computing and Informatics*, 15(2):129–135, 2019. doi:10.1016/j.aci.2017.12.002.

[24] FIRST. Common Vulnerability Scoring System v3.1: Specification Document. https://www.first.org/cvss/specification-document, 2019. "[Online; accessed 23-March-2023]".

[25] Miguel Garcia, Alysson Bessani, and Nuno Neves. Lazarus: Automatic management of diversity in bft systems. In *20th International Middleware Conference*, pages 241–254, 2019. doi:10.1145/3361525.3361550.

[26] Jay Jacobs, Sasha Romanosky, Benjamin Edwards, Idris Adjerid, and Michael Roytman. Exploit prediction scoring system (epss). *Digital Threats: Research and Practice*, 2(3):1–17, 2021. doi:10.1145/3436242.

[27] Seondong Heo, Soojin Lee, Bumsoon Jang, and Hyunsoo Yoon. Designing and implementing a diversity policy for intrusion-tolerant systems. *IEICE TRANSACTIONS on Information and Systems*, 100(1):118–129, 2017. doi:10.1587/transinf.2015edp7478.

[28] Atefeh Khazaei, Mohammad Ghasemzadeh, and Vali Derhami. An automatic method for CVSS score prediction using vulnerabilities description. *Journal of Intelligent & Fuzzy Systems*, 30(1):89–96, 2016. doi:10.3233/ifs-151733.

[29] Ali Shoker, Fernando Alves, and Paulo Esteves-Verissimo. ScaLOTA: Scalable Secure Over-the-Air Software Updates for Vehicles. In *2023 42nd International SRDS*, pages 151–161. IEEE, 2023. doi:10.1109/srds60354.2023.00024.

[30] Erick Silva, Tadeu Freitas, Rehana Yasmin, Ali Shoker, and Paulo Esteves-Verissimo. Evolve: A value-added services platform for electric vehicle charging stations. In *2025*

IEEE 101st Vehicular Technology Conference (VTC2025-Spring), pages 1–7, 2025. doi:10.1109/VTC2025-Spring65109.2025.11174448.

[31] Priscilla Koepke. Cybersecurity information sharing incentives and barriers. *Sloan School of Management at MIT University: Cambridge, MA, USA*, 2017.

[32] Ali Pala and Jun Zhuang. Information sharing in cybersecurity: A review. *Decision Analysis*, 16(3):172–196, 2019. doi:10.1287/deca.2018.0387.

[33] Vodafone Portugal. Cyberattack on Vodafone Portugal, February 2022. "[Online; accessed 07-September-2022]". URL: https://www.vodafone.pt/en/press-releases/2022/2/cyberattack-on-vodafone-portugal.html.

[34] Google. Bug Hunters. https://bughunters.google.com/. "[Online; accessed 06-March-2025]".

[35] Meta Platforms. Meta Bug Bounty. https://bugbounty.meta.com/. "[Online; accessed 06-March-2025]".

[36] Microsoft. Microsoft Security Response Center Bug Bounty. https://www.microsoft.com/en-us/msrc/bounty?rtc=1. "[Online; accessed 25-February-2023]".

[37] Joni Fraga and David Powell. A fault-and intrusion-tolerant file system. In *3rd international conference on computer security*, volume 203, pages 2–s2, 1985.

[38] David Powell, RJe Stroud, A Adelsbach, D Alessandri, C Cachin, S Creese, M Dacier, Y Deswarte, K Kursawe, JC Laprie, et al. Conceptual model and architecture. *Department of Computing Science Technical Report Series*, 2001.

[39] Paulo Verıssimo and N Ferreira Neves. Service and Protocol Architecture for the MAFTIA Middleware. *Deliverable D23, Project MAFTIA IST-1999-11583*, 2001.

[40] Paulo Veríssimo, Nuno Ferreira Neves, C Cachin, JA Poritz, David Powell, Y Deswarte, Robert J Stroud, and IS Welch. Intrusion-tolerant middleware: The MAFTIA approach. 2004.

[41] Jaynarayan H Lala. Foundations of Intrusion Tolerant Systems. In *Foundations of Intrusion Tolerant Systems, 2003 [Organically Assured and Survivable Information Systems]*, pages i–vii, 2003. doi:10.1109/FITS.2003.1264922.

[42] Miguel Correia, Paulo Veríssimo, and Nuno Ferreira Neves. The design of a COTS real-time distributed security kernel. In *European Dependable Computing Conference*, pages 234–252. Springer, 2002. doi:10.1007/3-540-36080-8_21.

[43] Marshall D Abrams, Sushil G Jajodia, and Harold J Podell. *Information security: an integrated collection of essays*. IEEE computer society press, 1995.

[44] P Pal, M Atighetchi, C Jones, I Keidar, D Levin, J Loyall, P Rubel, R Schantz, R Watro, and F Webber. Intrusion tolerance by unpredictable adaptation (ITUA). *Tech. rep., BBNT SOLUTIONS LLC CAMBRIDGE MA*, 2005. doi:10.21236/ada433567.

[45] Martine Couturier and Edith Wilkinson. Open advanced system for improved crisis management (OASIS). In *2nd International Information Systems for Crisis Response and Management (ISCRAM) Conference*, 2005.

[46] António Casimiro, Paulo Verissimo, Diego Kreutz, Filipe Araujo, Raul Barbosa, Samuel Neves, Bruno Sousa, Marilia Curado, Carlos Silva, Rajeev Gandhi, et al. Trone: Trustworthy and resilient operations in a network environment. In *IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN 2012)*, pages 1–6. IEEE, 2012. doi:10.1109/dsnw.2012.6264694.

[47] Paulo Veríssimo, Nuno Ferreira Neves, and Miguel Correia. CRUTIAL: The blueprint of a reference critical information infrastructure architecture. In *International Workshop on Critical Information Infrastructures Security*, pages 1–14. Springer, 2006. doi:10.1007/11962977_1.

[48] David Sames, Brian Matt, Brian Niebuhr, Gregg Tally, Brent Whitmore, and David Bakken. Developing a heterogeneous intrusion tolerant CORBA system. In *International Conference on Dependable Systems and Networks*, pages 239–248. IEEE, 2002. doi:10.1109/fits.2003.1264945.

[49] Steve Vinoski. Distributed object computing with CORBA. *C++ Report*, 5(6):32–38, 1993.

[50] Hans P Reiser and Rüdiger Kapitza. Vm-fit: Supporting intrusion tolerance with virtualisation technology. In *Workshop on Recent Advances on Intrusion-Tolerant Systems*, 2007.

[51] Paulo Sousa, Alysson Neves Bessani, Miguel Correia, Nuno Ferreira Neves, and Paulo Verissimo. Resilient intrusion tolerance through proactive and reactive recovery. In *13th Pacific Rim International Symposium on Dependable Computing (PRDC 2007)*, pages 373–380. IEEE, 2007. doi:10.1109/prdc.2007.52.

[52] Paulo Sousa, Alysson Neves Bessani, and Rafael R Obelheiro. The FOREVER service for fault/intrusion removal. In *2nd workshop on Recent advances on intrusiton-tolerant systems*, pages 1–6, 2008. doi:10.1145/1413901.1413906.

[53] Yih Huang and Arun Sood. Self-cleansing systems for intrusion containment. In *Workshop on self-healing, adaptive, and self-managed systems (SHAMAN)*, 2002.

[54] Lidong Zhou, Fred B Schneider, and Robbert Van Renesse. Coca: A secure distributed online certification authority. *ACM Transactions on Computer Systems (TOCS)*, 20(4):329–368, 2002. doi:10.1109/fits.2003.1264932.

[55] Amy Babay, Thomas Tantillo, Trevor Aron, Marco Platania, and Yair Amir. Network-attack-resilient intrusion-tolerant SCADA for the power grid. In *2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pages 255–266. IEEE, 2018. doi:10.1109/dsn.2018.00036.

[56] Yair Amir, Brian Coan, Jonathan Kirsch, and John Lane. Byzantine replication under attack. In *2008 IEEE International Conference on Dependable Systems and Networks With FTCS and DCC (DSN)*, pages 197–206. IEEE, 2008. doi:10.1109/dsn.2008.4630088.

[57] Paulo Sousa, Alysson N Bessani, Wagner S Dantas, Fábio Souto, Miguel Correia, and Nuno F Neves. Intrusion-tolerant self-healing devices for critical infrastructure protection. In *2009 IEEE/IFIP International Conference on Dependable Systems & Networks*, pages 217–222. IEEE, 2009. doi:10.1109/dsn.2009.5270333.

[58] Stephanie Bryant and Feiyi Wang. Aspects of adaptive reconfiguration in a scalable intrusion tolerant system. *Complexity*, 9(2):74–83, 2003. doi:10.1002/cplx.20007.

[59] Dick O'Brien, Rick Smith, Tammy Kappel, and Clint Bitzer. Intrusion tolerance via network layer controls. In *DARPA Information Survivability Conference and Exposition*, volume 1, pages 90–96. IEEE, 2003. doi:10.1109/discex.2003.1194875.

[60] Kim Hammar and Rolf Stadler. Intrusion tolerance for networked systems through two-level feedback control. In *2024 54th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pages 338–352. IEEE, 2024. doi:10.1109/dsn58291.2024.00042.

[61] James C Corbett, Jeffrey Dean, Michael Epstein, Andrew Fikes, Christopher Frost, Jeffrey John Furman, Sanjay Ghemawat, Andrey Gubarev, Christopher Heiser, Peter Hochschild, et al. Spanner: Google's globally distributed database. *ACM Transactions on Computer Systems (TOCS)*, 31(3):1–22, 2013. doi:10.1145/2491245.

[62] Apache Software Foundation. *ZooKeeper Administrator's Guide*. Apache ZooKeeper Documentation, Release 3.9.3, 2019. "[Online; accessed 07-June-2024]". URL: https://zookeeper.apache.org/doc/r3.9.3/zookeeperAdmin.html.

[63] Michael J Fischer, Nancy A Lynch, and Michael S Paterson. Impossibility of distributed consensus with one faulty process. *Journal of the ACM (JACM)*, 32(2):374–382, 1985. doi:10.1145/3149.214121.

[64] T. Freitas, J. Soares, M.E. Correia, and R. Martins. Deterministic or probabilistic? - A survey on Byzantine fault tolerant state machine replication. *Computers and Security*, 129, 2023. doi:10.1016/j.cose.2023.103200.

[65] Ramakrishna Kotla, Lorenzo Alvisi, Mike Dahlin, Allen Clement, and Edmund Wong. Zyzzyva: Speculative Byzantine Fault Tolerance. *ACM Trans. Comput. Syst.*, 27(4), jan 2010. doi:10.1145/1658357.1658358.

[66] Giuliana Santos Veronese, Miguel Correia, Alysson Neves Bessani, Lau Cheuk Lung, and Paulo Verissimo. Efficient Byzantine Fault-Tolerance. *IEEE Transactions on Computers*, 62(1):16–30, 2013. doi:10.1109/TC.2011.221.

[67] Rüdiger Kapitza, Johannes Behl, Christian Cachin, Tobias Distler, Simon Kuhnle, Seyed Vahid Mohammadi, Wolfgang Schröder-Preikschat, and Klaus Stengel. CheapBFT: Resource-Efficient Byzantine Fault Tolerance. In *7th ACM European Conference on Computer Systems*, EuroSys '12, page 295–308, New York, NY, USA, 2012. Association for Computing Machinery. doi:10.1145/2168836.2168866.

[68] Zarko Milosevic, Martin Biely, and André Schiper. Bounded Delay in Byzantine-Tolerant State Machine Replication. In *2013 IEEE 32nd International Symposium on Reliable Distributed Systems*, pages 61–70, 2013. doi:10.1109/SRDS.2013.15.

[69] João Sousa and Alysson Bessani. From byzantine consensus to bft state machine replication: A latency-optimal transformation. In *2012 Ninth European Dependable Computing Conference*, pages 37–48, 2012. doi:10.1109/EDCC.2012.32.

[70] C. Cachin and J.A. Poritz. Secure INtrusion-Tolerant Replication on the Internet. In *International Conference on Dependable Systems and Networks*, pages 167–176, 2002. doi:10.1109/DSN.2002.1028897.

[71] Miguel Correia, Nuno Ferreira Neves, and Paulo Veríssimo. From Consensus to Atomic Broadcast: Time-Free Byzantine-Resistant Protocols without Signatures. *The Computer Journal*, 49(1):82–96, 11 2005. doi:10.1093/comjnl/bxh145.

[72] Andrew Miller, Yu Xia, Kyle Croman, Elaine Shi, and Dawn Song. The Honey Badger of BFT Protocols. In *2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, page 31–42, New York, NY, USA, 2016. Association for Computing Machinery. doi:10.1145/2976749.2978399.

[73] Sisi Duan, Michael K. Reiter, and Haibin Zhang. BEAT: Asynchronous BFT Made Practical. In *2018 ACM SIGSAC Conference on Computer and Communications Security*, CCS '18, page 2028–2041, New York, NY, USA, 2018. Association for Computing Machinery. doi:10.1145/3243734.3243812.

[74] Bingyong Guo, Zhenliang Lu, Qiang Tang, Jing Xu, and Zhenfeng Zhang. *Dumbo: Faster Asynchronous BFT Protocols*, page 803–818. Association for Computing Machinery, New York, NY, USA, 2020. doi:10.1145/3372297.3417262.

[75] T. Freitas, J. Soares, M.E. Correia, and R. Martins. Skynet: a Cyber-Aware Intrusion Tolerant Overseer. In *53rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks - Supplemental Volume, DSN-S 2023*, 2023. doi:10.1109/DSN-S58398.2023.00034.

[76] Paulo Verissimo, Alysson Bessani, and Marcelo Pasin. The TClouds architecture: Open and resilient cloud-of-clouds computing. In *IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN 2012)*, pages 1–6. IEEE, 2012. doi:10.1109/dsnw.2012.6264686.

[77] Diego Kreutz, Fernando MV Ramos, and Paulo Verissimo. Towards secure and dependable software-defined networks. In *second ACM SIGCOMM workshop on Hot*

topics in software defined networking, pages 55–60, 2013. doi:10.1145/2491185.2491199.

[78] Jerome H Saltzer and Michael D Schroeder. The protection of information in computer systems. *IEEE*, 63(9):1278–1308, 1975. doi:10.1109/PROC.1975.9939.

[79] Harold Booth, Doug Rike, Gregory A Witte, et al. The national vulnerability database (nvd): Overview. 2013.

[80] Veneta Yosifova, Antoniya Tasheva, and Roumen Trifonov. Predicting vulnerability type in common vulnerabilities and exposures (CVE) database with machine learning classifiers. In *2021 12th National Conference with International Participation (ELECTRONICA)*, pages 1–6. IEEE, 2021. doi:10.1109/electronica52725.2021.9513723.

[81] Anatoliy Gorbenko, Alexander Romanovsky, Olga Tarasyuk, and Oleksandr Biloborodov. Experience report: Study of vulnerabilities of enterprise operating systems. In *2017 IEEE 28th International Symposium on Software Reliability Engineering (ISSRE)*, pages 205–215. IEEE, 2017. doi:10.1109/issre.2017.20.

[82] Robert Thomson, Naoya Ito, Hinako Suda, Fangyu Lin, Yafei Liu, Ryo Hayasaka, Ryuzo Isochi, and Zhou Wang. Trusting tweets: The Fukushima disaster and information source credibility on Twitter. In *Iscram*, 2012.

[83] MISP Threat Sharing. https://www.misp-project.org/. "[Online; accessed 11-February-2023]".

[84] Sandro Pinto and Nuno Santos. Demystifying arm trustzone: A comprehensive survey. *ACM computing surveys (CSUR)*, 51(6):1–36, 2019. doi:10.1145/3291047.

[85] Oualid Demigha and Ramzi Larguet. Hardware-based solutions for trusted cloud computing. *Computers & Security*, 103:102117, 2021. doi:10.1016/j.cose.2020.102117.

[86] Roberto Di Pietro and Flavio Lombardi. Virtualization Technologies and Cloud Security: advantages, issues, and perspectives. In *From Database to Cyber Security: Essays Dedicated to Sushil Jajodia on the Occasion of His 70th Birthday*, pages 166–185. Springer, 2018. doi:10.1007/978-3-030-04834-1_9.

[87] Wazuh agents. https://documentation.wazuh.com/current/getting-started/components/wazuh-agent.html. "[Online; accessed 11-February-2023]".

[88] David Kennedy, Jim O'gorman, Devon Kearns, and Mati Aharoni. *Metasploit: the penetration tester's guide*. No Starch Press, 2011.

[89] hping3. https://www.kali.org/tools/hping3/. "[Online; accessed 13-February-2023]".

[90] theHarvester. https://gbhackers.com/theharvester-information-gathering-tool/, 2021. "[Online; accessed 13-February-2023]".

[91] Suhatati Tjandra, Indra Maryati, and Joshua Theopilus. Automated Software Testing for Multi Platform Applications using Katalon. 2021. doi:10.33508/wt.v20i1.3114.

[92] Manjit Kaur and Raj Kumari. Comparative study of automated testing tools: Testcomplete and quicktest pro. *International Journal of Computer Applications*, 24(1):1–7, 2011.

[93] Javier Pastor-Galindo, Pantaleone Nespoli, Félix Gómez Mármol, and Gregorio Martínez Pérez. The not yet exploited goldmine of osint: Opportunities, open challenges and future trends. *IEEE Access*, 8:10282–10304, 2020. doi:10.1109/access.2020.2965257.

[94] Miguel Garcia, Alysson Bessani, Ilir Gashi, Nuno Neves, and Rafael Obelheiro. Analysis of operating system diversity for intrusion tolerance. *Software: Practice and Experience*, 44(6):735–770, 2014. doi:10.1002/spe.2180.

[95] Sudhir Singh and Nasib Singh Gill. Analysis and study of k-means clustering algorithm. *Int. J. Eng. Res. Technol*, 2(7):2546–2551, 2013.

[96] Erich Schubert. Stop using the elbow criterion for k-means and how to choose the number of clusters instead. *ACM SIGKDD Explorations Newsletter*, 25(1):36–42, 2023. doi:10.1145/3606274.3606278.

[97] Sefa Eren Sahin and Ayse Tosun. A conceptual replication on predicting the severity of software vulnerabilities. In *23rd International Conference on Evaluation and Assessment in Software Engineering*, pages 244–250, 2019. doi:10.1145/3319008.3319033.

[98] Joana Cabral Costa, Tiago Roxo, João BF Sequeiros, Hugo Proenca, and Pedro RM Inacio. Predicting cvss metric via description interpretation. *IEEE Access*, 10:59125–59134, 2022. doi:10.1109/access.2022.3179692.

[99] Muhammad Sidik Asyaky and Rila Mandala. Improving the performance of HDBSCAN on short text clustering by using word embedding and UMAP. In *2021 8th international conference on advanced informatics: Concepts, theory and applications (ICAICTA)*, pages 1–6, 2021. doi:10.1109/icaicta53211.2021.9640285.

[100] Israel Mendonça, Antoine Trouvé, Akira Fukuda, Kazuaki J Murakami, CF Tsai, YH Hu, MC Wang, KE Liu, X Yu, Y Yuan, et al. On Clustering Algorithms: Applications in Word-Embedding Documents. *J. Comput.*, 14(2):88–92, 2019. doi:10.17706/jcp.14.2.88-92.

[101] Tadeu Freitas, Carlos Novo, Inês Dutra, João Soares, Manuel E Correia, Benham Shariati, and Rolando Martins. A risk manager for intrusion tolerant systems: Enhancing hal 9000 with new scoring and data sources. *Software: Practice and Experience*, 2025. doi:10.1002/spe.70016.

[102] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013. doi:10.48550/arXiv.1301.3781.

[103] Clément Elbaz, Louis Rilling, and Christine Morin. Fighting N-day vulnerabilities with automated CVSS vector prediction at disclosure. In *15th International Conference on Availability, Reliability and Security*, pages 1–10, 2020. doi:10.1145/3407023.3407038.

[104] Shaofeng Kai, Fan Shi, Jinghua Zheng, et al. Vuldistilbert: A cps vulnerability severity prediction method based on distillation model. *Security and Communication Networks*, 2023, 2023. doi:10.1155/2023/2118305.

[105] CJ van Rijsbergen. *Information retrieval*. Butterworth-Heinemann, 1979.

[106] Gerald J Kowalski. *Information retrieval systems: theory and implementation*, volume 1. springer, 2007. doi:10.1016/s0898-1221(97)80226-x.

[107] Chris Buckley and Alan F Lewit. Optimization of inverted vector searches. In *8th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 97–110, 1985. doi:10.1145/253495.253515.

[108] Douglass R Cutting, David R Karger, Jan O Pedersen, and John W Tukey. Scatter/-gather: A cluster-based approach to browsing large document collections. In *ACM SIGIR Forum*, volume 51, pages 148–159, 2017. doi:10.1145/133160.133214.

[109] Oren Zamir, Oren Etzioni, Omid Madani, and Richard M Karp. Fast and intuitive clustering of Web documents. In *KDD*, volume 97, pages 287–290, 1997.

[110] Michael Steinbach, George Karypis, and Vipin Kumar. A comparison of document clustering techniques. 2000.

[111] Ying Zhao and George Karypis. Comparison of Agglomerative and Partitional Document Clustering Algorithms. 2002. doi:10.21236/ada439503.

[112] Bjornar Larsen and Chinatsu Aone. Fast and effective text mining using linear-time document clustering. In *fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 16–22, 1999. doi:10.1145/312129.312186.

[113] Vivek Kumar Singh, Nisha Tiwari, and Shekhar Garg. Document clustering using k-means, heuristic k-means and fuzzy c-means. In *2011 International conference on computational intelligence and communication networks*, pages 297–301, 2011. doi:10.1109/cicn.2011.62.

[114] John A Hartigan. *Clustering algorithms*. John Wiley & Sons, Inc., 1975. doi:10.2307/2529577.

[115] Xin Liu, Yihong Gong, Wei Xu, and Shenghuo Zhu. Document clustering with cluster refinement and model selection capabilities. In *25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 191–198, 2002. doi:10.1145/564410.564411.

[116] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905, 2000. doi:10.1109/34.868688.

[117] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on pattern analysis and machine intelligence*, 24(5):603–619, 2002. doi:10.1109/34.1000236.

[118] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pages 226–231, 1996. doi:10.1109/icde.1998.655795.

[119] Leland McInnes and John Healy. Accelerated hierarchical density based clustering. In *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 33–42, 2017. doi:10.1109/icdmw.2017.12.

[120] Silke Wagner and Dorothea Wagner. Comparing clusterings: an overview. 2007. doi:10.5445/IR/1000011477.

[121] Katherine Cardoso Petulante Fernandes, Simon Lucas Jonker, Weizhi Meng, and Brooke Lampe. ScrapeIOC: Designing a Web-scraping Tool for Malware Detection based on Indicators of Compromise. In *2023 IEEE International Conference on Metaverse Computing, Networking and Applications (MetaCom)*, pages 124–128. IEEE, 2023. doi:10.1109/metacom57706.2023.00034.

[122] Fernando Alves, Aurélien Bettini, Pedro M Ferreira, and Alysson Bessani. Processing tweets for cybersecurity threat awareness. *Information Systems*, 95:101586, 2021. doi:10.1016/j.is.2020.101586.

[123] Philipp Kuehn, David N Relke, and Christian Reuter. Common vulnerability scoring system prediction based on open source intelligence information sources. *Computers & Security*, 131:103286, 2023. doi:10.1016/j.cose.2023.103286.

[124] Common Vulnerability Scoring System Calculator. https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator. "[Online; accessed 08-April-2024]".

[125] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[126] Rebecca Killick, Paul Fearnhead, and Idris A Eckley. Optimal detection of changepoints with a linear computational cost. *Journal of the American Statistical Association*, 107(500):1590–1598, 2012. doi:10.1080/01621459.2012.737745.

[127] Sally Adam. Unpatched Vulnerabilities: The Most Brutal Ransomware Attack Vector. https://news.sophos.com/en-us/2024/04/03/unpatched-vulnerabilities-the-most-brutal-ransomware-attack-vector/, 2024. "[Online; accessed 07-April-2024]".

[128] Peter Mell, Karen Scarfone, and Sasha Romanosky. Common vulnerability scoring system. *IEEE Security & Privacy*, 4(6):85–89, 2006. doi:10.1109/msp.2006.145.

[129] Tadeu Freitas, Erick Silva, Rehana Yasmin, Ali Shoker, Manuel E. Correia, Rolando Martins, and Paulo Esteves-Verissimo. EVSOAR: Security Orchestration, Automation and Response via EV Charging Stations. In *2025 IEEE 101st Vehicular Technology Conference (VTC2025-Spring)*, pages 1–7, 2025. doi:10.1109/VTC2025-Spring65109.2025.11174597.

[130] Daniel Stenberg. DISPUTED, not REJECTED. *LWN.net*, 2024. "[Online; accessed 27-July-2025]". URL: https://lwn.net/Articles/963240/.

[131] Rajkumar Singh Rathore, Chaminda Hewage, Omprakash Kaiwartya, and Jaime Lloret. In-vehicle communication cyber security: challenges and solutions. *Sensors*, 22(17):6679, 2022. doi:10.3390/s22176679.

[132] Ondrej Burkacky et al. Cybersecurity in automotive. Technical report, McKinsey, 2020.

[133] Nasser Nowdehi. Automotive Communication Security. 2019.

[134] Falk Langer. Establishing an automotive cyber defense center. 2019. doi:10.13154/294-6652.

[135] Jenny Hofbauer, Kevin Klaus Gomez Buquerin, and Hans-Joachim Hof. *From SOC to VSOC: Transferring Key Requirements for Efficient Vehicle Security Operations*. Ruhr-Universität Bochum, 2023.

[136] Mera Nizam-Edden Saulaiman, Bálint László Iványi, Eszter Kail, Tamás György Pozsonyi, Kristóf Zsombor Kövesi, Rajmund Kail, Benedek Máté Tóth, Ákos Csilling,

and Anna Bánáti. Developing SIEM and Log Management for Automotive Network in a Simulated Environment. In *2024 IEEE 22nd Jubilee International SISY*, pages 000239–000244. doi:10.1109/sisy62279.2024.10737536.

[137] Philipp Meyer, Timo Hackel, Falk Langer, Lukas Stahlbock, Jochen Decker, Sebastian A Eckhardt, Franz Korf, Thomas C Schmidt, and Fabian Schüppel. A security infrastructure for vehicular information using sdn, intrusion detection, and a defense center in the cloud. In *2020 IEEE VNC*, pages 1–2, 2020. doi:10.1109/vnc51378.2020.9318351.

[138] Vita Santa Barletta, Danilo Caivano, Mirko De Vincentiis, Azzurra Ragone, Michele Scalera, and Manuel Ángel Serrano Martín. V-soc4as: A vehicle-soc for improving automotive security. *Algorithms*, 16(2):112, 2023. doi:10.3390/a16020112.

[139] Miguel Martín-Pérez, Jordi Marias Parella, Javier Fernández, Pablo de Juan Fidalgo, Jordi Casademont, Antonio Álvarez Romero, and Rodrigo Diaz-Rodriguez. A testbed for a nearby-context aware: Threat detection and mitigation system for connected vehicles. In *2023 IEEE JNIC*, pages 1–8, 2023. doi:10.23919/jnic58574.2023.10205891.

[140] Hendrik C Ferreira, Lutz Lampe, John Newbury, and Theo G Swart. *Power line communications: theory and applications for narrowband and broadband communications over power lines*. John Wiley & Sons, 2011. doi:10.1002/9780470661291.

[141] Upstream Security. Automotive Cybersecurity Standards and Regulations. *Upstream Auto*, 2025. "[Online; accessed 29-August-2025]". URL: https://upstream.auto/automotive-cybersecurity-standards-and-regulations/.

[142] Cybellum. UNECE WP.29 Automotive Cybersecurity Regulation Explained, 2024. "[Online; accessed 29-August-2025]". URL: https://cybellum.com/blog/understanding-unece-wp29-automotive-cybersecurity-regulation/.

[143] Secure Privacy. Smart Vehicle Data Ownership: Compliance & Innovation, 2025. "[Online; accessed 29-August-2025]". URL: https://secureprivacy.ai/blog/smart-vehicle-data-ownership.

[144] Julien Freudiger, Emiliano De Cristofaro, and Alex Brito. Privacy-friendly collaboration for cyber threat mitigation. Technical report, Tech. rep. Nov, 2014. doi:10.48550/ARXIV.1403.2123.

[145] Tim van de Kamp, Andreas Peter, Maarten H Everts, and Willem Jonker. Private sharing of IOCs and sightings. In *2016 ACM on Workshop on Information Sharing and Collaborative Security*, pages 35–38, 2016. doi:10.1145/2994539.2994544.

[146] Roland van Rijswijk-Deij, Gijs Rijnders, Matthijs Bomhoff, and Luca Allodi. Privacy-conscious threat intelligence using DNSBLoom. In *2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, pages 98–106. IEEE, 2019.

[147] Shahriar Badsha, Iman Vakilinia, and Shamik Sengupta. Privacy preserving cyber threat information sharing and learning for cyber defense. In *2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 0708–0714. IEEE, 2019. doi:10.1109/ccwc.2019.8666477.

[148] Juan R Trocoso-Pastoriza, Alain Mermoud, Romain Bouyé, Francesco Marino, Jean-Philippe Bossuat, Vincent Lenders, and Jean-Pierre Hubaux. Orchestrating collaborative cybersecurity: a secure framework for distributed privacy-preserving threat intelligence sharing. *arXiv preprint arXiv:2209.02676*, 2022. doi:10.48550/arXiv.2209.02676.

[149] Gina Fisk, Calvin Ardi, Neale Pickett, John Heidemann, Mike Fisk, and Christos Papadopoulos. Privacy principles for sharing cyber security data. In *2015 IEEE Security and Privacy Workshops*, pages 193–197. IEEE, 2015. doi:10.1109/spw.2015.23.

[150] Iman Vakilinia, Deepak K Tosh, and Shamik Sengupta. Privacy-preserving cybersecurity information exchange mechanism. In *2017 International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS)*, pages 1–7. IEEE, 2017. doi:10.23919/spects.2017.8046783.

[151] José M de Fuentes, Lorena González-Manzano, Juan Tapiador, and Pedro Peris-Lopez. PRACIS: Privacy-preserving and aggregatable cybersecurity information sharing. *Computers & Security*, 69:127–141, 2017. doi:10.1016/j.cose.2016.12.011.

[152] Farhan Sadique, Khalid Bakhshaliyev, Jeff Springer, and Shamik Sengupta. A System Architecture of Cybersecurity Information Exchange with Privacy (CYBEX-P). In *2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 0493–0498, 2019. doi:10.1109/CCWC.2019.8666600.

[153] Dennis Heimbigner and Dennis McLeod. A federated architecture for information management. *ACM Transactions on Information Systems (TOIS)*, 3(3):253–278, 1985. doi:10.1145/4229.4233.

[154] Inji Wijegunaratne, George Fernandez, and John Valtoudis. A federated architecture for enterprise data integration. In *2000 Australian Software Engineering Conference*, pages 159–167. IEEE, 2000. doi:10.1109/aswec.2000.844573.

[155] Inji Wijegunaratne and George Fernandez. *Distributed Applications Engineering: Building new applications and managing legacy applications with distributed technologies*. Springer Science & Business Media, 2012. doi:10.1007/978-1-4471-1550-2.

[156] Elsa Rizk and Ali H. Sayed. A Graph Federated Architecture with Privacy Preserving Learning. In *2021 IEEE 22nd International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pages 131–135, 2021. doi:10.1109/SPAWC51858.2021.9593148.

[157] Anantha K Bangalore and Arun K Sood. Securing web servers using self cleansing intrusion tolerance (SCIT). In *2009 Second International Conference on Dependability*, pages 60–65, 2009. doi:10.1109/depend.2009.15.

[158] Jiangshan Yu, David Kozhaya, Jeremie Decouchant, and Paulo Esteves-Verissimo. Repucoin: Your reputation is your power. *IEEE Transactions on Computers*, 68(8):1225–1237, 2019. doi:10.1109/tc.2019.2900648.

[159] Rick Kazman, Gregory Abowd, Len Bass, and Paul Clements. Scenario-based analysis of software architecture. *IEEE software*, 13(6):47–55, 2002. doi:10.1109/52.542294.