

UNIVERSITY OF TARTU
Faculty of Science and Technology
Institute of Computer Science
Computer Science Curriculum

Mykyta Luzan

Secure Data Sharing in the Internet of Vehicles Using Blockchain-based Federated Learning

Master's Thesis (30 ECTS)

Supervisor(s): Mubashar Iqbal, PhD
Raimundas Matulevičius, Prof. PhD

Tartu 2025

Secure Data Sharing in the Internet of Vehicles Using Blockchain-based Federated Learning

Abstract:

The Internet of Vehicles enables connected vehicles to share data and collaboratively learn to enhance road safety and traffic efficiency. Federated learning has emerged as a promising approach for enabling privacy-preserving collaborative learning among vehicles, allowing them to jointly train machine learning models without sharing raw sensitive data. However, the centralized architecture commonly used in federated learning introduces significant security vulnerabilities that can compromise system integrity and reliability.

While extensive research exists on federated learning security in general, there is insufficient analysis of how these security challenges manifest in specific application contexts, particularly in dynamic environments like IoV. Here we show that integrating Hyperledger Fabric's permissioned blockchain with zero-knowledge proofs creates a comprehensive security framework that effectively protects federated learning systems against both model tampering, aggregation protocol violation, and unauthorized access while maintaining privacy.

Our systematic analysis and implementation reveals that blockchain technology can address core vulnerabilities in centralized federated learning architectures while preserving their privacy benefits, demonstrating advantages over previous approaches that relied solely on cryptographic protocols or trusted third parties. By validating our framework through a concrete IoV data sharing implementation, we establish a practical foundation for securing federated learning in distributed environments.

The implications of this research extend beyond vehicular networks to any domain requiring secure collaborative learning among distributed participants. As autonomous systems become increasingly interconnected, this work demonstrates how combining blockchain with federated learning can enable trustworthy data sharing while preserving both privacy and security.

Keywords: Internet of Vehicles (IoV), Federated Learning (FL), Blockchain, Zero-Knowledge Proofs (ZKP), Secure Data Sharing, Secure Collaborative Learning, Information System Security Risk Management (ISSRM).

CERCS: P170 Computer science, numerical analysis, systems, control

Andmete Turvaline Jagamine Sõidukite Internetis Plokiahela-põhise Hajutatud Õppimisega

Lühikokkuvõte:

Sõidukite Internet (Internet of Vehicles, IoV) võimaldab ühendatud sõidukitel jagada andmeid ja koostööl põhinevalt õppida, et parandada liikluse ohutust ja tõhusust. Hajutatud õppimine (federated learning) on kerkinud perspektiivikaks lähenemiseks privaatsust säilitava koostööõppe võimaldamiseks sõidukite vahel, võimaldades neil ühiselt treenida masinõppe mudeleid ilma tundlike algandmete jagamiseta. Siiski toob hajutatud õppimise levinud tsentraliseeritud arhitektuur kaasa märkimisväärsed turvariskid, mis võivad ohustada süsteemi terviklikkust ja usaldusväärsust. Kuigi eksisteerib ulatuslik teaduskirjandus hajutatud õppimise turvalisuse kohta üldiselt, pole piisavalt analüüsitud, kuidas need turbeprobleemid avalduvad spetsiifilistes rakendusvaldkondades, eriti dünaamilistes keskkondades nagu sõidukite internet.

Käesolevas uurimuses näitame, et Hyperledger Fabric'i lubatud plokiahela integreerimine nullteadmiste tõenditega loob tervikliku turbemudeli, mis tõhusalt kaitseb hajutatud õppimise süsteemi mudeli manipuleerimise, agregeerimisprotokollide rikkumise ja volitamata juurdepääsu eest, säilitades seejuures privaatsuse. Meie süstemaatiline analüüs ja rakendus näitab, et plokiahela tehnoloogia suudab kõrvaldada tsentraliseeritud hajutatud õppimise arhitektuuride põhilised haavatavused, säilitades samal ajal nende privaatsuse eelised. See demonstreerib eeliseid varasemate lähenemiste ees, mis tuginesid ainult krüptograafilistele protokollidele või usaldusväärsetele kolmandatele osapooltele.

Meie raamistiku valideerimine konkreetse sõidukite andme jagamise rakenduse kaudu loob praktilise aluse hajutatud keskkondades hajutatud õppimise turvamiseks. Käesoleva uurimuse mõjud ulatuvad sõidukivõrkudest kaugemale, hõlmates kõiki valdkondi, kus on vaja turvalist koostööõpet hajutatud osalejate vahel. Kuivõrd autonoomsed süsteemid muutuvad järjest ühendatumateks, näitab käesolev töö, kuidas plokiahela ja hajutatud õppimise kombineerimine võimaldab usaldusväärset andme jagamist, säilitades nii privaatsuse kui ka turvalisuse.

Võtmesõnad: Sõidukite Internet (IoV), Hajutatud Õppimine (FL), Plokiahel, Nullteadmiste Tõendid (ZKP), Turvaline Andme jagamine, Turvaline Koostööõpe, Infosüsteemide Turvariskide Juhtimine (ISSRM).

CERCS: P170 Arvutiteadus, arvutusmeetodid, süsteemid, juhtimine

Acknowledgements

Cyber-security Excellence Hub in Estonia and South Moravia

This work is part of the Cyber-security Excellence Hub in Estonia and South Moravia (CHESS: <https://chess-eu.cs.ut.ee>) project funded by the European Union under Grant Agreement No. 101087529. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Research Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.

Supervisors

I would like to express my deepest gratitude to my supervisors, Mubashar Iqbal and Raimundas Matulevičius, for their unwavering support and guidance throughout my thesis journey. Their consistent assistance during our regular meetings, thoughtful feedback, and understanding during challenging periods helped me stay on track. Their supportive approach and dedication enabled me to overcome obstacles and successfully complete this thesis. The knowledge and experience they shared have been invaluable to both my academic growth and personal development.

Use of AI

During the development of this Master's thesis, several AI-powered tools were utilized in accordance with academic integrity guidelines, primarily serving as assistive technologies rather than content generators. The usage of these tools was focused on enhancing the technical quality and readability of the work while maintaining original thought and analysis throughout the research process. The use of AI was strictly limited to supporting roles in accordance with academic guidelines, ensuring that the intellectual contribution and academic integrity of the research were maintained throughout the process.

Grammarly¹, an AI-powered writing assistant, was employed during the final editing phase to ensure grammatical accuracy and improve the overall clarity of the text. This tool was specifically used for proofreading and identifying potential language issues in the near-final draft, serving as a technical aid rather than a content generator.

For research and development purposes, several large language models were consulted, including ChatGPT 4o² and Claude 3.5 Sonnet³. These tools were primarily utilized as knowledge-expansion resources and brainstorming aids, helping to explore

¹Grammarly. (2024). Grammarly [AI-powered writing assistant]. <https://www.grammarly.com/>

²OpenAI. (2024). ChatGPT (May 13 version) [large language model]. <https://chat.openai.com/>

³Anthropic. (2024). Claude 3.5 Sonnet (June 21 version) [large language model]. <https://claude.ai/>

different perspectives on thesis-related topics. They were particularly valuable for vocabulary enhancement, offering alternative phrasings and synonyms to improve the clarity of technical explanations. Importantly, all information obtained through these tools was independently verified through academic sources, and proper citations were maintained throughout.

For the visual elements of the thesis, Microsoft Designer⁴ was utilized to generate icons for architectural diagrams, enhancing the visual presentation of technical concepts while maintaining professional standards. All architectural diagrams were fully designed and created by the author, ensuring that the visual representation of the system architecture reflects the author's original work and understanding.

In the technical implementation phase, Cursor⁵, an AI-enhanced code editor powered by Claude 3.5 Sonnet API, was employed as a development tool. Cursor combines traditional code editing capabilities with AI assistance, providing features such as intelligent code suggestions, automated debugging assistance, and context-aware code completions. This usage was limited to technical assistance and did not extend to core algorithm development or critical system design decisions.

⁴Microsoft. (2024). Microsoft Designer [AI image generation tool]. <https://designer.microsoft.com/>

⁵Cursor Inc. (2024). Cursor [AI-enhanced code editor]. <https://www.cursor.com/>

Contents

1	Introduction	8
1.1	Motivation	8
1.2	Problem Description	9
1.3	Research Questions	10
1.4	Research Method	10
1.5	Contribution	11
1.6	Structure	12
2	Background	14
2.1	Federated Learning	14
2.2	Blockchain	15
2.3	Internet of Vehicles	16
2.4	Security Risk Management	18
2.5	Summary	19
3	Use Case: FL Data Sharing in IoV	20
3.1	System Context and Actors	20
3.2	Process Flow	21
3.3	Summary	22
4	Literature Review	24
4.1	Attacks	24
4.1.1	Poisoning Attacks	25
4.1.2	Free Rider Attacks	26
4.1.3	Model Inversion Attacks	27
4.1.4	Membership Inference Attacks	28
4.1.5	Communication Attacks	29
4.1.6	Third-Party Software Attacks	30
4.2	Defense Mechanisms	31
4.2.1	Authorization and Authentication	32
4.2.2	Blockchain	32
4.2.3	Zero-Knowledge Proof	33
4.2.4	Differential Privacy	34
4.2.5	Secure Aggregation	34
4.2.6	Robust Training	35
4.2.7	Anomaly Detection	35
4.2.8	Other	35
4.3	Summary	36

5	Security Risks Assessment	37
5.1	Threat Model for Centralized FL	37
5.2	Asset-Related Concepts	42
5.2.1	Business assets	42
5.2.2	Security criteria	43
5.2.3	System assets	43
5.3	Risk-Related Concepts	46
5.4	Treatment-Related Concepts	54
5.5	Summary	56
6	Solution Design and Implementation	58
6.1	Solution Design	58
6.1.1	Solution Overview	59
6.1.2	Identity Management	61
6.1.3	Network Initialization and Participants Registration	63
6.1.4	Global Model Request and Distribution	65
6.1.5	Local Model Training and Submission	66
6.1.6	Models Aggregation, ZKP Validation, and Submission	68
6.2	Implementation	70
6.2.1	Flower - Federated Learning Framework	70
6.2.2	EZKL - Zero-Knowledge Proof Compiler	72
6.2.3	Hyperledger Fabric SDK	75
6.2.4	Deployment	78
6.3	Experiment	82
6.4	Summary	83
7	Concluding Remarks	85
7.1	Limitations	85
7.2	Answers to Research Questions	86
7.3	Future Work	87
	List of References	99
	Appendix	100
	I. Glossary	100
	II. Extended Risk-Related Concepts	101
	III. Security Requirements	111
	IV. Security Controls	113
	V. Licence	115

1 Introduction

We are currently experiencing the emergence of the Internet of Vehicles, a new paradigm that integrates intelligent computing and vehicle networking. The latest technological advances in the industrial IoT have provided researchers [1, 2] with the ability to utilize billions of low-cost and readily available wireless sensor nodes. These nodes continuously monitor a certain set of parameters in a given environment, including trajectories, traffic information, and multimedia data. However, the massive amount of data generated by these sensors must be efficiently utilized in order to make decisions based on it more reliable. Due to the dynamic nature of such systems, it is important to provide provenance and control over the data quality.

Federated Learning is a decentralized machine learning approach that enables training models across multiple decentralized edge devices or servers holding local data samples without exchanging them. In the context of the IoV, FL offers a promising solution for collaborative learning while preserving the privacy of sensitive vehicle data. This approach allows for continuous improvement of machine learning models that can enhance various aspects of IoV, such as traffic prediction, autonomous driving capabilities, and personalized services

Traditional centralized machine learning approaches require data to be aggregated in a central location, which raises privacy concerns and may not be feasible in IoV scenarios due to the distributed nature of data generation and storage.

The integration of blockchain technology with FL has garnered significant interest among researchers due to its potential to enhance the security, transparency, and decentralization of the learning process. Blockchain’s distinctive attributes, such as immutability, distributed consensus, and smart contracts, make it an appealing prospect for improving the reliability and credibility of FL systems. Consequently, blockchain-powered FL frameworks are being actively explored and employed in diverse industrial domains, including IoV, due to their heightened security and efficiency compared to conventional centralized FL approaches.

1.1 Motivation

Data sharing in the IoV faces challenges due to concerns about data security and privacy. Data generated by IoV infrastructures is often sensitive in nature and must be shared using unreliable inter-vehicle communications. This requires privacy-preserving mechanisms to protect it from privacy abuse. Otherwise, data providers may be discouraged from sharing their data for further analysis.

To address this problem, a FL approach to data sharing is proposed [3]. FL processes and transmits vehicular big data without sharing sensitive local information from each device. Each vehicle trains a local model and propagates it to other peers. This transforms the data sharing problem into a machine learning task and mitigates some privacy risks.

Additionally, collaborative training of different agents like vehicles, RSUs, or cellular base stations, can improve network and storage resources utilization without sharing large amounts of raw data records. This is especially important for real-time systems in the IoV, where lower latency is required.

However, this approach is still vulnerable to security attacks and struggles with the single point of failure. Data providers must trust the centralized server that aggregates models, as well as other peers that may be dishonest. If the server is affected by any malicious threat, then the gathered local data can be stolen or violated by attackers. Moreover, the centralized aggregator can be a bottleneck in terms of the computational capabilities of the system. The IoV can include millions of actors, so it would be impossible to process all the data. To achieve both security and scalability, decentralization is required [4, 5, 3, 6].

Using blockchain technology, we can overcome these issues and ensure security among untrusted edge users. Blockchain provides decentralization, transparency, immutable state, traceability, and provenance through distributed immutable ledgers in a peer-to-peer network. FL powered by blockchain eliminates centralization risks, provides trust among devices, and improves the reliability and robustness of the system. However, communication costs, incentive allocation rules, security, and privacy protection against FL-based attacks are still problems that need to be addressed [5, 1].

1.2 Problem Description

In the domain of FL, the centralized architecture for model aggregation introduces significant security vulnerabilities that can compromise system integrity and reliability. While existing research has explored various attacks and defense mechanisms for FL systems in general, there is insufficient analysis of how these security challenges manifest in specific application contexts, particularly in dynamic environments like the IoV. Although FL protects raw data privacy through distributed training, its centralized aggregation process remains vulnerable to both malicious participants and compromised infrastructure. This research aims to bridge this gap by conducting a systematic security analysis of FL in a concrete IoV use case. Through detailed examination of assets, risks, and vulnerabilities in this specific context, we develop and validate a blockchain-based security framework incorporating ZKPs. Rather than attempting to address all IoV-specific challenges, our work focuses on demonstrating how blockchain technology can enhance security for FL systems while maintaining their privacy benefits, using vehicular data sharing as an illustrative example of implementation in practice.

1.3 Research Questions

Considering the motivation and problem statement, this study will concentrate on addressing the following primary research question:

How blockchain technology could be used to mitigate the security risks in the centralized FL data sharing process in the context of the IoV?

Subquestions:

- **RQ1:** What are the security risks of centralized FL data sharing?
- **RQ2:** How to build a threat model using blockchain technology to mitigate the security risks of centralized FL data sharing in the context of IoV?
- **RQ3:** How to design and implement secure data sharing using blockchain-based FL?

1.4 Research Method

We adopt a design science research approach to achieve our objectives and answer the research questions. This method drives innovation by analyzing, designing, implementing, and managing information effectively. Further research is needed to evaluate existing solutions for the project. The most suitable methodology for this scenario is design science [7, 8]. Table 1 outlines the guidelines for applying design research in information systems and their alignment with this thesis.

In this thesis, the design science guidelines provide a structured framework to ensure both rigor and effectiveness in the research. To begin, we identified and addressed key challenges in FL-based data sharing by conducting a thorough literature review using snowball sampling to uncover existing attacks and defense mechanisms for FL (G2, G5). We then performed a security risk analysis of centralized and blockchain-based FL solutions, leading to the development of several artifacts, including threat models, ISSRM and CIA models, and BPMN diagrams (G3, G4, G5). Additionally, we designed, implemented, and validated a blockchain-based FL solution for data sharing within the IoV context (G7), supported by various design artifacts such as UML diagrams for component and deployment architecture (G4). The validation process involved assessing durability against different threat scenarios (G6).

In summary, these guidelines provide a structured path for achieving impactful and rigorous research outcomes. Each aspect of the design science methodology is carefully integrated into the structure of this thesis, ensuring relevance, innovation, and clarity in addressing the challenges of FL-based data sharing in IoV systems.

Table 1. Design science guidelines for this thesis.

Guideline	Description	Thesis Artifact
G1: Design as an Artifact	Design-science research must create a functional artifact (e.g., construct, model, method, or instantiation).	Blockchain-based FL solution for data sharing in IoV.
G2: Problem Relevance	The goal is to develop technology-driven solutions to significant business problems.	Literature review for attacks and defense mechanisms for FL.
G3: Design Evaluation	The artifact's utility and effectiveness must be rigorously demonstrated through evaluation.	Security risks analysis of centralized and blockchain-based solutions: ISSRM, CIA model. Proof of concept implementation validation.
G4: Research Contributions	Design-science research should make verifiable contributions in artifacts, foundations, or methodologies.	Threat model, BPMN diagrams of FL-based data sharing, SRM for centralized and decentralized FL-based data sharing in IoV, UML diagrams for the solution design.
G5: Research Rigor	The research must apply rigorous methods in the creation and evaluation of the artifact.	Literature review, BPMN diagrams, UML diagrams.
G6: Design as a Search Process	Creating an effective artifact requires using available resources to meet goals within the problem's constraints.	Use case design and implementation of the blockchain-based FL solution and evaluation of implementation.
G7: Communication of Research	Research must be communicated clearly to both technical and managerial audiences.	Comprehensive thesis report (literature review, security risks analysis, design, implementation).

1.5 Contribution

This thesis makes several contributions to the field of secure FL-based data sharing in the context of IoV. These contributions span from comprehensive theoretical analyses to practical implementations and evaluations, as outlined below:

1. **Comprehensive Literature Review:** Conducted an extensive literature review using snowball sampling to identify current attacks and defense mechanisms in FL. This review serves as the foundation for understanding the vulnerabilities and security challenges in FL-based data sharing systems.
2. **Security Risk Analysis:** Developed a comprehensive security risk analysis framework that includes a detailed STRIDE-based threat model for centralized FL in IoV environments, identifying multiple distinct threats across various categories. This includes extensive vulnerability mappings between system assets and potential

attacks, along with systematic analysis using the ISSRM model to evaluate security risks and their impacts on business assets. The framework culminates in a detailed set of security requirements and control components for securing FL systems.

3. **Design of Blockchain-based Solution:** Designed a blockchain-based FL solution for secure data sharing in IoV. We present a blockchain-based security architecture that combines permissioned blockchain with identity management system and role-based access control with ZKPs for FL aggregation. Created BPMN diagrams to map out the data flow in the proposed FL-based IoV scenario alongside UML diagrams detailing the system components and deployment architecture.
4. **Security Validation:** Developed a concrete implementation and validation of our security framework through a working proof-of-concept that integrates FL coordination, ZKP verification, and blockchain infrastructure. The implementation demonstrates practical protection against critical attack scenarios identified in our threat model, with substantial coverage of the specified security requirements and controls.

1.6 Structure

This thesis is organized into the following chapters:

1. **Introduction:** Provides the motivation for securing data sharing in IoV, outlines research questions focused on blockchain-based security enhancement for FL, and presents the design science research methodology and contributions.
2. **Background:** Introduces core concepts of FL, blockchain technology, IoV, and security risk management that form the foundation for understanding the research.
3. **Use Case: FL Data Sharing in IoV:** Presents a practical scenario of FL-based data sharing between vehicles and infrastructure, illustrating system actors, process flows, and real-world applications.
4. **Literature Review:** Examines existing security threats and defense mechanisms in FL through systematic analysis of current research.
5. **Security Risks Assessment:** Develops comprehensive threat models and security analysis using STRIDE methodology and ISSRM framework, evaluating assets, risks, and potential blockchain-based countermeasures.
6. **Solution Design and Implementation:** Details the architecture and implementation of a blockchain-based FL solution using Hyperledger Fabric and ZKPs, including deployment and experiment description.

7. **Discussion:** Evaluates the research outcomes against the initial research questions, acknowledges implementation limitations, and suggests future research directions.
8. **Conclusion:** Summarizes the key findings and contributions of the thesis.

Each chapter builds upon the previous ones, creating a coherent narrative that progresses from theoretical foundations to practical implementation and analysis. This structure ensures a comprehensive exploration of the research questions and a clear presentation of the study's contributions to the field.

2 Background

The field of machine learning is rapidly growing in various directions. One of the most promising applications is the combination of Internet of Things (IoT) technologies to collaboratively train and share models. FL is a useful technique for training machine learning models on local devices without sharing sensitive data. However, there are security issues and a lack of transparency due to the centralization of servers that perform computations and store data. These techniques struggle with a single point of failure and are exposed to various security attacks and data leaks. Blockchain can be introduced to overcome these issues. This topic is especially important in the domain of the Internet of Vehicles due to its distributed design.

This section introduces the core technologies and concepts underpinning our research: FL for privacy-preserving distributed machine learning, blockchain for enhanced security and trust, and the IoV as our application domain. We also examine Security Risk Management methodologies that guide our analysis of threats and countermeasures. Understanding these fundamental concepts and their interactions provides the necessary foundation for addressing security challenges in IoV data sharing, which we explore in subsequent chapters.

2.1 Federated Learning

Federated learning is an innovative privacy-preserving approach that enables secure and efficient data sharing across edge computing and distributed systems. At its core, FL allows multiple data providers to collaboratively train a global model without exposing their raw data. Traditional machine learning systems rely on centralized processing units where data must be gathered and transmitted for training, even in cloud computing environments. FL takes a fundamentally different approach by coordinating decentralized training, where each participant uses their local dataset. A curator then aggregates these locally trained models to create a global model that is shared among all participants. This architecture allows organizations to benefit from collective learning while protecting sensitive data and reducing privacy risks during transmission. Furthermore, FL enhances network efficiency and scalability since only model weights of fixed size need to be shared, rather than large volumes of raw data [4, 1, 5].

Based on data characteristics, FL can be implemented in several ways. Horizontal FL is used when datasets share feature spaces but contain different sample sets. Vertical FL applies when datasets have the same samples but different features. When datasets differ in both features and samples, federated transfer learning provides a solution [9, 10].

Model aggregation serves as the cornerstone of FL, combining individually trained models into a unified whole. Initially, FL implementations required identical architectures across all local models to ensure uniform parameter structures. This enabled straightforward federation through parameter averaging, a technique known as federated

averaging [11]). However, this approach faces challenges when model performance varies significantly between participants. To address this limitation, weighted federated averaging was developed, allowing more accurate local models to have greater influence on the final aggregated model [4, 1].

The field continues to face challenges from inefficient client training and static hyperparameter usage. Advanced aggregation functions offer promising solutions to these issues. A particularly innovative approach leverages genetic algorithms, clustering edge devices based on their models' hyperparameters and applying evolutionary updates to optimize configurations. This method can even determine optimal model architectures. While these advances show significant potential for improving flexibility and performance, substantial research opportunities remain in this area [12].

Despite these advances, centralized aggregation in classical FL architectures presents several critical challenges. The approach remains vulnerable to single points of failure and malicious attacks in distributed untrusted environments. High data volumes from multiple participants can overwhelm central curators, creating communication bottlenecks. Additionally, maintaining efficient, traceable, and transparent model management and synchronization becomes increasingly difficult at scale. Blockchain technology emerges as a promising solution to address these fundamental limitations.

2.2 Blockchain

Blockchain is a revolutionary decentralized and immutable ledger technology that operates on peer-to-peer networks. Its defining characteristic is decentralization, distributing data control across network nodes rather than concentrating it in a central authority. The system relies on miners or validators who verify and append transaction blocks to the immutable ledger through consensus algorithms such as Proof-of-Work (PoW) or Proof-of-Stake (PoS). Smart contracts - self-executing code deployed on the blockchain - enable automated processes and sophisticated access control mechanisms.

The blockchain's architecture ensures data integrity by incorporating hash values of previous blocks into each new block. This design automatically rejects unauthorized modifications during validation, creating a trustworthy system with built-in traceability and provenance. Through its combination of consensus mechanisms, tamper-proof records, and smart contracts, blockchain enables secure peer-to-peer communication without requiring trusted intermediaries. By eliminating central servers, the system significantly enhances robustness and resilience against single-point failures [5].

Blockchain implementations utilize two distinct storage formats: on-chain and off-chain [13]. On-chain storage, which maintains data directly in the blockchain ledger, faces inherent size limitations and typically contains textual, compressed, or metadata information. Off-chain storage complements these limitations by keeping larger files - such as images, videos, and documents - in external storage while recording their

cryptographic hashes in blockchain transactions. This hybrid approach enables efficient data referencing while maintaining integrity verification through hash comparison.

Blockchain networks operate in either public (permissionless) or private (permissioned) configurations [10]. Public blockchains, exemplified by Bitcoin and Ethereum, permit unrestricted, anonymous participation in transaction creation, validation, and consensus processes. In contrast, private blockchains require explicit invitation from network managers and typically feature customizable consensus mechanisms, controlled participation, and enhanced speed, efficiency, and security. These characteristics make permissioned blockchains particularly suitable for privacy-sensitive applications in medical, banking, and traffic management systems.

Consensus protocols form the foundation of blockchain architecture by governing transaction processing and block validation. These algorithms establish crucial structural and computational constraints for the network. The widely-adopted PoW consensus in public blockchains, for instance, requires miners to compete in solving computationally intensive cryptographic puzzles to validate new blocks. While this approach effectively prevents Distributed Denial-of-Service (DDoS) and Byzantine attacks, its substantial computational requirements create scalability challenges and increase operational costs. These limitations make PoW less suitable for applications requiring low-latency communication, particularly in the IoV scenarios [1, 4, 14].

While blockchain technology offers promising solutions for security and decentralization, its effective implementation in IoV systems requires careful consideration of the specific challenges and requirements of vehicular networks. The choice of consensus mechanism, storage strategy, and network configuration must be optimized for the dynamic nature of vehicular communications and the need for real-time data processing. Understanding these considerations becomes particularly important when examining the IoV architecture and its unique security challenges.

2.3 Internet of Vehicles

The Internet of Vehicles (IoV) represents a new paradigm in intelligent transportation, where interconnected vehicles and infrastructure collaborate to enhance road safety, traffic efficiency, and driving experiences [1]. At its core, IoV consists of three main components: vehicles equipped with sensing and communication capabilities, Roadside Units (RSUs) that provide edge computing services, and Multi-access Edge Computing (MEC) servers that handle more complex computational tasks. While vehicles typically have limited computing and communication resources, RSUs and MECs offer greater processing power and storage capabilities, connected through a hierarchical communication structure [1].

The IoV architecture enables various forms of communication, including vehicle-to-vehicle (V2V), vehicle-to-infrastructure (V2I), and vehicle-to-everything (V2X) interactions. These connections are often organized as Vehicular Ad-hoc Networks (VANETs),

providing flexible and dynamic networking capabilities that adapt to changing traffic conditions [2]. To effectively manage the massive amount of data generated within this ecosystem, IoV implementations typically follow a three-layer structure [3]:

The perception layer serves as the foundation, collecting raw data from various vehicle sensors and environmental monitoring systems. Given the privacy-sensitive nature of this data and its substantial volume, the layer implements data processing techniques to protect and compress the information before transmission. This is where privacy-preserving technologies like FL can play a crucial role in protecting sensitive data while enabling collaborative learning.

The networking layer handles communication between vehicles and infrastructure using various wireless technologies (GSM, Wi-Fi, LTE, 5G) and protocols, with Dedicated Short-Range Communications (DSRC) being particularly common. The dynamic topology of vehicular networks presents significant challenges in ensuring secure, efficient data transmission while maintaining low latency. This layer is where security mechanisms like blockchain can be integrated to enhance data protection and trust.

The application layer provides the intelligence and decision-making capabilities of the IoV system. It manages storage, processes the aggregated data, and implements various services such as traffic prediction, route optimization, and emergency response coordination. For example, RSUs can continuously monitor traffic conditions and coordinate traffic signals for optimal flow, or quickly disseminate emergency alerts about accidents or severe weather conditions [2].

However, the distributed nature of IoV presents significant security and privacy challenges. RSUs, while crucial for edge computing and data management, are considered semi-trusted entities as they are typically deployed in unsecured locations, making them vulnerable to compromise [14]. Vehicles may be reluctant to share sensitive data with RSUs due to privacy concerns, while peer-to-peer data sharing between vehicles raises issues of unauthorized access and the need for decentralized security mechanisms.

These security challenges are further complicated by resource constraints and the dynamic nature of vehicular networks. Vehicles cannot support massive data storage or complex security protocols, yet the data they generate is increasingly fine-grained and requires low-latency processing. Additionally, the spatial and temporal relevance of vehicular data adds another layer of complexity to security requirements [14].

Understanding these architectural components, communication patterns, and security challenges is essential for developing effective security solutions for IoV data sharing. In the following chapters, we will explore how FL and blockchain technologies can be combined to address these challenges while maintaining the efficiency and functionality of IoV systems.

2.4 Security Risk Management

Security Risk Management (SRM) is an analytical procedure that helps identify valuable assets, stakeholders, and operations within a system. It provides guidance to find and implement appropriate solutions for specific situations and mitigation strategies. SRM is concerned with lowering the risk of intentional unauthorized harm to valuable assets to a level acceptable to the system's stakeholders [15].

The ISSRM domain model addresses security risk management at three different conceptual levels: asset-related concepts, risk-related concepts, and risk treatment-related concepts [16]. Assets are classified as business assets (information, processes, capabilities essential to the business) and system assets (components of IS like hardware, software, networks that support business assets). Security criteria (Confidentiality, Integrity, and Availability) act as constraints on business assets to define their security needs [17]. The full ISSRM domain model is depicted in Figure 1.

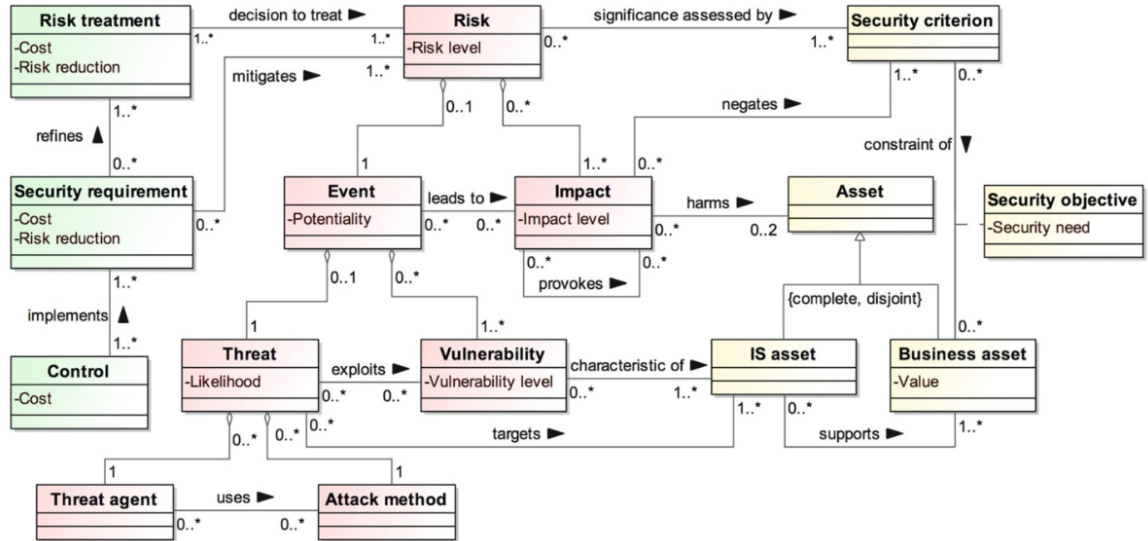


Figure 1. The ISSRM domain model [17]

In the risk-related domain, a risk combines a threat with one or more vulnerabilities leading to negative impacts on assets. A threat agent uses attack methods to target system assets by exploiting their vulnerabilities. The resulting impact can affect both business and system assets by negating their security criteria [15]. The risk treatment domain deals with decisions and requirements to mitigate identified risks through security controls - countermeasures that implement the security requirements to protect assets.

This three-part conceptual framework will guide our analysis of security risks in FL for IoV data sharing. By systematically examining assets, risks, and treatments, we can evaluate potential threats to the FL system and determine appropriate blockchain-based

countermeasures. The ISSRM approach helps ensure we consider both technical and organizational aspects of security, from protecting sensitive vehicle data to implementing system-wide controls.

The model's systematic nature also aligns well with our research methodology, providing a structured way to analyze security requirements and design blockchain-based solutions. This framework will help us bridge the gap between identifying theoretical security risks and implementing practical countermeasures in our IoV FL system.

2.5 Summary

This chapter introduced the core technologies and concepts that form the foundation of our research. We began by examining FL as a privacy-preserving approach for distributed machine learning, highlighting its benefits for data sharing while identifying key challenges around centralized aggregation and security vulnerabilities. We then explored blockchain technology, discussing how its decentralized architecture, consensus mechanisms, and smart contracts can enhance security and trust in distributed systems. The IoV section detailed the three-layer architecture of modern vehicular networks and their unique security challenges, particularly regarding data sharing and privacy preservation. Finally, we presented the Security Risk Management framework, introducing the ISSRM domain model that will guide our systematic analysis of security threats and countermeasures in subsequent chapters. Together, these components provide the necessary background for understanding the security challenges in IoV data sharing and how blockchain-based FL can address them while maintaining efficiency and privacy.

3 Use Case: FL Data Sharing in IoV

Building upon the core concepts introduced in the Background Chapter, such as FL, Blockchain technology, and the Internet of Vehicles, this chapter presents a practical use case that combines these elements to address secure data sharing challenges in intelligent transportation systems. This thesis presents a case study of FL as a data sharing process between vehicles in an intelligent transportation system scenario.

This use case was specifically chosen because it exemplifies the key challenges in modern intelligent transportation systems: the need to leverage collective intelligence from distributed data sources while protecting sensitive information about individual vehicles and their users. The dynamic nature of vehicular networks, with vehicles constantly joining and leaving the system, also makes it an ideal scenario for examining security challenges in distributed systems.

3.1 System Context and Actors

The primary actors in our system include Vehicles, Road Side Units (RSUs), and the Intelligent Transportation System (ITS). Vehicles are mobile nodes equipped with sensors and computing capabilities that generate traffic data and participate in FL. RSUs are fixed infrastructure nodes that serve as aggregation points and facilitate communication. The ITS is the broader system that utilizes the shared knowledge for traffic management.

Our framework aims to provide a universal (task-indifferent) secure FL-based data sharing mechanism for various machine learning use cases in the IoV. While the framework supports multiple ML tasks, we use traffic trajectory prediction and road sign classification as concrete examples to illustrate the workflow.

Example 1: Traffic Trajectory Prediction. Consider a scenario where vehicles need to predict future traffic patterns. Each vehicle collects geographic coordinates from GPS sensors, timestamps of vehicle movements, speed and acceleration data, and environmental conditions affecting traffic flow.

This data is particularly sensitive as it can reveal individual travel patterns, frequent destinations, and personal habits. Traditional centralized approaches would require vehicles to share this raw data, creating significant privacy risks. Additionally, the data volumes involved make centralized collection and processing impractical for real-time applications.

Instead of exposing raw data, our FL-based approach enables vehicles to collaborate by sharing only model updates derived from their local data. This preserves privacy while still allowing the system to learn meaningful traffic patterns for prediction.

Example 2: Road Sign Classification. Another relevant use case is road sign classification, where vehicles need to recognize and interpret various road signs for safe and efficient navigation. Each vehicle uses its camera sensors to capture images of road signs, which are then processed by on-board machine learning models.

Sharing raw images of road signs can inadvertently expose sensitive information about the vehicle's location and surroundings. Moreover, different vehicles may encounter different subsets of road signs depending on their routes, leading to imbalanced local datasets that can hinder the performance of individually trained models.

By applying FL, vehicles can collectively improve their road sign classification capabilities without directly sharing raw image data. Each vehicle trains its local model on its own image dataset, and only the model updates are shared and aggregated to enhance the global model. This approach ensures that all vehicles benefit from the collective knowledge while preserving the privacy of their individual experiences.

3.2 Process Flow

The data sharing process consists of three main phases that deliver value to ITS users (see Figure 2):

Data Sharing Initiation: The process begins when at least two authorized Vehicles (registered in the ITS with Personal Vehicle Profile) approach a crossroad with an RSU. The Vehicle requests the data sharing process via authentication request to the RSU. After receiving the details of data sharing participants, the RSU registers them in a new data sharing session and broadcasts a message to start the FL process.

Federated Learning: In our centralized scenario, FL operates at two levels (see Figure 3):

1. Local Training (Vehicle Level):
 - Vehicles train models using their private local data
 - Only model updates are shared, protecting raw data privacy
 - Training occurs within resource constraints of vehicular hardware
2. Global Aggregation (RSU Level):
 - RSU serves as curator, managing the global model lifecycle
 - Coordinates model distribution and update collection
 - Performs validation and aggregation of local updates
 - Uses Federated Averaging for model aggregation
 - Manages training rounds until convergence

Model Predictions: The resulting global model captures collective traffic patterns while preserving individual privacy. Each vehicle can then use this model with its recent local data to make predictions about traffic conditions. This distributed approach enables real-time decision making while maintaining data privacy. We should keep in mind that the real-world timing constraints require efficient processing. The entire interaction between vehicles and RSUs should complete within brief encounters (e.g., while stopped at a traffic light). This adds practical performance requirements to our security considerations.

3.3 Summary

In this chapter, we presented a compelling use case for FL-based data sharing in the IoV context, using traffic trajectory prediction and road sign classification as concrete examples. We demonstrated how FL enables privacy-preserving collaborative learning among vehicles while addressing the unique security challenges of dynamic vehicular networks. The process flow was detailed in three phases, highlighting the advantages of the FL-based approach.

This use case sets the stage for our security analysis in subsequent chapters, where we'll examine potential vulnerabilities in this centralized FL approach and explore how blockchain technology can address these security challenges. Key areas we'll investigate include authentication and authorization mechanisms, data and model integrity protection, privacy preservation during model sharing, resistance to malicious participants, and scalability of secure operations.

By understanding how these components interact in a practical scenario, we can better analyze the security requirements and design appropriate countermeasures. The following chapters will delve into the details of our threat modeling, risk assessment, and proposed blockchain-based solution for secure FL in IoV environments.

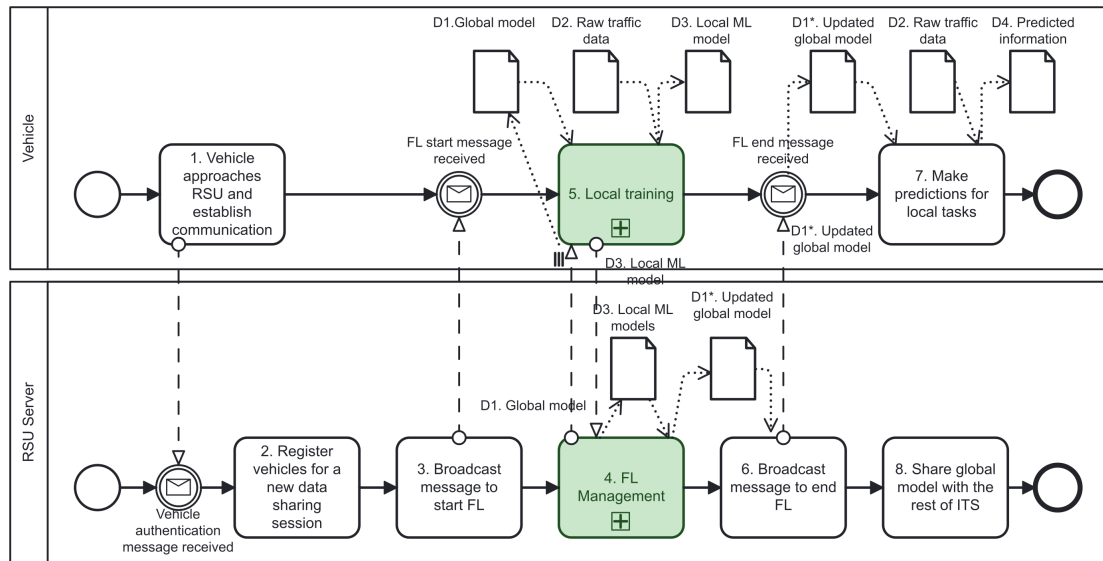


Figure 2. BPMN diagram of data sharing in IoV with the help of centralized FL.

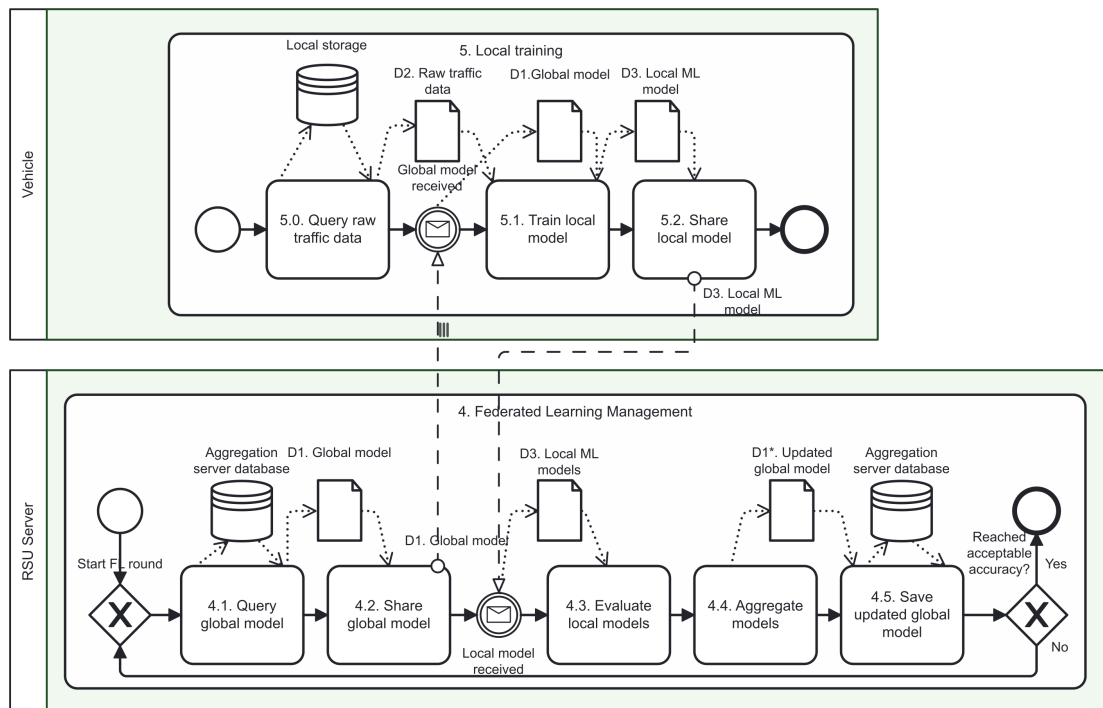


Figure 3. BPMN diagram of the detailed FL process.

4 Literature Review

This chapter presents a comprehensive review of security aspects in FL, particularly focusing on its application in the IoV. In this section, we answer the research question **RQ1: What are the security risks of centralized FL data sharing?** To conduct this review systematically, we employed a snowballing sampling technique that allowed us to thoroughly explore the current state of research in this domain.

Our review process began with analyzing several comprehensive surveys [9, 13, 18, 19, 20] that provided foundational understanding of security challenges in FL and IoV systems. Using these surveys as seed papers, we followed both forward and backward snowballing approaches. Forward snowballing involved identifying newer papers that cited our seed papers, while backward snowballing examined the references within our initial set of papers. This iterative process helped us uncover relevant research that might have been missed in a traditional keyword-based search.

For instance, starting from Ferrag et al.’s [9] comprehensive survey of federated deep learning for cybersecurity in IoT applications, we discovered several key papers discussing specific attack vectors such as model poisoning attacks [21]. Similarly, following references from Chhetri et al.’s [13] survey on blockchain-based FL led us to explore various defense mechanisms and their implementations.

This systematic approach allowed us to identify two main research streams: (1) the evolution of attack methodologies targeting FL systems, and (2) the development of defense mechanisms, particularly those leveraging blockchain technology. We organize our findings into two main sections:

First, we examine various attack vectors targeting centralized FL systems, categorizing them based on their impact on system security and privacy. Second, we analyze existing defense mechanisms, with particular attention to blockchain-based solutions and their effectiveness in mitigating identified threats. This structured analysis aims to provide a comprehensive understanding of the current security landscape in FL-based data sharing for IoV systems.

Our literature review reveals that while FL offers promising solutions for privacy-preserving data sharing in IoV, it faces significant security challenges that require careful consideration and robust countermeasures. The findings from this review inform our subsequent research design and the development of our proposed blockchain-based solution.

4.1 Attacks

In this section, we will delve into a detailed examination of the existing categories of attacks that target the FL process. Our focus will be on categorizing and describing these attacks based on their impact on privacy, integrity, and availability of FL systems.

By understanding these attack vectors, we can set the stage for discussing defense mechanisms to tackle them.

4.1.1 Poisoning Attacks

Poisoning attacks appear before or during the training and violate the integrity of the models. They can be broadly categorized into data poisoning, model poisoning (injection), and backdoor attacks. All the attack variants can result in manipulated model predictions that align with the malicious actors' desired outcomes. The success of such attacks compromises the integrity of FL processes and undermines the reputation and credibility of IoV systems.

Data poisoning attacks. In data poisoning attacks, an adversary with access to the training data manipulates it to induce undesirable behavior in the trained model. The attack vector typically involves injecting malicious data into the training dataset, exploiting security weaknesses such as lack of data validation and insufficient monitoring of training data. This could be done by compromising the data collection and transmission systems of multiple connected vehicles. The attacker could also manipulate the data labeling process, such as by falsifying the labeling process. Alternatively, they might exploit insider threats by bribing technicians or data annotators involved [22].

Model poisoning attacks. In model poisoning attacks, malicious actors manipulate the model's parameters or updates directly, causing it to behave in undesirable ways. The threat actors can be external individuals or organizations with the knowledge and resources to manipulate deep learning models or even malicious insiders within the organization developing the model. These attacks exploit security weaknesses such as insufficient access controls to the model's code and parameters, lack of proper secure coding practices, and inadequate monitoring of the model's activity [23].

In the work of Cao and Gong [21], model poisoning attacks were divided into untargeted and targeted attacks. Untargeted attacks aim to reduce the overall accuracy of the global model. For instance, the MPAF (Model Poisoning Attacks on Federated Learning based on Fake Clients) attack allows an attacker to inject fake clients that send carefully crafted model updates. These updates are designed to drag the global model towards a malicious base model with poor accuracy. Meanwhile, in targeted attacks, the adversary attempts to cause misclassification for specific inputs while maintaining good performance on other inputs. For example, in an IoV context, an attacker might try to make the model misclassify a particular type of vehicle or road sign.

Transfer learning and model skewing attacks can be considered as model poisoning variations. In transfer learning attacks, adversaries exploit the process of adapting pre-trained models to new tasks, potentially introducing malicious behaviors during fine-

tuning. Model skewing attacks, on the other hand, involve manipulating the distribution of training data to bias the model’s behavior [24, 25].

Backdoor attack. This is a type of data poisoning attack that aims to manipulate a subset of the training data, making machine learning models trained on this tampered dataset are vulnerable to test sets with similar embedded triggers. In a backdoor scenario, an attacker aims for the FL process to produce a joint model that achieves high accuracy on both its main task and an attacker-chosen backdoor subtask. The attacker also seeks to ensure that the model retains high accuracy on the backdoor subtask for multiple rounds after the attack. In contrast, traditional data poisoning attacks aim to degrade the model’s performance across large parts of the input space, while Byzantine attacks seek to prevent the model from converging [26, 27].

The attacker can simply train its model on backdoored inputs, but this naive approach is ineffective against FL. The aggregation process cancels out most of the backdoored model’s contributions, causing the joint model to quickly forget the backdoor. Additionally, the attacker would need to be frequently selected, making the poisoning process slow. In contrast, model replacement ensures that the attacker’s contribution survives averaging and is transferred to the global model. This method is a single-shot attack, where the global model exhibits high accuracy on the backdoor task immediately after being poisoned. In addition, a distributed backdoor attack may be even more persistent and effective than a centralized one [26, 27].

4.1.2 Free Rider Attacks

Free rider attacks are a type of attack on FL systems where malicious clients attempt to benefit from the collaborative learning process without contributing valid data or computational resources. Lin et al. first introduced the concept of free rider attacks in FL. The key motivation for free rider attacks is to deceive the system to gain rewards or access to the final global model without incurring the costs of participation.

In the paper[28], the authors introduce two main types of attacks: random weights and delta weights attacks. During random weights attack the attacker generates random values within a plausible range to construct fake gradient updates. This simple approach attempts to mimic legitimate gradient updates. Meanwhile, during delta weights attack, the attacker constructs gradient updates by subtracting the global model weights from two consecutive rounds. This more sophisticated attack leverages information from the FL process to generate more realistic fake updates. Free rider attacks can be further enhanced by adding Gaussian noise to the fake gradient updates. This makes the updates appear more natural and helps avoid detection, especially when multiple free riders are present.

These attacks significantly impact FL by causing unfair resource allocation and rewards, degrading model performance when a large portion of clients are free riders,

and violating the collaborative nature of the learning process. As FL systems become more widely deployed, especially in scenarios with incentive mechanisms, free rider attacks pose an emerging threat that must be addressed.

4.1.3 Model Inversion Attacks

Model inversion attacks aim to reverse-engineer or reconstruct private training data from the shared model information (updates, output logits, gradients, or parameters) in FL systems [29]. These attacks pose a significant privacy risk by potentially revealing sensitive information about individual participants' training data. Model inversion attack effectiveness varies based on model architecture, training procedures, and aggregation mechanisms. Deeper models and increased batch sizes typically enhance privacy. However, advanced attacks can still extract sensitive information in certain FL scenarios despite these protective measures. The key model inversion attacks discussed in the literature include:

Gradient inversion attacks. These attacks aim to reconstruct input data from the gradients shared during FL updates. These attacks work by minimizing the difference on the loss function between the received gradients (calculated on a client's local dataset) and the gradients of synthesized data [30, 31].

Parameter-based inversion attacks exploit the parameters of the target model. For example, attackers can use a generative adversarial network (GAN) with built-in local models to recover private data [30]. GAN approaches have been employed to synthesize plausible training samples that match the shared gradients or updates [32]. While these methods may not perfectly reconstruct private data, they can reveal general characteristics of the original data.

Notable works include Deep Leakage from Gradients (DLG) [33], which optimizes dummy inputs and labels to match shared gradients, enabling the reconstruction of private training samples. Its improved version [34] analytically recovers ground truth labels. See Through Gradients [35] is an enhanced attack capable of reconstructing high-fidelity images and restoring ground truth labels from gradients. The Gradient Leakage Evaluation Framework [36] offers a systematic approach to assess various gradient leakage attacks, including analytical methods for specific network architectures.

Confidence-based inversion attacks. In FedMD⁶, a novel attack method called Paired-Logits Inversion (PLI) demonstrates that private data can be reconstructed solely

⁶FL with Model Distillation (FedMD) is an enhanced FL approach where each party uses both its private data and a public dataset for local training, sending predicted logits on the public dataset to the server. In FedMD, the server can only access these predicted logits on the public dataset, preventing leakage from gradients or parameters of local models.

from shared public knowledge without access to gradients or direct private information. This attack exploits the “confidence gap” between predictions made by local models trained on private data and server models that lack this private information [30].

Another research [37] reveals additional vulnerabilities in model confidence patterns. The authors observed that models typically exhibit larger confidence gaps for positive examples compared to negative ones, especially in class-imbalanced problems like disease prediction or ad conversion. This asymmetry can be exploited by analyzing cosine similarities between examples: same-class examples tend to have positive similarities, while different-class examples have negative similarities. In imbalanced datasets, which are common in many real-world applications. This pattern allows attackers to infer labels without needing exact class distributions.

4.1.4 Membership Inference Attacks

These attacks aim to determine whether a specific data record was used to train a given machine learning model, effectively breaching the privacy of individuals whose data was used in the training process [38]. It’s important to distinguish membership inference attacks from other techniques like model inversion. While model inversion aims to retrieve information about inputs or class features from a model’s output, it doesn’t actually determine if a the specific record was used in training. The key difference lies in the specificity of the information being extracted: membership inference provides a binary answer about a record’s presence in the training data, whereas model inversion attempts to reconstruct general characteristics of the input or class [39].

The core idea behind membership inference is to exploit differences in how machine learning models behave on data they were trained on versus data they haven’t seen before. Attackers typically frame this as a classification problem, training an “attack model” to distinguish between the target model’s behavior on training inputs and its behavior on inputs not used in training [39].

In **black-box** scenarios, where attackers only have access to the model’s predictions, more sophisticated techniques are required. One such approach is “shadow training,” where multiple imitation models are created to mimic the behavior of the target model. These shadow models are trained on known datasets, allowing the attacker to generate labeled examples of member and non-member behaviors. This labeled data is then used to train the attack model, which can subsequently be applied to the actual target model [39, 40].

In **white-box** scenarios, on the other hand, attackers can leverage internal model information such as gradients when available. In the context of FL white-box membership inference, an adversary usually performs attacks during aggregation as the central server, as it receives the updates from local models with known architecture. Recent techniques like CS-MIA have shown that by analyzing confidence scores over multiple rounds of FL, attackers can improve their inference accuracy. However, their effectiveness can

diminish as the number of participants increases, making it harder to isolate the influence of any single participant's data [40].

Another paper [41] introduces a novel perspective on privacy attacks in FL, focusing on subject membership inference rather than individual data point inference. The authors argue that in cross-silo FL, an adversary is more likely to have access to a subject's data distribution rather than specific records, and a subject's data may be spread across multiple organizations. They propose two new black-box attacks for subject membership inference and demonstrate their effectiveness on real-world data and synthetic federation configurations. The study reveals that these attacks can be powerful even with limited information and without access to exact training records.

4.1.5 Communication Attacks

Another group of communication-based attacks exploits vulnerabilities across various layers of the FL system in vehicular networks, including physical, transport, network, and authentication/authorization layers. These attacks can be categorized into the following types: identity-based, message modification and replay-based, and network availability-based attacks. Each type targets different aspects of the communication process, potentially compromising the entire FL system. These attacks pose significant threats to the key security triad of system integrity, data confidentiality, and resource availability.

Identity based attacks. Involve an attacker impersonating another node in a network by changing its identity. By impersonating trusted nodes, attackers take advantage of the network's inherent trust mechanisms. Attackers can manipulate identification data (identity spoofing) such as IP addresses, MAC addresses, or application-layer identities to masquerade as authorized entities. These attacks often involve eavesdropping on legitimate communications to gather information or to insert malicious data. Some variants involve creating numerous fake identities to amplify the attack's impact or to manipulate consensus in the network. The ultimate goal is often to undermine the integrity of the entire system by introducing unreliable or malicious data through seemingly legitimate sources. In the FL process, it can lead to privacy violations (sensitive data exposure) and the injection of false model updates, compromising the learning process and the resulting global model [9, 18].

Message modification based attacks. These attacks involve malicious nodes interfering with the normal flow of network communications in various ways. The attacker may modify legitimate messages passing through their node, delete important packets entirely, or inject fabricated messages into the network. The common goal is to disrupt proper network functioning and compromise data integrity. In an output integrity

attack scenario, an attacker aims to modify or manipulate the output of a machine learning model in order to change its behavior or cause harm to the system it is used in [42]. By manipulating message content or flow, attackers can redirect traffic, drop critical information, spread false data, or gain unauthorized access to confidential data. This class of attacks exploits a lack of traceability, encryption, and identity management in the communication protocols. It can lead to serious consequences such as denial of service, misinformation propagation, privacy breaches, and overall degradation of network performance and reliability. The unpredictable and diverse nature of these attacks makes them particularly challenging to detect and mitigate, requiring robust security measures at multiple levels of the network architecture [18].

Replay message-based attacks Replay message-based attacks in vehicular networks involve the malicious reinjection of previously transmitted data into the network. In these attacks, adversaries capture and store valid messages from earlier communications, then retransmit them at a later time. This reintroduction of old data can have several negative impacts on the network: bandwidth consumption, deception of legitimate nodes, traffic manipulation, and data control. The key characteristic of these attacks is the exploitation of message validity without considering message freshness. By recycling once-legitimate data, attackers can manipulate network behavior while maintaining an appearance of authenticity. This makes such attacks particularly challenging to detect and mitigate, as the messages themselves may pass standard security checks. The goal is often to disrupt normal network operations, redirect traffic, or gain unauthorized control over data flow, all while evading detection by using seemingly valid messages [19, 18].

Network availability-based attacks. Network availability-based attacks in vehicular networks aim to disrupt or prevent legitimate users from accessing network resources and services. The most common types of attacks affecting network availability are DoS, DDoS, and jamming attacks. These attacks include resource (bandwidth, memory, or energy) exhaustion, service disruption, configuration tampering, and traffic flooding. The effects can range from slowing down network performance to completely blocking message exchange between vehicles. The overall objective of these attacks is to compromise the availability and reliability of the vehicular network, potentially leading to serious consequences in time-sensitive and safety-critical applications. Their effectiveness often lies in exploiting the open nature of vehicular networks and the limited resources of individual nodes [19, 18, 9].

4.1.6 Third-Party Software Attacks

Third-party software attacks exploit the reliance of developers on external code libraries and frameworks, which may contain vulnerabilities or malicious components. These

attacks are particularly concerning in the context of FL systems for vehicular networks, given their complex architecture and integration of various third-party components. The primary types of attacks in this category are AI supply chain attacks and arbitrary code execution attacks, both of which can compromise the integrity and privacy of the FL process.

AI supply chain attacks. AI supply chain attacks occur when an adversary modifies or replaces a machine learning library, model, or associated data used within a system [43]. These attacks exploit the trust placed in third-party resources and can have far-reaching consequences, such as data theft, result manipulation, and model failure.

For example, an attacker might identify a project’s dependencies on specific open-source packages and libraries. By modifying the code of one of these packages and uploading the altered version to a public repository, the attacker can introduce malicious code into the target system. When an unsuspecting organization downloads and installs this compromised package, the malicious code is deployed alongside legitimate components. Without rigorous package verification processes, these vulnerabilities may remain undetected for extended periods, potentially compromising multiple systems and organizations.

Arbitrary code execution attacks. Arbitrary code execution attacks exploit software vulnerabilities within client applications, allowing attackers to execute arbitrary code. Such vulnerabilities often arise from insecure coding practices, leading to unexpected application behavior. By executing their own code on a target system or process, attackers can gain unauthorized access or perform unauthorized actions, potentially resulting in data breaches, data corruption, or complete system takeover [44].

A notable example of this vulnerability was recently discovered in PyTorch, a popular deep learning framework. The vulnerability allowed for arbitrary code execution due to the unsafe use of the `eval` function, highlighting the potential risks even in widely used and well-maintained software⁷.

4.2 Defense Mechanisms

To address the security and privacy challenges identified in FL systems, researchers have developed various defense mechanisms. These approaches aim to protect both the training process and the resulting models while maintaining system efficiency and effectiveness. In this section, we explore key defense strategies that have emerged. Each mechanism addresses specific vulnerabilities while often complementing other security measures to create comprehensive protection frameworks.

⁷<https://github.com/advisories/GHSA-47fc-vmwq-366v>

4.2.1 Authorization and Authentication

In IoV environments, while anonymous participation helps protect privacy, it also creates opportunities for malicious actors. Robust identity verification mechanisms are therefore essential for secure data sharing. Current approaches include Public Key Infrastructure (PKI) for managing digital certificates and two-way authentication protocols to verify communicating parties [45]. Song et al. [46] propose a fog computing-based authentication system using cluster heads for vehicle validation and monitoring. Building on this, Tian et al. [47] developed an integrated authentication architecture using trusted edge nodes as intermediaries between vehicles and registration servers, incorporating time tree networks for efficient data sharing. These mechanisms collectively enable secure participation while maintaining vehicle anonymity.

4.2.2 Blockchain

Blockchain technology offers several key characteristics that make it particularly suited for enhancing security in FL systems. These include decentralization to eliminate single points of failure, immutable ledgers for tracking model updates, consensus mechanisms for ensuring data integrity, and smart contracts for automating security protocols [45, 48]. The integration of blockchain with FL has shown promising results in addressing security concerns while maintaining system efficiency.

Recent surveys provide key insights into blockchain-FL architectures, categorizing them as decoupled (storage-focused), coupled (smart contract-based), or overlapped (blockchain nodes as trainers) [49]. Analysis of design patterns reveals that stronger fairness mechanisms tend to improve system robustness [50], while blockchain effectively addresses centralization and trust issues [51]. BAFFLE [52] eliminated the need for central aggregation through smart contracts, while Lu et al. [4] transformed data sharing into a verifiable learning problem.

Different blockchain platforms offer distinct advantages for FL implementation. Hyperledger Fabric provides efficient consensus mechanisms suitable for IoV applications [53], while Ethereum's robust smart contracts support general FL scenarios [54]. However, challenges remain in optimizing tradeoffs between robustness, efficiency, privacy, and fairness, particularly in areas of model compression, asynchronous aggregation, and cross-device compatibility [50].

Several implementations have explored blockchain-based FL for vehicular networks. Lu et al. [1] proposed PermiDAG, combining permissioned blockchain with local DAGs for asynchronous FL. Wang et al. [14] developed a consortium blockchain system using RSUs as preselected nodes with reputation-based sharing. Both approaches leverage blockchain's capabilities to enhance security while maintaining efficiency in dynamic vehicular environments.

Another research direction focuses on accountability and incentivization in FL.

Mugunthan et al. [54] and Li et al. [55] introduced frameworks that reward quality contributions while protecting privacy.

4.2.3 Zero-Knowledge Proof

Zero-knowledge proofs (ZKPs) are cryptographic protocols that enable a prover to convince a verifier about the validity of a statement without revealing any additional information beyond the statement's truth. A valid ZKP protocol must satisfy three fundamental properties: completeness (an honest verifier accepts true statements when both parties follow the protocol), soundness (a cheating prover has negligible probability of convincing the verifier of a false statement), and zero knowledge (the verifier learns nothing except the statement's validity). The field has evolved from early interactive protocols requiring bounded verifiers to more practical non-interactive zero-knowledge (NIZK) proofs, leading to succinct non-interactive arguments of knowledge (SNARKs). In the context of FL, ZKPs allow a prover (typically the aggregation server) to demonstrate to verifiers (participating clients) that specific computations were performed correctly, without disclosing the underlying data or intermediate results [56].

A significant application of ZKPs in FL is verifiable model aggregation. The aggregation server can generate proofs demonstrating that it correctly executed the federated averaging algorithm on client updates while maintaining the privacy of individual contributions. For instance, the zkDFL framework proposed by Ahmadi and Nourmohammadi utilizes ZKPs to prove both the correctness of aggregation computations and the inclusion of all client updates in the final result. Their approach leverages smart contracts for on-chain verification of these proofs, enabling clients to independently validate the aggregation process [56].

In a notable approach, Nguyen and Thai [57] propose a framework that combines secure aggregation with ZKPs to enable both privacy preservation and backdoor detection in FL. Their key innovation lies in shifting backdoor detection responsibility from the server to the clients while maintaining model privacy. Each client generates a ZKP demonstrating their model passes a pruning-based backdoor detection test without revealing actual model parameters. This elegantly resolves the conflict between privacy preservation and security validation, allowing the server to verify that local models are clean without actually seeing them.

The implementation of ZKPs for machine learning computations has been facilitated by advances in efficient arithmetization techniques. Angel et al. introduced Otti, a compiler that can generate efficient ZKP circuits for numerical optimization problems, including stochastic gradient descent used in neural network training [58]. In the context of vehicular networks, Smahi et al. demonstrated the practical application of zkSNARKs for verifying model updates (Federated SVM) in a blockchain-enabled FL system, showing their effectiveness in preventing malicious contributions while preserving privacy [59].

Recent research has explored various implementations of ZKP systems for FL. Zhang et al. developed a blockchain-based framework using ZKPs to verify the randomness of noise addition in differential privacy mechanisms, demonstrating practical verification times of around 2.27 seconds on the Hyperledger Fabric platform [53].

4.2.4 Differential Privacy

Differential Privacy (DP) provides mathematical guarantees for privacy preservation in FL by adding controlled noise to data or model updates. Initially introduced to deep learning by Abadi et al. [60], DP has evolved to address practical challenges in federated settings. Recent advances include adaptive optimization methods [61] and noise-resilient training approaches [62] that improve the privacy-utility trade-off. When implemented in IoV scenarios, DP can be applied at local (vehicle), central (RSU), or global levels, though achieving strong privacy guarantees may significantly impact model performance [41]. Integration with other technologies like blockchain and ZKPs has shown promise in providing verifiable privacy while maintaining practical efficiency [53].

4.2.5 Secure Aggregation

Secure aggregation (SA) represents a privacy-preserving mechanism in FL systems that enables the computation of aggregate statistics across multiple participants while protecting individual contributions. In the FL context, SA allows the server to compute the global model by aggregating local model updates without learning the individual updates from any specific client.

Recent research has made significant strides in improving the efficiency and security of SA protocols. Ma et al. [63] introduced Flamingo, a novel system designed specifically for multi-round secure aggregation in FL settings. Unlike previous protocols that require per-round setup, Flamingo implements a more efficient approach with lightweight dropout resilience.

Building upon these developments, Guan et al. [64] proposed OPSA, a multi-round one-pass secure aggregation framework that leverages Trusted Execution Environment (TEE) technology, achieving efficient communication and streamlined computation while providing verifiable aggregation.

A significant challenge in secure aggregation protocols is handling client stragglers - participants who respond slowly or drop out during the process. Taiello et al. [65] address this issue through two novel approaches: Eagle, a synchronous SA scheme designed to be independent of client failures, and Owl, the first SA solution suitable for asynchronous settings. Their implementation demonstrates significant performance improvements, with Eagle outperforming Flamingo by a factor of four in scenarios with realistic dropout rates.

4.2.6 Robust Training

Robust training represents a significant defensive approach in FL systems, focusing on enhancing model resilience against various adversarial attacks. The robust training techniques can be integrated into FL systems to create more resilient models that better withstand poisoning attempts and other adversarial attacks, which is especially important in critical applications like vehicular networks.

A significant advancement in robust training comes from Lecuyer et al. [66], who establish a connection between differential privacy and robustness against adversarial examples. Their PixelDP approach provides certified robustness guarantees that scale to large networks and datasets.

Chen et al. [67] propose a novel cost-aware approach that considers the asymmetric costs associated with feature manipulation in security applications. While their work focuses on tree ensembles, the underlying principles can be adapted to FL scenarios, where the cost of manipulating different features varies significantly.

4.2.7 Anomaly Detection

Several techniques have been proposed to detect malicious behavior in FL systems. Autoencoder-based detection identifies outliers in client updates through reconstruction error analysis [28, 68], though this approach struggles with sophisticated attacks. More advanced methods include Deep Autoencoding Gaussian Mixture Model (DAGMM) for high-dimensional analysis [69] and its enhanced version STD-DAGMM, which incorporates update standard deviation [28]. While these methods show promise in identifying suspicious updates by learning legitimate contribution patterns, they face challenges with sophisticated attacks and scenarios involving high proportions of malicious clients. Key considerations include efficiently handling high-dimensional data, adapting to non-IID distributions, and balancing detection accuracy with computational overhead.

4.2.8 Other

In addition to the main defense mechanisms discussed earlier, researchers have explored many other approaches to enhance the security and privacy of FL systems.

FL with Model Distillation (FedMD) enhances privacy by having participants share only predicted logits on public datasets rather than raw gradients or parameters [30]. This distillation approach creates an effective barrier against privacy attacks while enabling efficient collaboration. Mutual information regularization limits prediction-contained information about inputs, making model inversion attacks more difficult [70], while representation learning-based protection techniques like Soteria perturb data representations to protect against reconstruction attacks while maintaining performance [71]. Suspicion-based fault tolerance mechanisms such as Zeno enable SGD to handle

arbitrary numbers of faulty workers through ranking-based preference systems [68], and reputation tracking helps combat Sybil attacks by monitoring participant behavior over time [72].

These complementary approaches can be combined with primary security measures like differential privacy and secure aggregation to create comprehensive protection frameworks. The specific combination of defenses typically depends on application requirements, including computational resources, privacy needs, and anticipated threats.

4.3 Summary

This literature review has systematically examined the security landscape of FL systems, analyzing both attack vectors and defense mechanisms. Our analysis reveals several key findings:

The attack surface in FL is extensive and multifaceted. Data and model poisoning attacks can compromise training integrity, while sophisticated inference attacks like model inversion and membership inference can breach privacy even without direct access to training data. Communication-based attacks exploit vulnerabilities across multiple system layers, and third-party software attacks leverage weaknesses in external dependencies. This complex threat landscape requires a multi-layered approach to security.

To counter these threats, researchers have developed various defense mechanisms. Authorization and authentication form the foundation of security, while differential privacy and secure aggregation protect individual contributions. Anomaly detection helps identify malicious behavior, and robust training techniques enhance model resilience. However, many of these sophisticated defenses introduce significant computational overhead and complexity, making them challenging to implement in resource-constrained IoV environments.

Blockchain technology emerges as a particularly promising general-purpose solution, offering decentralization, immutability, and transparent consensus mechanisms that address many inherent vulnerabilities in centralized FL architectures. Among blockchain implementations, permissioned platforms like Hyperledger Fabric appear especially suitable for IoV scenarios due to their efficient consensus mechanisms and fine-grained access control. Zero-knowledge proofs complement blockchain by enabling verifiable computations while preserving privacy, particularly in the context of model aggregation.

For the remainder of this thesis, we will focus primarily on permissioned blockchain and ZKP-based aggregation verification, as they offer robust, general-purpose solutions that can be practically implemented while addressing core security requirements. This understanding of both threats and defenses will guide our subsequent development of a blockchain-based security enhancement for FL in IoV environments, ensuring we address the most critical vulnerabilities while maintaining system efficiency and practicality.

5 Security Risks Assessment

This section presents the security risk analysis of our case study regarding the centralized FL-based data sharing scenario of IoV. In this section, we answer the research question **RQ2: How to build a threat model using blockchain technology to mitigate the security risks of centralized FL data sharing in the context of IoV?**

We begin by introducing threat modeling as a systematic approach to identifying potential security vulnerabilities and attack vectors in our system. Following the threat modeling process, we will employ a security risk management model to identify components, assets, security risks, and countermeasures relevant to our use case. This comprehensive approach will allow us to thoroughly assess the security landscape of our centralized FL scenario in the IoV context.

Building on this analysis, we then explore how blockchain technology and ZKPs can serve as countermeasures to the identified security risks. This treatment-related concepts section will establish the foundation for our proposed blockchain-based solution design and implementation detailed in subsequent chapters, demonstrating how these technologies can enhance the security and trustworthiness of FL-based data sharing in IoV environments.

5.1 Threat Model for Centralized FL

Threat modeling is a proactive method used to find, list, and prioritize possible security threats, helping to create strong defenses. It is an important part of identifying risks in security risk management.

One key part of identifying threats is using a structured system for categorization. A common framework for this is called STRIDE, which includes six types of threats: Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, and Elevation of Privilege. The goal of categorizing threats in this way is to understand them from the attacker's point of view. Tools like data flow diagrams (DFDs) can help show where threats might target things like data sources, processes, data flows, and interactions in FL-based data sharing scenarios.

Threats can be further organized using threat trees, each representing a specific threat goal. Lists of common threats, with examples, can help in finding risks. Additionally, use and abuse cases illustrate how existing security measures might be circumvented or where vulnerabilities exist.

In order to create a threat model for centralized FL data sharing in the IoV, we used the STRIDE approach to brainstorm and identify possible attack vectors. This process involved gathering potential attacks and vulnerabilities based on a review of the existing literature about FL-related attacks in Chapter 4 and more general IoV attacks in CAPEC and ATT&CK taxonomies. The resulting threat model for centralized FL data sharing in the IoV is depicted in Figure 4. This model outlines 20 possible threats that an attacker

could exploit during the data sharing process, organized into six categories based on STRIDE:

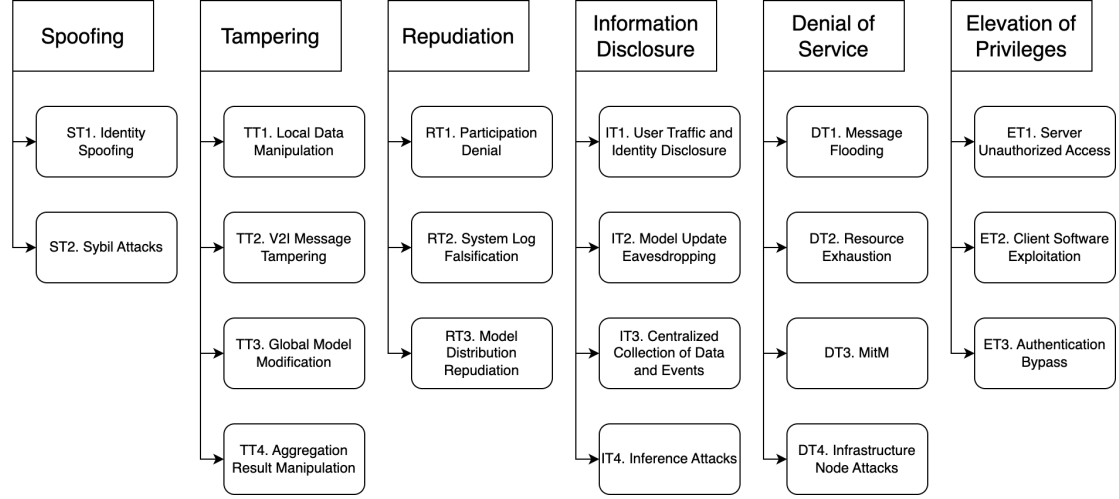


Figure 4. Threat model for the centralized FL data sharing in IoV.

Spoofing:

1. **ST1. Identity Spoofing:** An attacker impersonates a legitimate participant (vehicle or server) in the FL process, aligning with the Identity Spoofing (CAPEC-151 [73]) and Valid Accounts (ATT&CK-T1078 [74]) techniques. This could lead to unauthorized access to sensitive data or injection of malicious model updates [18].
2. **ST2. Sybil Attacks:** An attacker creates multiple fake identities to gain disproportionate influence in the FL process, aligning with Valid Accounts (ATT&CK-T1078 [74]) technique. This could manipulate the aggregation process and skew the global model towards the attacker's desired outcome [18, 72].

Tampering:

1. **TT1. Local Data Manipulation:** An attacker modifies the training data on a local device before model training, also known as Data Poisoning (OWASP-ML02:2023 [22]), Model Skewing (OWASP-ML08:2023 [24]), or Transfer Learning Attack (OWASP-ML07:2023 [25]). This could introduce biases or backdoors into the local model, potentially affecting the global model [18]. This threat is relevant to File Manipulation (CAPEC-165 [75]) and Runtime Data Manipulation (ATT&CK-T1565.003 [76]).

2. **TT2. V2I Message Tampering:** An attacker intercepts and modifies model updates, also known as Model Poisoning (OWASP-ML10:2023 [23]), or coordination messages during transmission. This tactic aligns with Transmitted Data Manipulation (ATT&CK-1565.002 [77]) and may involve Adversary-in-the-Middle (ATT&CK-T1557 [78]) techniques. This could introduce malicious changes to the model and mislead the FL participants [18].
3. **TT3. Global Model Modification:** An attacker with access to the central server alters the global model directly by overwriting the data in the central storage. This action, aligning with File Manipulation (CAPEC-165 [75]) and Stored Data Manipulation (ATT&CK-T1565.001 [79]), can introduce widespread vulnerabilities or biases into the entire FL system [18].
4. **TT4. Aggregation Result Manipulation:** An attacker interferes with the aggregation process to skew the results, even when individual updates are legitimate [24, 57]. This tactic aligns with Software Integrity Attack (CAPEC-184 [80]), File Manipulation (CAPEC-165 [75]), and Runtime Data Manipulation (ATT&CK-T1565.003 [76]), allowing the attacker to compromise the global model's accuracy and reliability.

Repudiation:

1. **RT1. Participation Denial:** A participant denies involvement in a specific round of FL, complicating accountability and obscuring the source of malicious updates. This aligns with Indicator Removal (ATT&CK-T1070 [81]) and is a form of a Repudiation Attack (OWASP [82]), where the attacker evades responsibility by disassociating themselves from the actions taken.
2. **RT2. System Log Falsification:** An attacker alters system logs to conceal malicious activities or create false records of legitimate actions. This tactic hinders investigations and hides traces of the attack, aligning with Log Injection/Tampering/Forging (CAPEC-93 [83]), File Manipulation (CAPEC-165 [75]), and Stored Data Manipulation (ATT&CK-T1565.001 [79]).
3. **RT3. Model Distribution Repudiation:** The central server denies distributing a specific version of the global model, causing potential inconsistencies and disputes among participants over the correct model version. This aligns with Indicator Removal (ATT&CK-T1070 [81]) and represents a Repudiation Attack (OWASP [82]), where accountability is evaded by denying involvement.

Information Disclosure:

1. **IT1. User Traffic and Identity Disclosure:** An attacker gains unauthorized access to sensitive training data stored on local devices, potentially leading to privacy breaches and exposure of personal or proprietary information. This aligns with the Exploiting Incorrectly Configured Access Control Security Levels (CAPEC-180 [84]) technique, where inadequate access controls are leveraged to obtain unauthorized data access.
2. **IT2. Model Update Eavesdropping:** An attacker intercepts model updates during data or message transmission, potentially revealing sensitive information about the training data or the target vehicle. This aligns with both the Adversary in the Middle (CAPEC-94 [85]) attack pattern and the Adversary-in-the-Middle (ATT&CK-T1557 [78]) technique, where an attacker positions themselves between communicating parties to intercept or modify transmitted data.
3. **IT3. Centralized Collection of Data and Events:** The central server gathers model updates and actions from FL participants for future analysis. This centralization can lead to the misuse of information and creates a single point of failure, potentially exposing sensitive data if the server is compromised. This issue relates to Adversary-in-the-Middle (ATT&CK-T1557 [78]), where an attacker may intercept communications, and Automated Collection (ATT&CK-T1119 [86]), where an adversary could exploit the centralized data repository to collect sensitive information in bulk.
4. **IT4. Inference Attacks:** An attacker leverages global or local models to infer private information about individual participants, potentially compromising user privacy without needing direct access to the training data. This relates to Model Inversion Attack (OWASP-ML03:2023 [29]), Membership Inference Attack (OWASP-ML04:2023 [38]), and Automated Exfiltration (ATT&CK-T1020 [87]), where sensitive information is obtained through indirect means.

Denial of Service:

1. **DT1. Message Flooding:** An attacker overwhelms the central server with many fake or unnecessary messages, potentially slowing down or crashing the aggregation process and disrupting the entire FL system [18]. This tactic aligns with Flooding (CAPEC-125 [88]), Network DoS: Direct Network Flood (ATT&CK-T1498.001 [89]), and Endpoint DoS (ATT&CK-T1499 [90]).
2. **DT2. Resource Exhaustion:** An attacker targets the computational or storage resources of participating devices, potentially preventing legitimate participants

from contributing to the FL process. This tactic is related to Resource Hijacking (ATT&CK-T1496 [91]), where an adversary seeks to deplete system resources for malicious purposes.

3. **DT3. Man-in-the-Middle (MitM):** An attacker intercepts and potentially alters communications between FL participants and the central server. This interference can disrupt the FL process by manipulating or blocking legitimate communications. This tactic is associated with Software Integrity Attack (CAPEC-184 [80]), Adversary in the Middle (CAPEC-94 [85]), and Adversary-in-the-Middle (ATT&CK-T1557 [78]).
4. **DT4. Infrastructure Node Attacks:** An attacker targets critical infrastructure components, such as RSUs or cellular base stations. This can create blind spots in the IoV network and disrupt peer communication. This tactic is associated with Endpoint DoS (ATT&CK-T1499 [90]) and Cellular Jamming (CAPEC-605 [92]).

Elevation of Privileges:

1. **ET1. Server Unauthorized Access:** An attacker gains unauthorized access to the central aggregation server. This could lead to a complete compromise of the FL process, including manipulation of the global model and access to all participant data. This tactic is related to Exploiting Incorrectly Configured Access Control Security Levels (CAPEC-180 [84]), Authentication Abuse (CAPEC-114 [93]), and Credential Access (ATT&CK-TA0031 [94]).
2. **ET2. Client Software Exploitation:** An attacker exploits vulnerabilities in the software running on vehicles or servers, such as arbitrary code execution, which aligns with AI Supply Chain Attacks (OWASP-ML06:2023 [43]) and Exploitation for Client Execution (ATT&CK-T1203 [44]). This could allow the attacker to manipulate local training or bypass security measures.
3. **ET3. Authentication Bypass:** An attacker circumvents authentication mechanisms to gain unauthorized access to the FL system. This could allow the attacker to impersonate legitimate participants or access restricted system parts. This tactic is associated with Exploiting Incorrectly Configured Access Control Security Levels (CAPEC-180 [84]), Authentication Abuse (CAPEC-114 [93]), and Credential Access (ATT&CK-TA0031 [94]).

This threat model covers the key security concerns in a centralized FL data sharing scenario for IoV, addressing potential vulnerabilities in the communication, computation, and data handling aspects of the system. It can serve as a foundation for developing robust security measures and mitigation strategies.

After providing the threat model of potential attacks and describing the prototype of our use case, let's take a closer look at its components to perform a security risk assessment. According to the ISSRM domain model discussed in Chapter 2.4, we need to define three main groups: asset-related, risk-related, and treatment-related concepts.

5.2 Asset-Related Concepts

In this section, we will address the question of which assets should be protected in our FL scenario for the IoV. We need to identify the appropriate business assets, system assets, and security criteria that must be ensured. In the following subsections, we will detail the specific business assets, IS assets, and security criteria relevant to our FL-IoV scenario. This analysis will provide a foundation for understanding what needs protection and why, informing our subsequent risk assessment and treatment strategies.

5.2.1 Business assets

Asset-related concepts describe what is valuable to an organization and worth protecting. An asset is anything that plays a role in accomplishing the organization's objectives and can be classified as either a business asset or a system asset.

In the context of FL-based data sharing between vehicles and infrastructure, business assets (BA) refer to sensitive data of each participant and intermediate transmitted data between them. This data creates business value for the ITS and must be protected. To achieve this, we can map the BA to the appropriate components of a BPMN diagram (see Figure 2):

1. **Global Model (D1).** This global FL model (or sub-global model for hierarchical FL) encapsulates traffic patterns of individual vehicles in the network and produce useful business value. It is extracted from an aggregation server database and propagated between all participants at the beginning of the FL process. The vehicles train their local models using it as a pre-trained base model. After aggregation, it can be updated and saved for each FL cycle. Later, the global model is used to make the final prediction.
2. **Raw Traffic Data (D2).** This is the data used as input to the machine learning models that bring business value, such as recent vehicle trajectories, sensor data, images from the camera, etc. This records are stored on each vehicle and considered as the most sensitive data of the entire data sharing process.
3. **Local ML Model (D3).** This intermediate machine learning model (and its weight updates) is trained locally on each device participating in the data sharing process. Later, it can be used in the aggregation process after passing the validation.

4. **Predicted Information (D4).** This is the global model output that was predicted using local raw traffic data from individual vehicles. This information provides the final business value for future ITS use cases.

5.2.2 Security criteria

Security criteria characterize security needs and are constraints on business assets. The listed business assets have security needs that constraint them:

1. **Confidentiality.** This criterion ensures that business assets are accessible only to authorized actors. In our scenario, particular emphasis is placed on protecting raw traffic data, which must remain strictly confidential and never be exposed to unauthorized parties. This safeguard is crucial for maintaining privacy and preventing potential misuse of sensitive information.
2. **Integrity.** This criterion guarantees the reliability of business assets by protecting them from tampering and corruption. Asset integrity is a critical characteristic essential for obtaining trustworthy results in both the FL-based data sharing process and subsequent inference operations. Maintaining data and model integrity throughout the entire workflow ensures the reliability and accuracy of the intelligent transportation system's outputs.
3. **Availability.** This criterion ensures that business assets are accessible on demand for normal system functionality. In our scenario, data exchange occurs in close proximity between vehicles and infrastructure. Consequently, asset availability must be guaranteed within the brief time window when vehicles are within range of the RSU. This constraint emphasizes the need for rapid and reliable data transfer to support real-time decision-making in the intelligent transportation system.

5.2.3 System assets

System (IS) assets are components of the information system that support business assets. These can be technological (hardware, software, network) or non-technological (people, facilities) elements that play a role in the system's security.

To support business assets and their objectives, the following system assets and their correspondence to steps on the BPMN diagram (see Figure 2) are introduced:

1. **SA1. Communication Layer:** This network layer organizes messaging and data transfer. It is utilized during steps 1, 3, 4.2, 5.2, 6, and 8.
2. **SA2. Authentication/Authorization Layer:** Central server resources are used to register vehicles in new data sharing sessions and manage network and data access.

This layer typically operates alongside the communication layer during steps 1, 3, 4.2, 5.2, 6, and 8. Additionally, it manages access control for FL participant registration, data querying (steps 2 and 5.0), and global model downloading/uploading from a dedicated database (steps 4.1 and 4.5).

3. **SA3. Server-Side Perception, Compute, and Storage Resources:** Computational resources of the central server manage the global model (D1) lifecycle and orchestrate the training, validation, and aggregation of local models (D3). These assets represent the execution environment and resources used across all server-side stages.
4. **SA4. Aggregation Server Database:** Medium for storing global models (D1) on the central server, associated with steps 4.1 and 4.5.
5. **SA5. Client-Side Perception, Compute, and Storage Resources:** Computational resources in vehicles are used to train and evaluate local machine learning models (D3) based on raw traffic data (D2). These assets represent the execution environment and resources used across all client-side stages.
6. **SA6. Training Process Algorithms and Software:** Algorithms and methods within the client-side execution environment for training local machine learning models associated with step 5.1.
7. **SA7. Evaluation/Validation Process Algorithms and Software:** Algorithms and methods within the server-side execution environment for evaluating (validating) trained local models (D3) performance, associated with step 4.3.
8. **SA8. Aggregation Process Algorithms and Software:** Algorithms and methods within the server-side execution environment for aggregating models (D3) into an updated global model (D1*), associated with step 4.4.
9. **SA9. Model Inference Algorithms and Software:** Algorithms and methods within the client-side execution environment for predicting useful traffic information (D4) using the updated global model (D1*) and raw traffic data (D2) associated with step 7.

It's important to note the relationships between these concepts. System assets support one or more business assets, though not all business assets necessarily have system support. Each business asset can be constrained by zero to several security criteria, and these criteria are used to assess the significance of risks [17].

Table 2 provides a comprehensive overview of all assets in the centralized FL-based data sharing scenario for IoV. This table illustrates how each system asset supports the confidentiality (C), integrity (I), and availability (A) of business assets.

A system asset supports confidentiality if it has read access to the business asset, integrity if it can modify or facilitate modification of the business asset's state, and availability if it can control access to the business asset in subsequent stages of the business process.

Analysis of relations between system assets, business assets, and security criteria:

1. Global (D1) and local models (D3) are dependent on the largest number of system assets. This extensive involvement across multiple stages of the FL process makes them critical components and potential focal points for vulnerabilities and risks. Consequently, a significant portion of our security countermeasures will need to focus on protecting and securing these system assets.
2. Raw traffic data (D2) and predictions (D4), while directly dependent on fewer system assets, are indirectly reliant on all preceding assets in the FL process. This interdependence underscores their importance and the need for comprehensive security measures throughout the entire system.
3. The protection of raw traffic data (D2) and predictions (D4) emerges as our highest priority. Despite their limited direct dependencies, these assets provide the greatest value to the overall Intelligent Transportation System (ITS). Their security is crucial not only for the integrity of the FL process but also for the privacy of individual vehicles and the reliability of the ITS as a whole.

This asset analysis lays the groundwork for our subsequent risk assessment and security strategy development. By understanding the relationships and dependencies between our system and business assets, we can more effectively identify potential vulnerabilities, anticipate risks, and design targeted countermeasures to ensure the security and efficacy of our FL-based data sharing system in the IoV context.

Table 2. Relations between system and business assets

	D1 Global Model	D2 Raw Traffic Data	D3 Local Models	D4 Predictions
SA1	C, I, A	-	C, I, A	-
SA2	C, I, A	C, A	C, A	-
SA3	C, I, A	-	C, I, A	-
SA4	C, I, A	-	-	-
SA5	C, I, A	C, I, A	C, I, A	C, I, A
SA6	C	C	C, I, A	-
SA7	-	-	C, I, A	-
SA8	C, I, A	-	C	-
SA9	C	C	-	C, I, A

5.3 Risk-Related Concepts

Our next step is to identify the risks and vulnerabilities associated with the centralized FL scenario using the ISSRM framework. The integration of FL in the IoV systems preserves privacy while leveraging collective data. However, this decentralized learning paradigm introduces new vulnerabilities and security risks.

In risk-related concepts, a risk combines a threat and vulnerabilities, leading to negative impacts on assets. A risk event consists of a threat exploiting vulnerabilities. A vulnerability is a weakness in a system asset, while a threat is an incident initiated by an agent using an attack method. The impact is the negative consequence of a risk, negating security criteria and harming assets. Impacts can affect both system and business assets. These concepts are interrelated: a risk comprises an event and impacts. Events can lead to multiple impacts, and impacts can result from various events. Impacts harm assets at both business and system levels, potentially violating multiple security criteria [17].

In our FL scenario for IoV, these concepts help us understand how vulnerabilities in the centralized architecture could be exploited by threats, leading to risks that impact the integrity, confidentiality, and availability of our system and business assets.

Our literature review (Chapter 4.1) identified several potential attacks and threats relevant to FL. Now, we will focus on their relationship to the defined assets in our IoV use case, examining how these threats could exploit vulnerabilities to create risk events and the potential impacts these risks could have on our system.

In our analysis of security risks in IoV-based FL, we categorize threats based on their source: client-side, server-side, and intermediary actors. While a comprehensive examination of all threat vectors is important (detailed in Appendix II), we focus primarily on server-side risks in this section. This focus is strategic, as the server infrastructure in IoV systems is more manageable and controllable compared to the diverse and transient nature of vehicle clients.

We turn our attention to the vulnerabilities and security risks posed by a malicious server. For this discussion, we assume that the IoV infrastructure can harm the integrity of the ITS only in the event of a compromised server. However, even without direct compromise, the ITS may violate certain privacy aspects of vehicle clients by exploiting the server's curiosity about clients' sensitive data. In this section, we will explore how malicious servers could attempt to exploit privileged access to ITS resources for unfair advantage or, in more severe cases, completely disrupt the system's operation.

The centralized nature of the FL process in IoV systems introduces several vulnerabilities that can be exploited by dishonest servers. Figure 5 illustrates how a compromised server can execute attacks within the FL process. The diagram highlights two key vulnerability points: the lack of data authenticity verification during model distribution and security breaches during the FL process. Figure 6 provides a detailed view of vulnerabilities during local model training on the vehicle side, specifically demonstrating how the lack of data authenticity verification and model privacy protection mechanisms can be

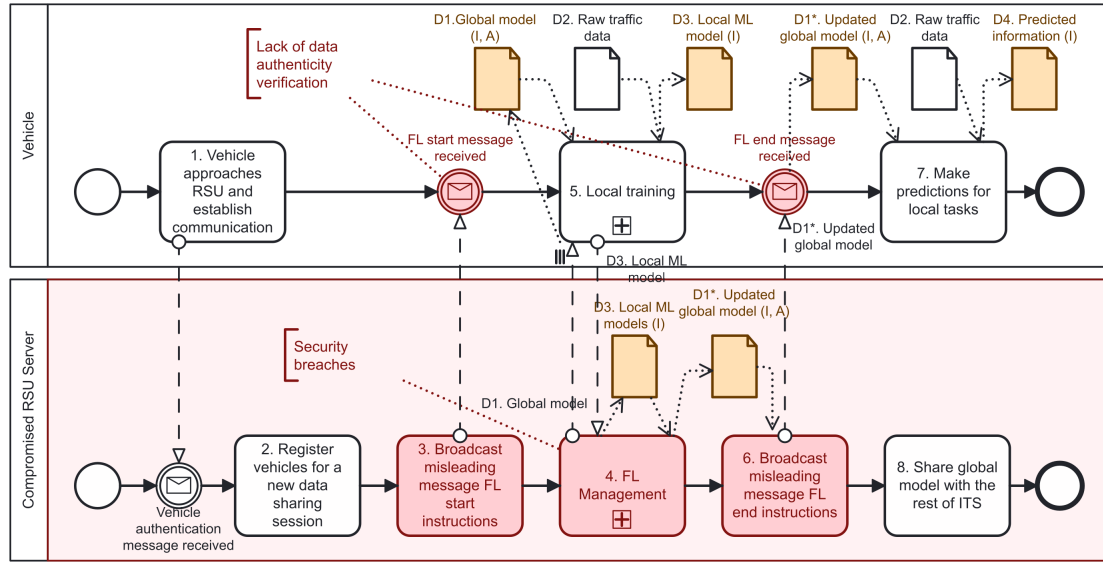


Figure 5. BPMN diagram with attack execution in FL by a compromised server

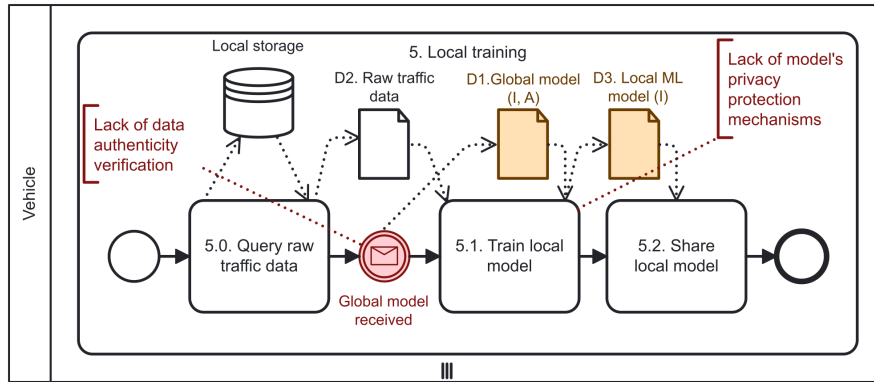


Figure 6. BPMN diagram of vulnerable local model training

exploited. These security weaknesses, along with others detailed in Table 3, create opportunities for various types of attacks that compromise the integrity and confidentiality of the FL process.

Security Breaches (V5). Security breaches serve as an umbrella term covering a wide array of vulnerabilities relevant to our use case, though their origins extend beyond the scope of this work. These breaches occur across various levels (physical, network, application, etc) and pose significant risks, potentially leading to unexpected server behavior or disrupting the entire FL process. This vulnerability affects the server infrastructure and opens the door to several types of attacks. These include unauthorized physical

access to server hardware, which in an IoV context could involve compromising RSUs (SA3) or central data centers (SA4). Attackers may also exploit weaknesses in network protocols (SA1) or configurations to gain unauthorized access to server systems or target vulnerabilities in server software (SA7, SA8), including the FL implementation itself, to control systems or access sensitive data. Additionally, attackers could manipulate authorized personnel to obtain access credentials (SA2) or confidential information about the server infrastructure.

Weak Centralized Access Control (V6). The centralized FL is exposed to a single point of failure. In a scenario of a compromised server, a malicious actor can take advantage of inadequately managed access rights (SA2, SA4) to query and submit global models from and to central storage. In addition, the malicious server can abuse the privileged access (SA2) to local models during FL aggregation, extracting sensitive information or insights that should remain private to individual vehicles or manipulating the management of the FL process.

Lack of Aggregation Monitoring and Validity (V7). The server should follow some expected protocol of aggregation. Without proper monitoring and validation of the aggre-

Table 3. Vulnerabilities in FL exploited by a compromised server

Id	Vulnerability	System Asset	Details
V5	Security breaches	SA1-SA4, SA7, SA8	Security breaches at various levels (physical, application, etc.) can cause unexpected server behavior or disrupt the entire FL process
V6	Weak centralized access control	SA2, SA4	A single point of failure in centralized FL allows a compromised server to query global and local models freely. It can also modify, delay responses, and distribute or upload faulty global models
V7	Lack of aggregation monitoring and validity	SA8	Lack of reliability, traceability, and control in aggregation can result in unrecognized malicious behaviors, such as global model tampering and poisoning
V8	Lack of data authenticity verification	SA1, SA2	FL participants should verify the source, validity, and authenticity of the data they receive from other parties
V9	Lack of server actions traceability	SA3, SA7, SA8	Lack of reliability, traceability, and control over server actions can lead to side activities outside the protocol, potentially compromising the system's security
V10	Lack of model's privacy protection mechanisms	SA6	Weak protection practices for local ML models may expose sensitive information encoded within them

gation process (SA8), malicious behaviors such as global model tampering performed by the server can go unrecognized. This vulnerability highlights the importance of ensuring the reliability and traceability of the aggregation step.

Lack of Data Authenticity Verification (V8). FL participants must be able to verify the source, validity, and authenticity of data received from other parties (SA1, SA2). Without strong verification mechanisms, malicious servers could manipulate data or alter FL commands to vehicle clients, misleading and disrupting the entire FL process. While encryption techniques are crucial in addressing this vulnerability, they are only one component of a broader, more robust security framework.

Lack of Server Actions Traceability (V9). Verifying the exact operations performed on the server, including model validation (SA7), aggregation (SA8), and overall system management (SA3), can be challenging. A compromised or malicious server could perform actions outside the prescribed FL protocol without detection, potentially introducing biases, errors, or backdoors into the global model during the aggregation process. These unauthorized actions could then propagate throughout the entire IoV system, affecting traffic management decisions, route optimizations, and overall system performance. Moreover, the lack of traceability makes it challenging to conduct thorough post-incident analyses, hindering the ability to identify the root causes of system anomalies or security breaches. This vulnerability underscores the critical need for implementing robust logging mechanisms, real-time monitoring systems, and regular auditing procedures to ensure the integrity and transparency of server operations in FL-based IoV systems.

Lack of Model's Privacy Protection Mechanisms (V10). Inadequate protection of local ML models (SA6) can expose sensitive information embedded within them. Even with properly configured access controls, a central server can still collect, compare, and analyze local model updates from different vehicles. This vulnerability allows curious servers to potentially infer private information about individual participants. Therefore, additional safeguards are necessary to protect local ML models from reverse engineering.

All the described vulnerabilities can be exploited by dishonest FL servers, leading to various attack scenarios. Table 4 presents three main attack methods that dishonest servers can perform in the FL process: unreliable communication attacks, model poisoning aggregation attacks, and model reverse-engineering attacks.

Unreliable Communication Attacks (SR4). A compromised server (ST1 Identity Spoofing, ET1 Server Unauthorized Access, ET2 Client Software Exploitation, ET3 Authentication Bypass) can exploit security breaches (V5), weak access controls (V6), lack of data authenticity verification (V8), and poor traceability of server actions (V9) to

conduct misleading FL communications. Without proper data authenticity verification (V8), the server could tamper with transmitted model data and coordination messages (TT2 V2I Message Tampering), modify the global model directly (TT3 Global Model Modification), and hide evidence of these actions through log manipulation (RT2 Sys-

Table 4. Security risks of FL in caused by a compromised server

Security Risk	Threat agent	Attack method	Threat	Vulnerability	Impact
SR4	Compromised aggregation server	Communication and model poisoning attacks	Compromised server can tamper with and distribute fake global models or FL process instructions. It can also delay responses, slowing down the FL process (ST1, TT2, TT3, RT2, RT3, DT2, DT3, DT4, ET1, ET2, ET3)	V5, V6, V8, V9	Integrity: D1, D3, D4 Availability: D1
SR5	Compromised aggregation server	Model poisoning attacks (unreliable aggregation)	Compromised server can manipulate the aggregation procedure or the global model's parameters, causing undesirable behavior. It then distributes this malicious model to other FL participants (ST1, TT2, TT3, TT4, RT2, RT3, ET1, ET2, ET3)	V5, V6, V7, V9	Integrity: D1, D3, D4
SR6	Compromised or curious server	Model inversion and membership inference attacks	A curious server can reverse-engineer model parameters to reconstruct sensitive data from FL participants (ST1, IT1, IT2, IT3, IT4, ET1, ET2, ET3)	V5, V6, V9, V10	Confidentiality: D1, D2, D3

tem Log Falsification). The compromised server could then deny distributing specific model versions (RT3 Model Distribution Repudiation), disrupting the FL process. Resource exhaustion attacks (DT2 Resource Exhaustion) and communication interference (DT3 Man-in-the-Middle, DT4 Infrastructure Node Attacks) can further degrade system performance.

The impact of such attacks primarily undermines the integrity of ITS assets. The global model (D1) would be compromised first, and after several FL rounds, local models (D3) would become distorted or poisoned, resulting in unreliable predictions (D4). Furthermore, the availability of the global model (D1) could be affected due to delayed responses to vehicle clients or wasted server computational resources.

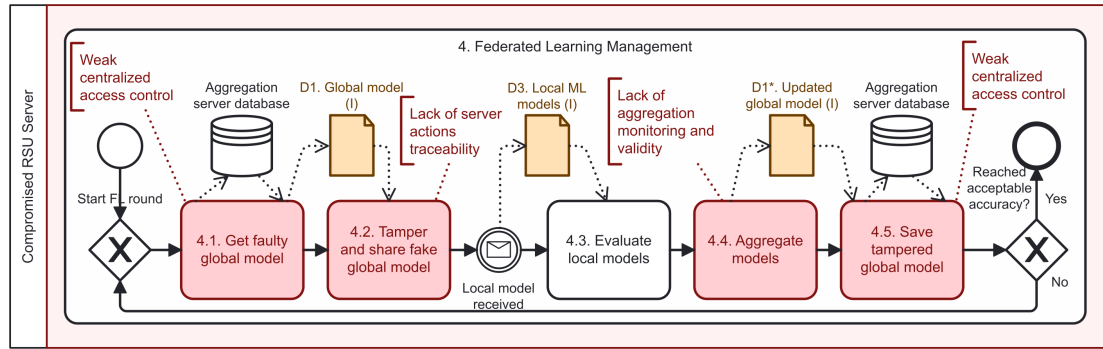


Figure 7. BPMN diagram of compromised server integrity attacks execution

Model Aggregation Poisoning Attacks (SR5). A compromised server (ST1 Identity Spoofing, ET1 Server Unauthorized Access, ET2 Client Software Exploitation, ET3 Authentication Bypass) can exploit security breaches (V5), weak centralized access controls (V6), and insufficient aggregation monitoring and validation (V7, V9) to manipulate the model aggregation process. The attacker could tamper with model updates during transmission (TT2 V2I Message Tampering), directly modify the global model in storage (TT3 Global Model Modification), and manipulate the aggregation results (TT4 Aggregation Result Manipulation) to introduce malicious behavior. For example, the aggregation server might intentionally skew the global model, causing it to misclassify specific inputs (such as road signs) or obscure critical traffic patterns while maintaining normal performance on other inference tasks. The attacker can hide evidence of these manipulations through log falsification (RT2 System Log Falsification) and later deny distributing the compromised model (RT3 Model Distribution Repudiation). This tainted model is then distributed to FL participants and stored in the central system, propagating the damage to ITS. Without strong monitoring and validation mechanisms, identifying and addressing the root cause of such attacks becomes extremely difficult.

The primary impact of such an attack targets the integrity of ITS assets. The global model (D1), local models (D3), and subsequent predictions (D4) can all be compromised, potentially causing widespread failures across the IoV system. The detailed scenario of communication and aggregation attacks is depicted in Figure 7.

Model Reverse-Engineering Attacks (SR6). In this scenario, a curious or compromised server (ST1 Identity Spoofing, ET1 Server Unauthorized Access, ET2 Client Software Exploitation, ET3 Authentication Bypass) attempts to reverse-engineer model parameters to reconstruct sensitive data from FL participants. This attack exploits multiple vulnerabilities, including security breaches (V5), weak access controls (V6), lack of server action traceability (V9), and insufficient privacy protection mechanisms for local models (V10). The attack exposes user traffic and identity information (IT1 User Traffic and Identity Disclosure) while capturing model updates through eavesdropping (IT2 Model Update Eavesdropping). The server's privileged position enables centralized collection of data and events (IT3 Centralized Collection of Data and Events), enhancing its ability to perform inference attacks (IT4 Inference Attacks). For instance, a compromised RSU server could collect local model updates from various clients across multiple FL rounds, using this accumulated data to execute different variations of model inversion attacks. By analyzing the subtle changes in model parameters over time, the attacker can potentially reconstruct the sensitive data used during training. The server's access to multiple versions of clients' models throughout the FL process enhances its capability to perform membership inference attacks, potentially identifying exact training samples used by each client.

The main impact of these attacks is on the confidentiality of ITS assets, with the global model (D1), raw data (D2), and local models (D3) all at risk of being compromised. This level of insight could reveal detailed information about individual vehicles' movements, preferences, or behaviors, severely compromising user privacy. The detailed scenario of reverse-engineering attacks is depicted in Figure 8.

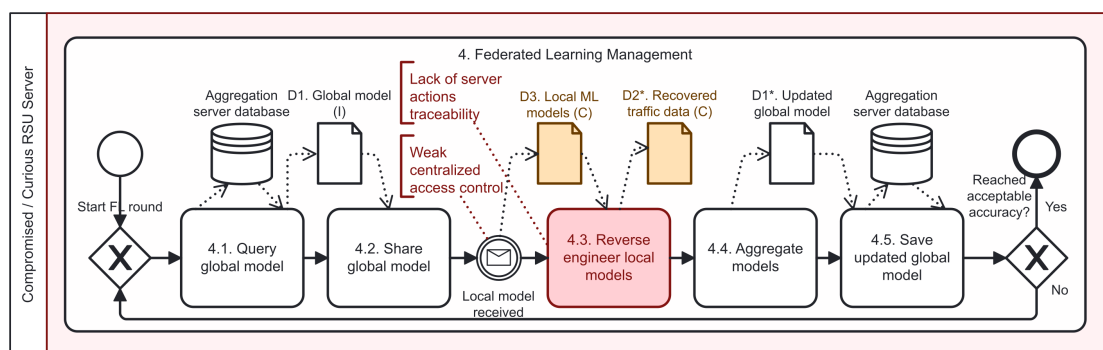


Figure 8. BPMN diagram of compromised server confidentiality attacks execution

Table 5. Impact of security risks on business assets

Business Asset	Confidentiality	Integrity	Availability	Number of Risks
D1. Global model	SR6, SR7, SR8	SR1, SR2, SR3, SR4, SR5, SR7	SR1, SR2, SR3, SR4, SR7	8 (14)
D2. Raw traffic data	SR6, SR8	~	~	2
D3. Local ML model	SR6, SR7, SR8	SR1, SR2, SR3, SR4, SR5, SR7	SR1, SR2, SR3, SR7	8 (13)
D4. Predicted traffic information	~	SR1, SR2, SR3, SR4, SR5, SR7	~	6

In this chapter, we have identified and categorized vulnerabilities and threats from three main sources: client-side, server-side, and man-in-the-middle actors. Table 5 summarizes the defined impact of the risks on protected assets. This analysis reveals several key insights:

1. **Pervasive Impact:** The global model (D1) and local ML models (D3) are the most widely affected assets, being impacted by 8 out of 8 identified risks. This underscores their central role in the FL process and highlights the need for robust protection mechanisms.
2. **Integrity Concerns:** The integrity of assets, particularly of the global model (D1), local models (D3), and predicted traffic information (D4), is at risk from the majority of the identified threats. This emphasizes the critical need for mechanisms to ensure data and model integrity throughout the FL process.
3. **Confidentiality Risks:** While fewer in number, the risks to confidentiality, especially for raw traffic data (D2), are significant. Despite only two risks being directly connected to raw traffic data, it remains the most sensitive information in our system and requires the highest level of protection. Availability Challenges: The availability of the global model (D1) and local models (D3) is threatened by multiple risks, which could potentially disrupt the entire FL process.
4. **Indirect Dependencies:** While raw traffic data (D2) and predictions (D4) have fewer direct risks, their security is indirectly dependent on the integrity and confidentiality of the models (D1 and D3). This highlights the interconnected nature of our system's security.

It's important to note that the list of security risks described in this system architecture is not exhaustive. Some risks may have been omitted due to their similarities, where different methods might lead to the same results. Additionally, we have not discussed threats connected to the perception level of the IoV system, as the main focus of this work is on the application of FL.

This risk analysis provides a foundation for developing our security strategy. It emphasizes the need for a multi-faceted approach that addresses integrity, confidentiality, and availability across all components of our FL system. Particular attention must be paid to protecting the global and local models, as they are the most widely impacted assets. Additionally, while raw traffic data has fewer direct risks, its sensitivity makes it a critical focus for our security measures.

In the next subsection, we will explore how blockchain technology can be leveraged to address these identified risks and enhance the overall security of our FL-based data sharing system in the IoV context.

5.4 Treatment-Related Concepts

Having identified the key vulnerabilities and security risks in our centralized FL scenario for IoV, we now focus on potential risk treatments and countermeasures. This section explores how introducing blockchain technology can serve as a comprehensive risk treatment strategy, primarily under the risk avoidance category.

To address the security risks identified in centralized FL for the IoV applications, we propose a decentralized solution leveraging Hyperledger Fabric (HLF) blockchain and ZKPs. This approach provides robust countermeasures to enhance security, privacy, and trust in the FL process. Below we detail the key HLF components [95] and how they mitigate specific risks, followed by a comprehensive workflow of the updated blockchain-based solution.

Distributed Ledger. The immutable, distributed ledger in HLF addresses several key vulnerabilities. For example, (i) mitigates lack of server actions traceability (V9) by maintaining a tamper-evident record of all transactions and events in the FL process. (ii) Enhances training process monitoring (V1) by providing an auditable trail of model updates and aggregations. (iii) Counters Global Model Modification (TT3) attempts, as any unauthorized changes would be immediately detectable.

Channels. HLF channels provide data isolation and access control. They mitigate weak centralized access control (V6) by allowing fine-grained control over who can access specific data and participate in particular FL tasks. Additionally, they reduce the risk of unauthorized data access and Information Disclosure threats (IT1-4) by segregating data and operations for different geographic areas or FL tasks.

Chaincode (Smart Contracts). Chaincode implements the business logic for FL operations. It addresses poor model validation (V3) and enhances robustness of model aggregation (V4) by encoding standardized validation and aggregation rules. Further-

more, it mitigates risks of Server Unauthorized Access (ET1) and Authentication Bypass (ET3) by automating critical processes and enforcing access controls.

Membership Service Provider (MSP). The MSP manages identities and access in the network. It counters Identity Spoofing (ST1) and Sybil Attacks (ST2) through robust identity management and authentication. The system also mitigates weak identity and access management (V12) by providing cryptographic identity verification.

Private Data Collections. PDCs enhance data privacy and access control. They address lack of model's privacy protection mechanisms (V10) by allowing sensitive data (individual model updates) to be shared only with authorized parties. PDCs also mitigate risks of Information Disclosure (IT1-4) by keeping sensitive data private while still allowing verification of its integrity.

Endorsement Policies. These policies define the rules for transaction validation. They enhance robustness of model aggregation (V4) by requiring multiple peers to validate transactions before they're committed. The policies also mitigate risks of Aggregation Result Manipulation (TT4) by ensuring multiple parties must agree on the validity of aggregations.

Ordering Service. The ordering service manages the creation and distribution of blocks. It addresses security breaches (V5) and weak centralized access control (V6) by decentralizing the critical function of block creation. The service also mitigates Denial of Service (DT1-4) risks by distributing the workload across multiple nodes.

Integration of Zero-Knowledge Proofs. While not a native HLF component, the integration of ZKPs helps to mitigate Aggregation Result Manipulation (SR7). They enhance aggregation monitoring and validity (V7) by allowing verification of computational correctness without revealing underlying data.

As shown in Table 6, the proposed blockchain-based architecture provides a comprehensive security framework through its eight distinct countermeasures. The complementary nature of these components creates multiple layers of protection, effectively transforming the centralized FL system into a robust decentralized solution suitable for IoV applications. This systematic approach ensures that each identified vulnerability and threat is adequately addressed while maintaining system efficiency and scalability.

In the context of security risk management, risk treatments are approaches taken to address identified risks. These can include risk avoidance, reduction, transfer, or retention. In our case, integrating blockchain into the FL process represents a significant

Table 6. Blockchain-based countermeasures

Id	Countermeasure	Vulnerabilities	Threats
C1	Distributed Ledger	V1, V9	TT3
C2	Channels	V6	IT1, IT2, IT3, IT4
C3	Chaincode (Smart Contracts)	V3, V4	ET1, ET3
C4	Membership Service Provider (MSP)	V12	ST1, ST2
C5	Private Data Collections	V10	IT1, IT2, IT3, IT4
C6	Endorsement Policies	V4	TT4
C7	Ordering Service	V5, V6	DT1, DT2, DT3, DT4
C8	Zero-Knowledge Proofs	V7	TT3

shift in system architecture and functionality, effectively avoiding many risks associated with centralized systems.

It’s important to note the relationships between risk treatment-related concepts. Each identified risk has at least one risk treatment, and these treatments are refined into specific security requirements. However, in risk acceptance or avoidance cases, no security requirements may be needed. Conversely, a single security requirement may address multiple risk treatments. Security requirements mitigate risks, although not all can be mitigated through security requirements alone. Finally, controls are implemented to fulfill one or more security requirements, forming the concrete measures that bring our risk treatment strategy to life [17].

In our work, we explored how these measures can be implemented as security requirements and controls, providing a robust framework for mitigating risks in our decentralized, FL system for IoV. As a result, we listed all security requirements and control components in Appendix III and IV.

5.5 Summary

In this chapter, we conducted a systematic security risk analysis of centralized FL-based data sharing in IoV. We began with developing a comprehensive threat model using the STRIDE methodology, which identified 20 distinct threats across six categories: Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, and Elevation of Privilege. This threat model provided a foundational understanding of potential attack vectors and vulnerabilities in centralized FL systems.

Building on this threat model, we applied the ISSRM framework to structure our analysis into three main components: asset-related, risk-related, and treatment-related concepts. The asset analysis identified critical business assets (global model, raw traffic data, local models, and predictions) and system assets (communication layers, computational resources, and algorithms), along with their security criteria of confidentiality, integrity, and availability.

Our risk analysis examined how the identified threats could exploit system vulnera-

bilities, leading to the identification of eight major security risks (SR1-SR8) originating from client-side, server-side, and man-in-the-middle actors. These risks were analyzed in detail, with particular attention to their potential impacts on system assets and security criteria.

For treatment-related concepts, we proposed a blockchain-based solution incorporating Hyperledger Fabric and ZKPs. This approach includes eight key countermeasures: distributed ledger, channels, chaincode, membership service provider, private data collections, endorsement policies, ordering service, and ZKPs. These components work together to address the identified vulnerabilities while maintaining system efficiency.

The analysis culminated in the development of 53 specific security requirements and 39 control components, categorized by implementation status. This comprehensive assessment provides the foundation for our blockchain-based solution design and implementation detailed in subsequent chapters, offering a robust framework for securing FL-based data sharing in IoV environments.

This systematic analysis not only highlighted the security challenges in centralized FL systems but also demonstrated how blockchain technology can effectively address these challenges, setting the stage for a more secure and trustworthy data sharing architecture in IoV applications.

6 Solution Design and Implementation

This section explores the research question **RQ3: How to design and implement secure data sharing using blockchain-based FL?** To address this question, we present the design and implementation of a blockchain-based FL solution as a core artifact, aligning with design science research guidelines (G4, G6, G7) outlined in Table 1. The solution is detailed through architectural diagrams that illustrate the design choices made to mitigate the security risks identified in Chapter 5. Following the design overview, the implementation is evaluated under selected threat scenarios.

6.1 Solution Design

In response to the security challenges inherent in centralized FL systems for the IoV, we propose an innovative solution combining Hyperledger Fabric (HLF) blockchain technology with ZKPs. Our solution specifically targets vulnerabilities related to system integrity and provenance in the centralized FL architecture. We focus on regulating server-side behavior and its interaction with clients, as system designers have more control over RSUs than individual vehicles. The implemented proof-of-concept encompasses several key components: well-defined access controls through private blockchain identities and policies, verifiable server-side aggregation using ZKPs, tamper-proof communication via encryption and digital signatures, and robust consensus-based mechanisms for model submission and distribution.

Our choice of HLF stems from its enterprise-focused architecture, which provides essential features for IoV environments, such as fine-grained access control, high transaction throughput, and modular consensus mechanisms. As a permissioned blockchain, HLF offers robust identity management and role-based access control that is particularly suited for RSU networks and vehicle interactions. The permissioned nature of HLF also enables more efficient consensus mechanisms compared to public blockchains, making it particularly suitable for the performance requirements of IoV systems. Its smart contract framework (chaincode) enables flexible implementation of FL logic while supporting complex data structures necessary for model management.

The integration of ZKPs with HLF creates a powerful framework for verifiable computations while preserving privacy. This combination allows us to verify the correctness of model aggregation without exposing individual contributions, providing mathematical guarantees for computation integrity while maintaining the confidentiality of participant data. However, the ZKP for client-side computations has limited practicality, as verifying the full training process is still a challenging task under research.

While our solution addresses core security concerns, certain aspects remain outside the current implementation scope. We do not implement defense methods against specific poisoning attacks from vehicles, as these would require control over vehicle data and actions, potentially compromising the privacy guarantees fundamental to FL. Similarly,

we omit specific defenses against model reverse-engineering attacks from the server side, as this represents a distinct research area with ongoing developments. These scope limitations allow us to maintain focus on establishing a secure, verifiable FL framework while preserving essential privacy properties in the IoV environment.

6.1.1 Solution Overview

At the core of our solution is a distributed network of RSUs functioning as full nodes within the Hyperledger Fabric blockchain. These RSUs form the backbone of our system, participating in consensus mechanisms and maintaining a shared, immutable ledger. This decentralized structure eliminates single points of failure and enhances the overall resilience of the FL process.

Vehicles, the primary data contributors in our IoV ecosystem, interact with the blockchain network through nearby RSUs, which act as gateway peers. This architecture accommodates the transient nature of vehicles and their resource constraints while ensuring robust participation in the FL process.

To manage the complexities of FL in a blockchain environment, we implement a suite of smart contracts or chaincodes in Hyperledger Fabric terminology. These chaincodes orchestrate crucial aspects of the FL lifecycle, including global model storage and retrieval, round management, access control, and logging of participation and model access events. A key feature of our chaincode implementation is the integration of ZKP verification, which allows for the validation of aggregation correctness without exposing sensitive data.

Our solution employs a multi-channel architecture within Hyperledger Fabric to enhance data isolation and access control. Separate channels are created for different geographic areas or specific FL tasks, allowing for granular control over data sharing and participant access. This structure supports the diverse and dynamic nature of IoV environments while maintaining strict security boundaries. The high-level diagram of this setup is presented in Figure 9.

Our system's FL process follows a well-defined workflow. It begins after the initialization phase, where vehicles and RSUs acquire their credentials within the blockchain private network. Vehicles register for FL rounds using their CA-issued credentials that can be recognized by RSUs. Model distribution to vehicles is controlled by local MSP access checks, ensuring only authorized participants receive the current global model.

Local training occurs off-chain on individual vehicles, preserving data privacy. Upon completion, vehicles submit their model updates to RSUs, which then use chaincode to log these submissions and store model updates in private data collections with aligned hash on the channel ledger. The aggregation of model updates is performed off-chain by designated RSUs, accompanied by the generation of ZKPs to verify the correctness of the aggregation process without revealing individual contributions.

The submission of new global models to the ledger is a critical process in our system.

RSUs submit updated global models along with their corresponding ZKPs through chaincode. Other RSUs in the network then validate these transactions, including verification of the ZKPs. Only when consensus is reached among the RSUs is the new global model accepted and stored on the ledger, ensuring the integrity and trustworthiness of the model updates.

Comprehensive logging and provenance tracking are integral to our solution. All significant events in the FL process, including model access, submissions, and aggregations, are logged on the ledger through chaincode functions. These logs serve as an immutable audit trail, crucial for maintaining transparency and accountability in the system.

Our solution leverages TLS encryption and digital signature capabilities for securing data at rest and in transit within the blockchain network. So, it addresses the model updates eavesdropping with the following information disclosure in the system.

To enhance data obfuscation, values within chaincode can be encrypted—partially or fully—using standard cryptographic algorithms like AES before transactions are sent to the ordering service and blocks are appended to the ledger. After encrypted data is recorded on the ledger, only users with the corresponding decryption key can access the

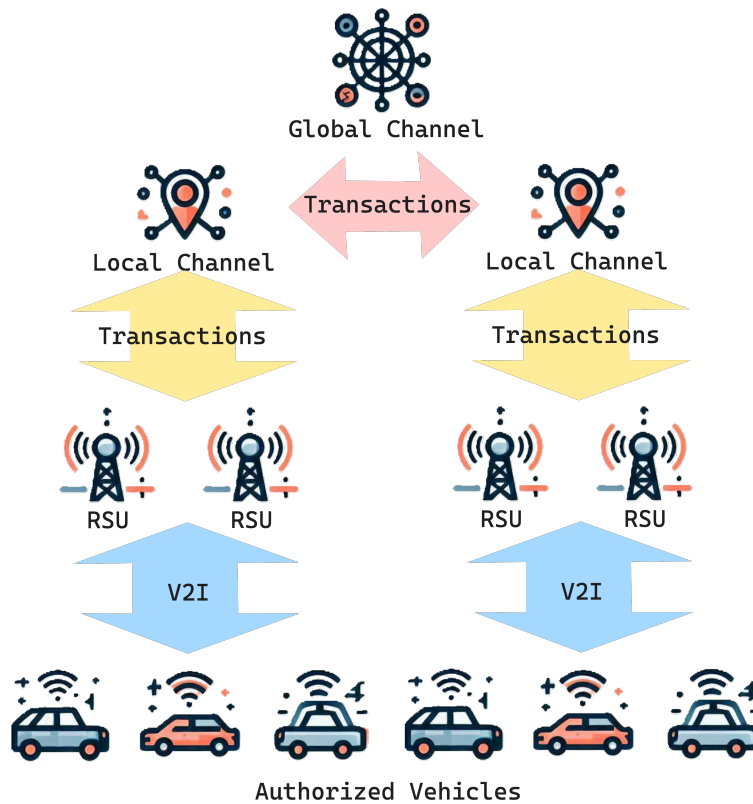


Figure 9. High-level Blockchain-based FL architecture for IoV

original information [95].

To address performance considerations, particularly regarding the storage of large model files, our system adopts a hybrid approach. Large files are stored off-chain or with limited access, with only model hashes and essential metadata retained on the blockchain ledger. This strategy optimizes storage efficiency while maintaining the integrity and traceability of the FL process.

In conclusion, our proposed solution combining Hyperledger Fabric and ZKPs offers a robust, secure, and privacy-preserving framework for FL in the IoV environments. By leveraging blockchain technology's strengths in decentralization, immutability, and consensus, along with the privacy guarantees of ZKPs, we address the critical security and trust challenges inherent in centralized FL systems. This approach not only enhances the security and privacy of the FL process but also paves the way for more efficient and trustworthy collaborative learning in dynamic IoV ecosystems.

6.1.2 Identity Management

Access control in our system is implemented through fine-grained policies encoded in chaincodes and MSPs, determining which vehicles can participate in specific FL rounds. We utilize Hyperledger Fabric's private data collections to handle particularly sensitive information, providing an additional layer of data protection within the blockchain network. Private data collections allow a subset of organizations on a channel to endorse, commit, or query private data without making it available to all channel members. Figure 10 depicts the detailed diagram with access control details.

A blockchain network includes various actors such as peers, orderers, client applications, administrators, and others. Each of these actors, entities within or outside the network that can interact with its services, holds a unique digital identity, encapsulated in an X.509 digital certificate issued by a CA. These identities are essential because they define each actor's specific permissions and access rights to resources and information within the blockchain network [95].

The minimal CA structure for a Hyperledger Fabric-based IoV implementation centers around a single global CA that serves as the root of trust for the entire network. This CA operates as the primary authority for issuing and managing digital certificates across all network participants and organizations. For RSU nodes, the CA issues specialized certificates that encode their specific roles within the FL process - whether they act as aggregators tasked with combining local models into global ones, validators responsible for verifying aggregation ZKPs, or endorsers who participate in the transaction validation process. On the vehicle side, the CA issues client certificates to authorized FL participants, enabling them to interact with the network and contribute their local model updates. Each certificate includes appropriate attributes and extensions that define the entity's role, permissions, and organizational affiliation, allowing the network's MSPs to enforce access control and operational policies. While this simplified CA structure may

not provide the same level of security isolation as a hierarchical multi-CA setup, it offers a practical starting point for proof-of-concept implementations.

The Hyperledger Fabric MSP architecture for the IoV implements a hierarchical structure that clearly defines organizational boundaries, roles, and access policies across different levels of the network. At the foundation are the organization-level MSPs, where each group of RSUs operates under its own MSP containing peer identities, administrative credentials, and organizational certificates. Similarly, a dedicated vehicle organization MSP manages the identities of all authorized vehicle clients, ensuring consistent identity management across the network.

Building upon these organizational MSPs, local channel MSPs establish the operational framework for specific regions. These channel-level MSPs incorporate the relevant

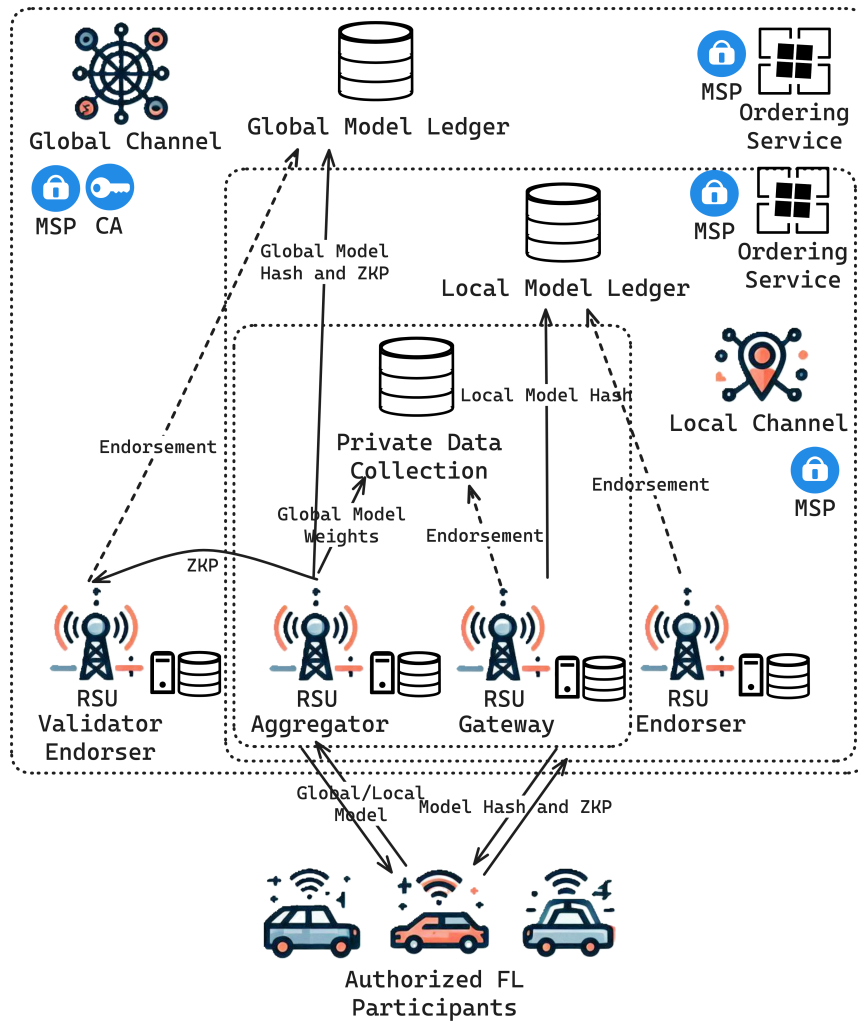


Figure 10. Detailed Blockchain-based FL architecture for IoV

RSU and vehicle organization MSPs while defining specific policies for peer roles, endorsement requirements, and cross-organizational interactions. Each local channel also includes its own ordering service MSP, enabling consensus operations within that regional scope. This regional structure allows for efficient management of local FL operations while maintaining security boundaries.

At the highest level, the global channel MSP provides network-wide coordination by incorporating all RSU organization MSPs, the vehicle organization MSP, and its own ordering service MSP. This global structure enables cross-regional operations and network-wide FL coordination while respecting organizational boundaries and roles. Throughout this hierarchy, each peer maintains local copies of all relevant MSPs, enabling efficient, decentralized identity verification and policy enforcement without requiring external validation services. This comprehensive MSP architecture creates a robust foundation for secure, scalable, and efficient operation of the FL system across the vehicular network while maintaining clear organizational boundaries and role-based access control.

6.1.3 Network Initialization and Participants Registration

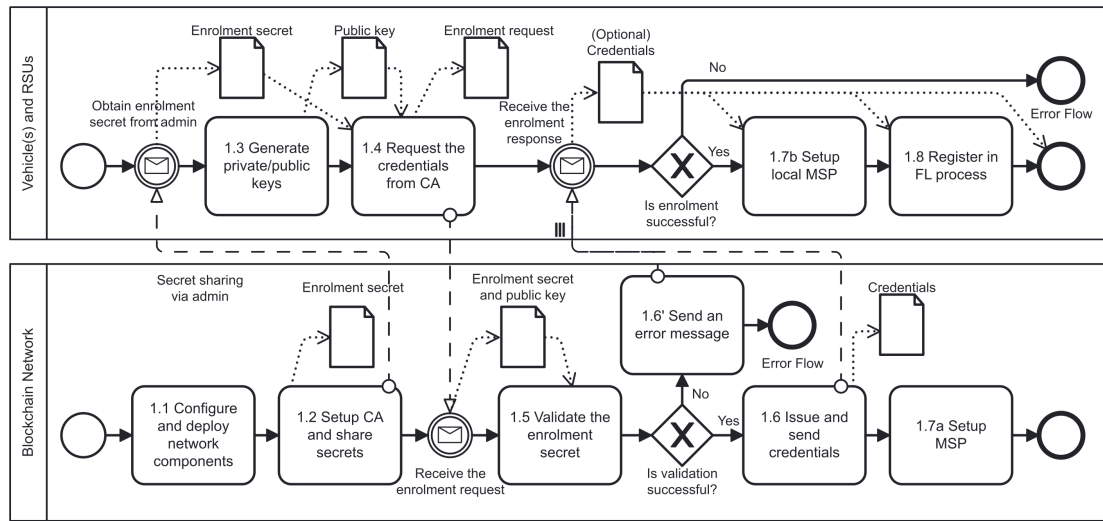


Figure 11. BPMN diagram of network initialization and participants registration

The BPMN diagram in Figure 11 illustrates the network initialization process and participant registration in our blockchain-based FL system for IoV. The process begins with the blockchain network configuration, where an administrator configures and deploys the network components, including setting up the CA (steps 1.1-1.2). This foundational step establishes a distributed trust architecture that directly addresses the vulnerability of centralized access control identified in our threat analysis. The CA setup provides

the cryptographic backbone needed to mitigate identity spoofing threats through robust identity verification.

For Vehicles and RSUs, the registration process starts with obtaining an enrollment secret from the network admin through a secure, out-of-band channel. The Vehicle/RSU then generates a pair of private/public keys, which are fundamental to secure communication within the network (step 1.3). Using the enrollment secret and generated keys, the Vehicle/RSU requests credentials from the CA, with the request encrypted to ensure confidentiality during transmission (step 1.4). This multi-layered security approach specifically addresses weak identity management vulnerability and helps prevent authentication bypass threats and man-in-the-middle attacks.

On the blockchain network side, after the initial setup, the network processes participant enrollment requests through a series of security validations (step 1.5). It validates the enrollment secret provided in these requests to ensure only pre-approved entities can join the network. If validation is successful, the network will issue and send credentials to the requesting entity (step 1.6). This strict validation process helps prevent Sybil attacks through rigorous identity verification and mitigates unauthorized access threats. The encrypted communication channels used during this process address information disclosure risks.

The network then sets up the channel MSP, which governs the overall access control and identity management for the blockchain channel (step 1.7a). Upon receiving the enrollment response, the Vehicle/RSU verifies if the enrollment was successful. If so, it proceeds to set up local MSP (step 1.7b). The MSP manages identities and access control within the network, establishing a fine-grained access control system that addresses privilege escalation risks. Finally, the CA credentials are used by RSUs to authenticate and authorize vehicles in the FL sessions (step 1.8), ensuring that only verified participants can contribute to the FL process.

This comprehensive registration process creates a robust security foundation that directly addresses several key vulnerabilities identified in our threat model. It eliminates weak centralized access control through distributed trust, addresses lack of authentication via multi-layer identity verification, mitigates security breaches through cryptographic protocols, and provides audit trail capabilities, reducing repudiation risks. The use of enrollment secrets, public-key cryptography, and CA ensures that only authorized entities can participate in the network, creating a trusted environment for FL in the context of IoV while balancing the need for security, privacy, and efficient collaboration.

This secure enrollment phase is essential for establishing trusted participation in subsequent FL operations. By implementing these security measures during the initialization and registration process, we create a strong foundation that helps prevent many of the attacks and vulnerabilities identified in our earlier threat analysis, ensuring the integrity and security of the entire FL system.

6.1.4 Global Model Request and Distribution

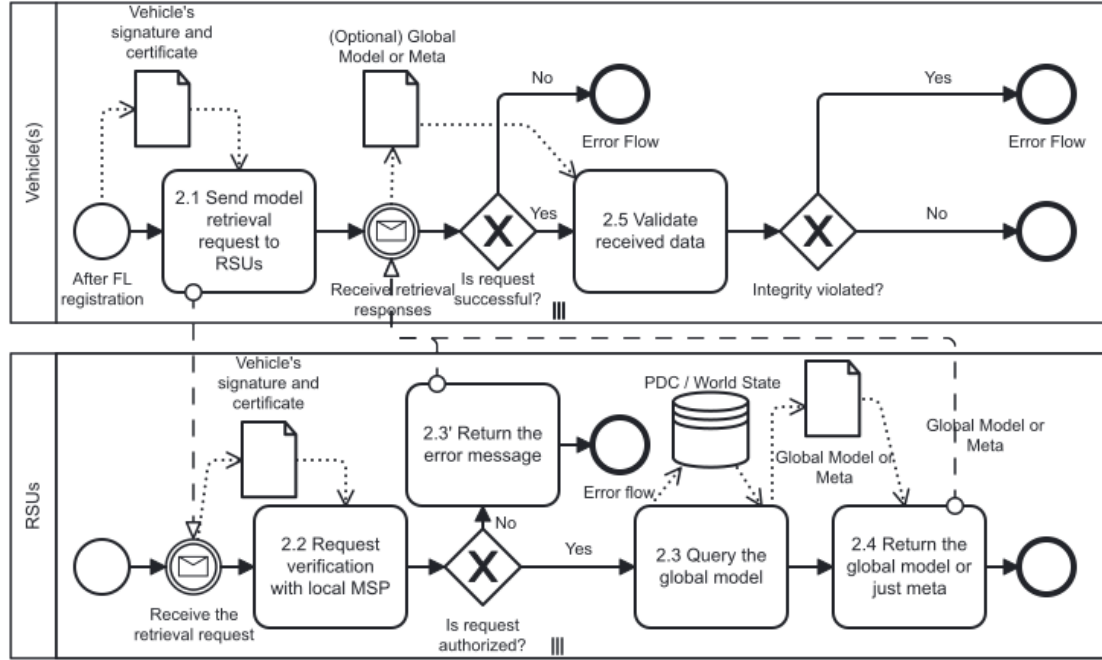


Figure 12. BPMN diagram of global model request

The BPMN diagram in Figure 12 illustrates the process of requesting and distributing the global model in our blockchain-based FL system for IoV. This process implements several critical security measures that directly address the vulnerabilities and threats identified in our earlier analysis, particularly those related to model distribution integrity and unauthorized access.

The vehicle initiates the process after successful FL registration by sending a model retrieval request (step 2.1) that includes its digital signature and certificate. This comprehensive request structure directly addresses identity spoofing threats and authentication bypass attempts. By requiring cryptographic signatures and certificates, the system mitigates the vulnerability of weak identity management identified in our threat analysis.

When RSUs receive the retrieval request, they implement a robust verification process (step 2.2) using their local MSP. This verification step is crucial for preventing malicious actors from obtaining unauthorized access to the global model. The use of local MSP verification also helps mitigate the risks associated with weak centralized access control by distributed access management.

A key security decision in our design is the parallel query and validation approach. Multiple RSUs independently process the request, with some providing the complete global model while others supply hashes and metadata for verification (steps 2.3-2.4).

The global model weights is stored in PDC with attached hash identifier on-chain. The vehicles can detect tampering attempts and ensure model integrity by checking the integrity of received data against metadata from multiple RSUs (step 2.5). This validation process helps prevent the distribution of compromised models while maintaining an auditable trail of model access and distribution, addressing repudiation concerns.

The entire process operates over secure HTTPS connections, with error-handling mechanisms at multiple stages. These measures collectively address the confidentiality risks associated with model update eavesdropping and help prevent unauthorized data exposure.

By implementing these layered security measures and leveraging blockchain's distributed nature, our solution creates a secure and verifiable model distribution process that significantly reduces the attack surface compared to traditional centralized approaches. Combining cryptographic signatures, distributed verification, and cross-validation makes it substantially more difficult for attackers to compromise the global model distribution process while maintaining system efficiency and scalability.

6.1.5 Local Model Training and Submission

The BPMN diagram in Figure 13 illustrates the local model training and submission process in our blockchain-based FL system for IoV. This process incorporates multiple security layers that directly address several critical vulnerabilities and threats identified in our earlier analysis, particularly those related to model integrity, privacy preservation, and unauthorized submissions.

The process begins on the vehicle side with carefully designed steps that enhance security while preserving privacy. After receiving the global model, the vehicle queries its raw traffic data from local storage (step 3.1), keeping sensitive data strictly on-device. The vehicle then trains its local model using this private data and the previously received global model (step 3.2), enabling collaborative learning while keeping raw data unexposed. Before submission, the vehicle computes a hash of the trained model. This cryptographic hash serves multiple security purposes: it provides tamper evidence for model integrity verification, creates a unique identifier for model tracking and auditing, and enables validation without exposing the complete model, effectively addressing model tampering threats and ensuring non-repudiation. Then, the vehicle submits the model to the ledger via a gateway RSU (step 3.3).

Upon receiving a model submission, the RSU validates the submitted data (step 3.4), checking both the vehicle's credentials and the model's integrity. This multi-faceted verification addresses multiple security concerns by mitigating identity spoofing threats through credential verification, preventing unauthorized submissions via MSP validation, and detecting tampering attempts through hash verification. A key security decision is the use consensus-based submission via chaincode invocation (step 3.5), so multiple peers will have to verify the transaction before accepting it to the ledger.

The blockchain network layer provides security through its consensus mechanism. The endorsing peers first validate the transaction proposal (step 3.6), followed by transaction simulation (step 3.7) and ordering service processes of block creation (step 3.8). This distributed validation prevents malicious model submissions, ensures agreement on model acceptance across the network, and creates an immutable record of model submissions.

The process concludes with updating the world state (step 3.9), ensuring all authorized peers maintain a consistent view of the system with only model hashes and meta as public records. This design maintains system-wide consistency without exposing sensitive data and enables auditing and verification through public hashes. Actual local models remain available only to the direct aggregation server to minimize the data centralization. It successfully addresses both integrity and confidentiality requirements identified in our

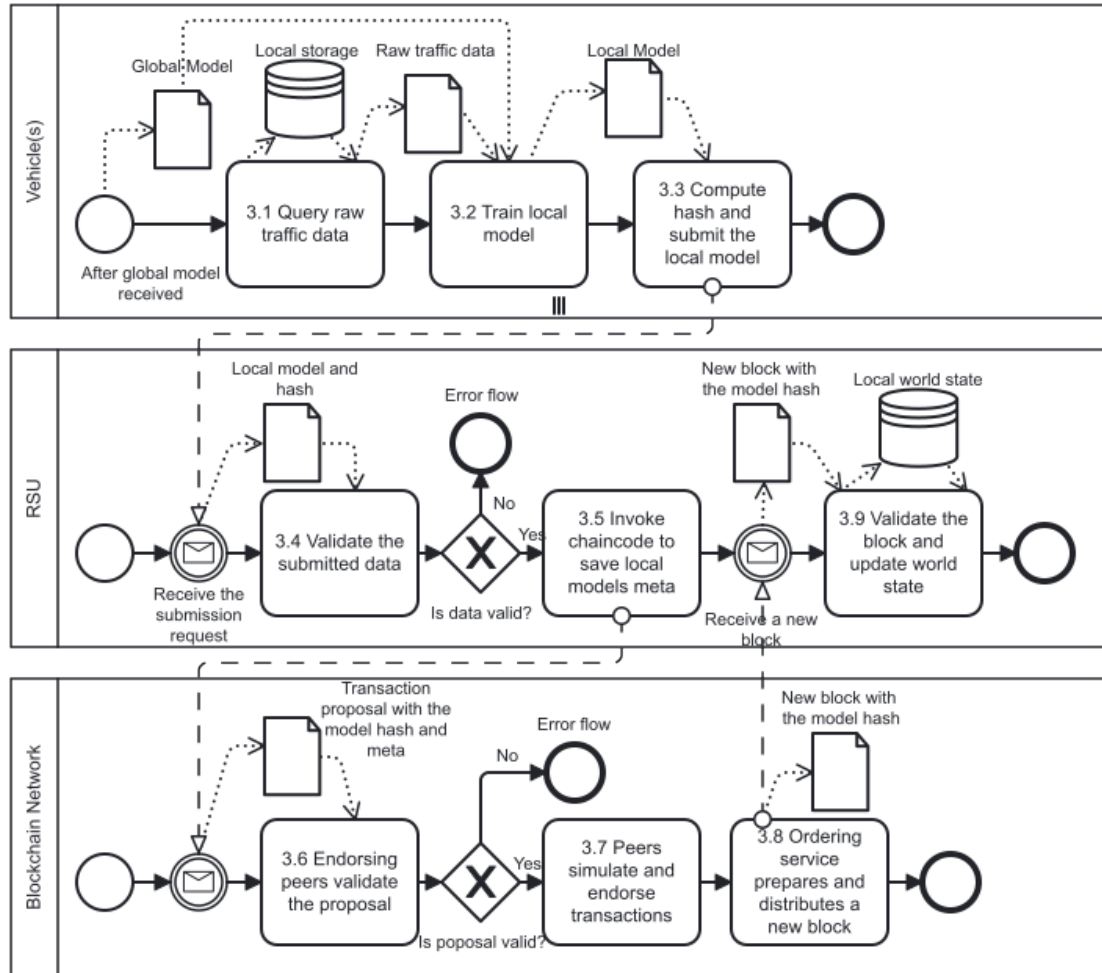


Figure 13. BPMN diagram of local model training and submission

security analysis.

By implementing these layered security measures and leveraging blockchain's distributed nature, our solution creates a secure and privacy-preserving model submission process that significantly reduces the attack surface compared to traditional centralized approaches. The combination of local training, cryptographic hashing, and distributed validation makes it substantially more difficult for attackers to compromise the FL training and submission process while maintaining system efficiency and scalability.

6.1.6 Models Aggregation, ZKP Validation, and Submission

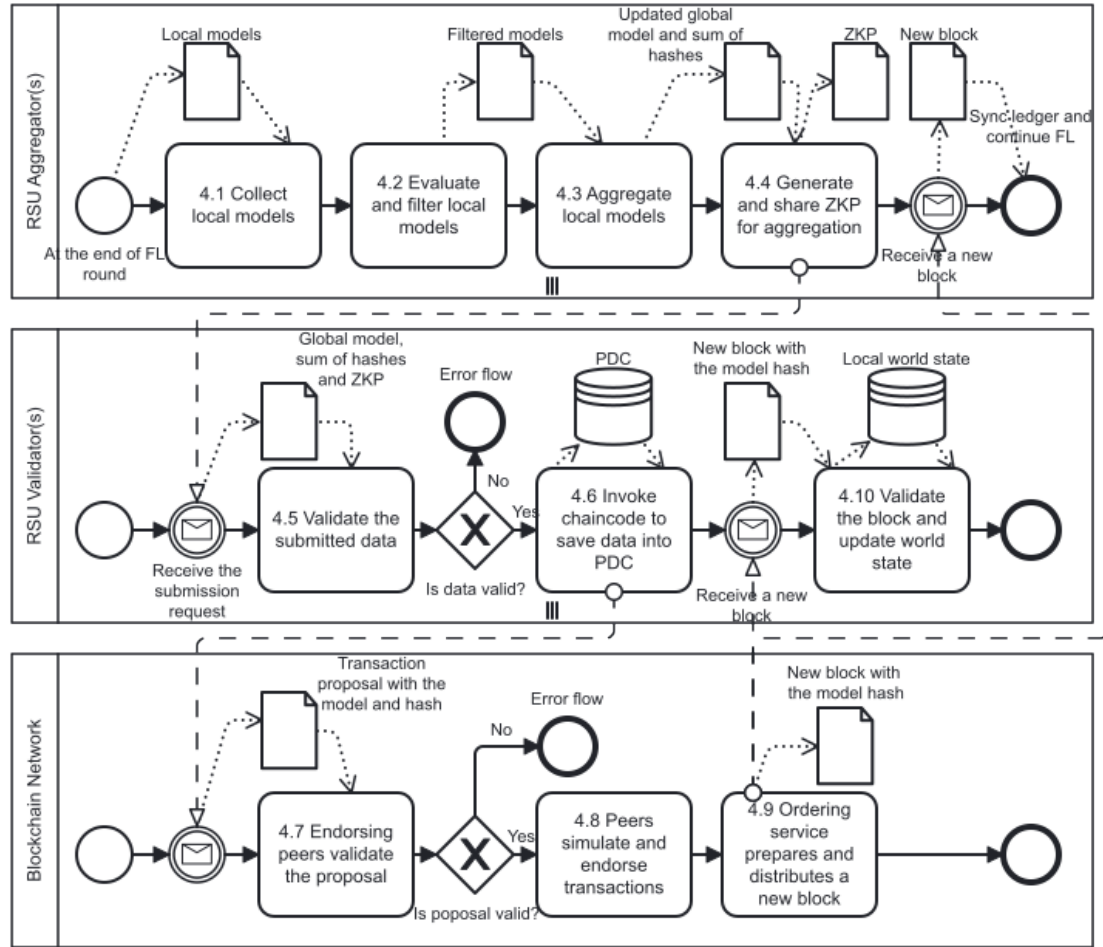


Figure 14. BPMN diagram of models aggregation, ZKP validation, and submission

The BPMN diagram in Figure 14 illustrates the model aggregation and ZKP verification process in our blockchain-based FL system. This process represents a crucial

security innovation that addresses several key vulnerabilities identified in our threat analysis, particularly those related to aggregation integrity and verification.

At the end of each FL round, the designated RSU aggregator collects local models shared directly with him by the FL clients (step 4.1). The RSU then performs an evaluation and optional filtering of the local models (step 4.2), implementing mechanisms to detect and exclude potentially malicious or low-quality contributions. This filtering process potentially addresses model poisoning threats and helps maintain the integrity of the final aggregated model.

The aggregation process itself incorporates several security innovations. The RSU aggregates the filtered local models and calculates the sum of model hashes used in aggregation (step 4.3). A key security enhancement is the generation of ZKPs for the aggregation process. The aggregator computes ZKP for both the aggregated model and the sum of hashes (step 4.4). These proofs mathematically demonstrate that the aggregation was performed correctly without revealing any information about individual models, addressing both integrity and privacy concerns identified in our threat analysis.

The validation layer, implemented by RSU validators, provides additional security through multi-stage validation (step 4.5). Validators receive the aggregation submission, which includes the new global model, the sum of hashes, meta, and ZKP. They also query hashes of local models that must be used in the aggregation. The validation process includes verifying the ZKP to ensure correct aggregation and sum calculation, comparing the claimed sum of hashes against independently calculated values, and validating the format and structure of the new global model. This thorough verification process helps prevent manipulation of the aggregation process and ensures the integrity of the global model. Ideally, this process should be implemented via an extension of the endorsement process where ZKP validation will be performed by all endorsement peers. ZKP can also be validated by vehicles right before the local training. However, it will be less resource efficient, as the ledger may store redundant faulty data, and vehicles will have to spend their computing resources on ZKP validation.

Then, the global model hash and validation results can be submitted by invoking chaincode (step 4.6) to save data into PDC. Our system protects sensitive global model weights from unauthorized access while enabling selective data sharing among authorized peers. The blockchain network layer further enhances security through its consensus mechanism. The endorsing peers validate the transaction proposal (step 4.7), including the ZKP verification results. After successful validation, peers simulate the transaction and create endorsements (step 4.8). The ordering service then collates these endorsements into a new block (4.9), creating an immutable record of the aggregation process and its verification. This distributed consensus approach effectively prevents manipulation of the aggregation results and provides a transparent, auditable trail of the entire process.

The process concludes with updating the world state across all peers (step 4.10), ensuring a consistent view of the system while maintaining privacy by separating public

and private data. This final stage creates a verifiable record of the aggregation process while protecting sensitive information about individual contributions.

By implementing these sophisticated security measures, particularly the combination of ZKPs and blockchain-based verification, our solution guarantees the integrity of the aggregation process while preserving privacy. This approach effectively addresses the vulnerabilities of centralized aggregation systems while maintaining efficiency and scalability. The multi-layered verification process, combined with cryptographic proofs and distributed consensus, makes it extremely difficult for malicious actors to manipulate the aggregation process or compromise the resulting global model.

6.2 Implementation

6.2.1 Flower - Federated Learning Framework

We use the Flower framework [96] as the foundation for our FL implementation. This Python framework implements a hub-and-spoke topology for FL, where a central server coordinates multiple distributed clients. The framework employs a thoughtful architectural separation between long-running communication and task-specific processes. On the server side, SuperLink manages client connections, while the embedded ServerApp contains the FL logic. Similarly, clients use SuperNode for network communication and embedded ClientApp for local training tasks. This separation enables multiple FL projects to share the same infrastructure while maintaining proper isolation between different training tasks.

Flower provides a flexible configuration system that enables fine-grained control over client behavior during the FL process. The framework supports dynamic per-round configuration, allowing server-side adjustment of parameters based on training progress. This capability is particularly valuable for implementing adaptive learning strategies or responding to changing network conditions.

At the core of Flower’s server implementation is the Strategy class, which defines the FL protocol through key steps: model setup, client selection and configuration, aggregation, and model validation. This modular design allows custom algorithms to be implemented by extending the base Strategy class and overriding specific methods as needed.

Flower implements robust security measures through a combination of authentication mechanisms and channel security. All communication channels are protected using TLS encryption, ensuring data confidentiality during transmission. Client authentication employs ECDH key exchange and HMAC verification, providing strong identity validation. The server maintains and validates a list of known client public keys, establishing a trusted network of participants.

The framework supports comprehensive logging and verification through its built-in modification system (mods). These mods act as middleware, enabling transaction

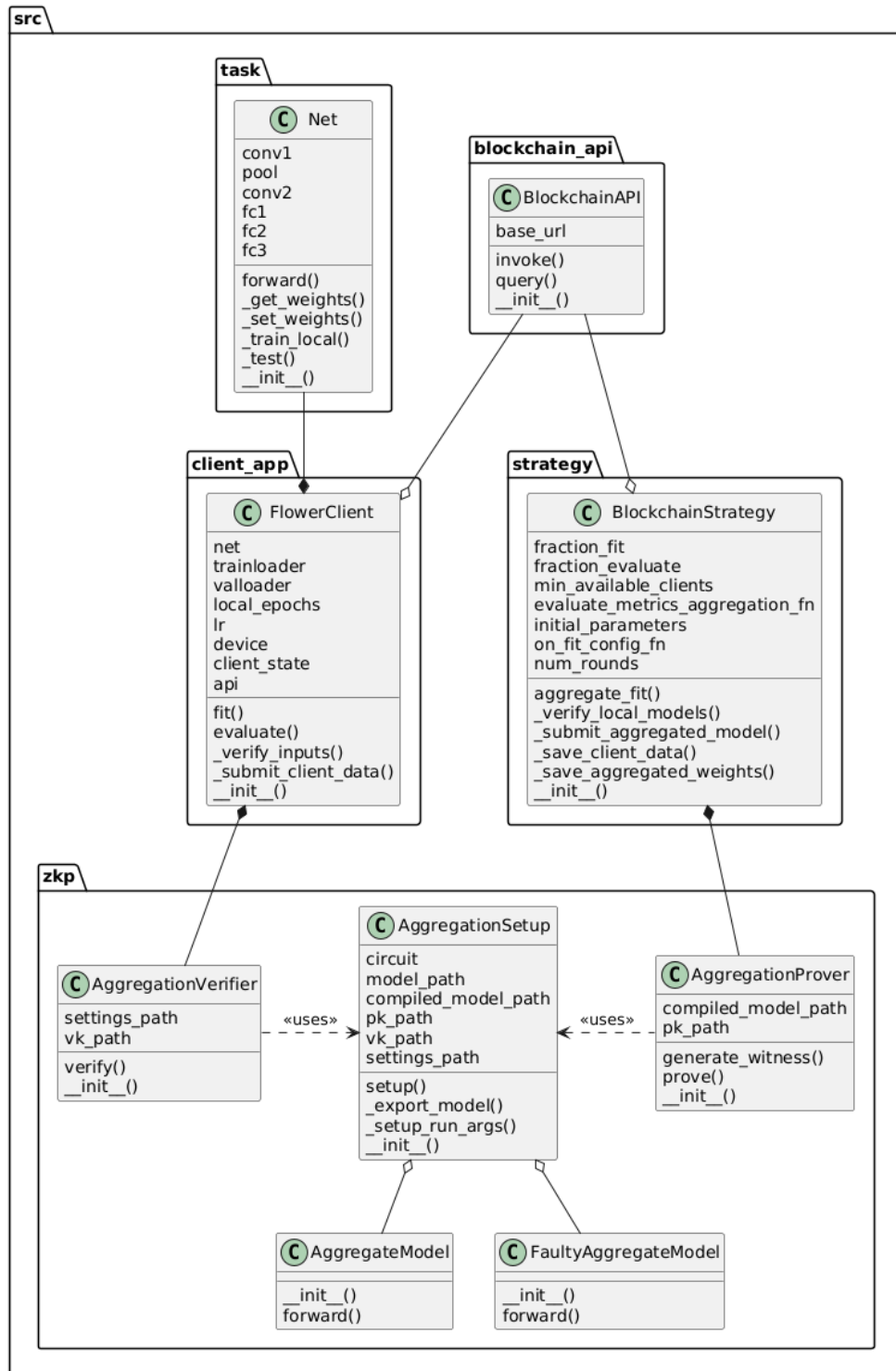


Figure 15. Class diagram for FL and ZKP implementation

logging and verification at various points in the communication flow. The mod system provides integration points for blockchain submission and verification, allowing each transaction to be recorded and validated. Mods can be chained together, enabling multiple verification steps and creating a robust audit trail of the FL process.

To ensure secure communication, we enable TLS encryption and client authentication across all channels. Client-specific parameters are managed through Strategy callbacks, allowing for flexible configuration while maintaining security. This comprehensive approach creates a robust foundation for secure, verifiable FL while taking full advantage of Flower’s built-in capabilities for scalability and reliability.

Our blockchain-enabled FL system leverages Flower’s architecture to create a secure and verifiable learning environment. To establish a stable network infrastructure, we utilize the SuperLink for aggregation servers on RSUs and SuperNode components for local training on the vehicles side. In Figure 15, we can see a detailed look at the class structure of the implementation. The custom BlockchainStrategy incorporates model aggregation and ZKP generation. Meanwhile, FlowerClient verifies the ZKP of the received global model before local training. The system interacts with the blockchain via REST API (BlockchainAPI) to record and verify all transactions, creating an immutable audit trail of the learning process.

As shown in Figure 16, the sequence diagram illustrates the implementation of our FL protocol using Flower’s architecture. The process begins when a Flower Client joins an FL run, triggering the creation of a global model through chaincode invocation. During each FL round, the Aggregator distributes the global model along with its ZKP to clients. Clients verify both the model and proof before proceeding with local training. After training, clients submit their local models through chaincode, which are then verified and aggregated by the Aggregator. The Aggregator generates new ZKPs for the aggregated global model before storing it on the blockchain and completing the round. At the end of the run, the Aggregator performs the evaluation and stops the whole FL process.

6.2.2 EZKL - Zero-Knowledge Proof Compiler

To provide verifiable model aggregation in our FL system, we integrate EZKL [97] - a system for generating ZKP for AI computations. EZKL enables us to prove that the weighted averaging of model updates was performed correctly without revealing the individual model weights or client contributions.

The weighted federated averaging is implemented as a PyTorch model that takes client model weights and the number of training samples as inputs and produces the aggregated model parameters (weighted average) as output. This model is then converted to the ONNX format, which EZKL uses to generate ZKP. The process involves three main phases: setup, proof, and validation. AggregationSetup, AggregationVerifier, and AggregationProver represent these phases in the zkp package in Figure 15.

The implementation involves three distinct phases. First, the setup phase is performed

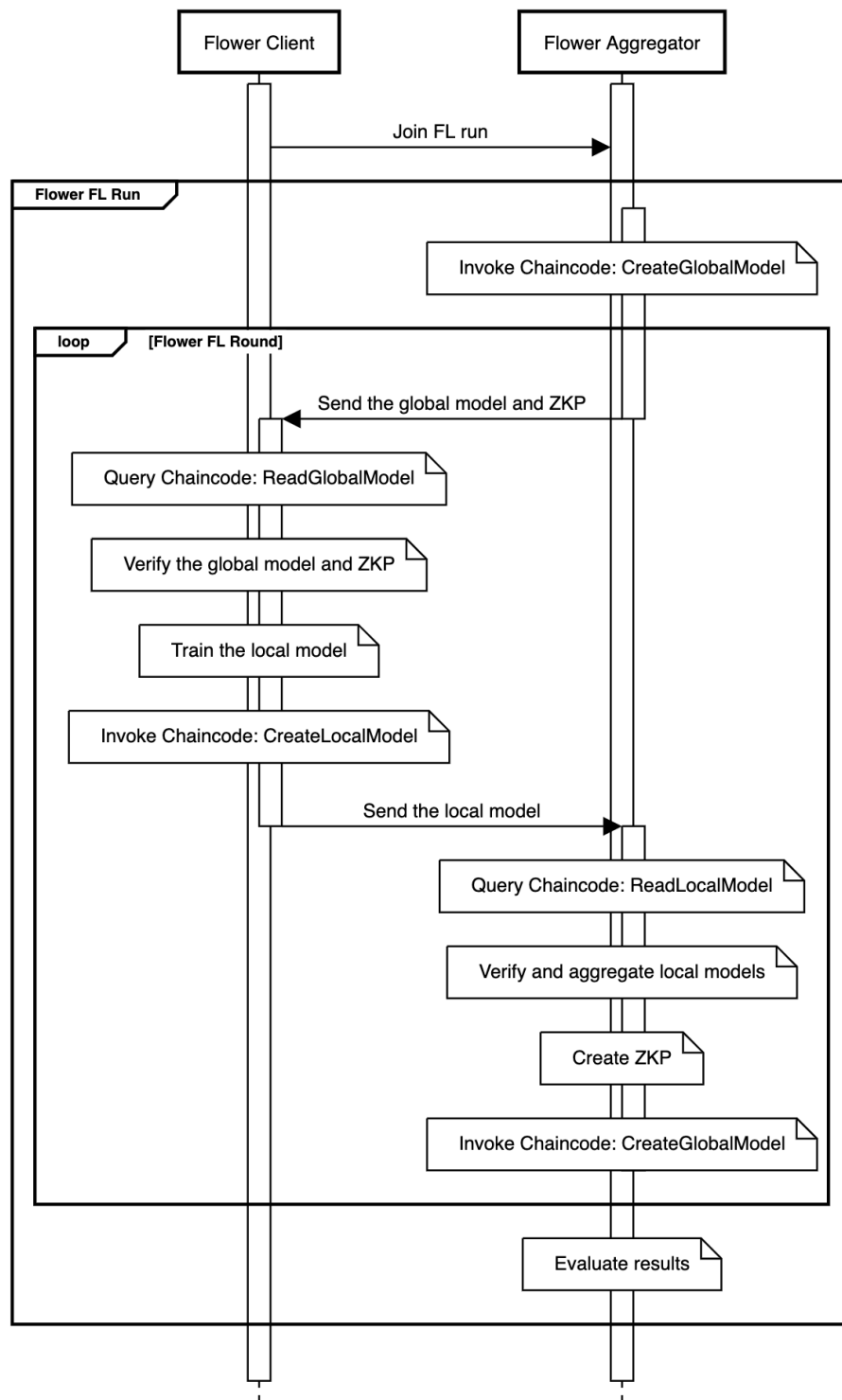


Figure 16. Sequence diagram for FL data sharing

once for a particular machine learning task and model architecture. During setup, we define the circuit by exporting the PyTorch aggregation model (AggregateModel or FaultyAggregateModel) to ONNX format and use EZKL to generate the cryptographic proving and verification keys. These keys are then distributed to the clients and servers - the proving key enables servers to generate proofs of correct aggregation, while the verification key allows clients to validate these proofs.

During active FL process, the proving phase occurs whenever model aggregation is performed. The server generates a proof demonstrating that it correctly executed the weighted averaging of client updates according to their dataset sizes. This proof can verify that proper weights were used without revealing the actual model parameters or individual client contributions.

Finally, in the verification phase, any participant with the verification key can check the correctness of the aggregation process. The verification is computationally efficient and requires only the verification key and public parameters. This allows clients to ensure their updates were properly incorporated into the global model without needing access to other clients' private data or the server's internal computations.

The current implementation has several important limitations. First, the circuit size grows linearly with both the number of model parameters and participating clients, which significantly impacts proof generation time. This could create substantial computational overhead for large-scale FL systems with many participants or complex models.

A more fundamental limitation is that while the proof verifies the mathematical correctness of the aggregation process, it cannot independently verify the authenticity of the input data. To address this, a more comprehensive verification system would be needed. The server should calculate ZKP for both the weighted average and cryptographic hashes of the input data, proving that the same inputs were used for both operations simultaneously [56].

Clients could then verify that their contributions were included by comparing the hash of their submitted data against the list of input hashes proven by the server. This approach would provide end-to-end verification that not only was the averaging done correctly, but it was performed on the intended input data.

Hash functions specifically optimized for ZKP, particularly Mimc7 and Poseidon, are commonly used in zero-knowledge applications [53, 56, 98, 99]. These hash functions are designed to be efficiently represented in arithmetic circuits, making them more suitable for generating ZKP compared to traditional hash functions like SHA-256.

However, implementing this enhanced verification system faces technical challenges. Currently, EZKL does not support efficient ZKP for cryptographic hash functions due to limitations in ONNX data type conversions. Implementing them would require using alternative circuit compilers that provide direct support for these cryptographic primitives. This represents an important area for future development to enable fully verifiable FL systems.

ZKP systems aim for completeness, ensuring valid proofs are verified, and soundness, guaranteeing that accepted proofs represent true statements. However, underconstrained circuits can allow invalid proofs to deceive verifiers. While this issue stems from the SNARK generated by EZKL, rather than the machine learning model itself, EZKL, as a compiler, has an advantage over manually written circuits in mitigating such risks. Nevertheless, its active development status leaves room for potential vulnerabilities.

6.2.3 Hyperledger Fabric SDK

Our implementation leverages the Hyperledger Fabric SDK [95] to enable secure interaction between FL components and the blockchain network. The implementation consists of several key components that work together to maintain the integrity and privacy of the FL process. The class diagram for the Local and Global Model chaincodes is depicted in Figure 17.

The Local Model chaincode handles individual model contributions from vehicles, storing model hashes and metadata while enforcing strict access controls. Each local model entry includes the model's cryptographic hash, training metrics such as the number of samples used, and references to the source global model version.

The Global Model chaincode manages the aggregation process and maintains the evolution of the global model through successive rounds of training. It implements private data collections to protect sensitive model information while making public metadata available to all participants. The data model stored on the ledger is structured to maintain privacy while enabling verification. Public data includes model hashes, round identifiers, and ZKP references, allowing all participants to verify the training process. Global Model chaincode invokes Local Model chaincode as part of the integrity validation of referenced models in aggregation. Private data collections store sensitive information like model weights, with access restricted to specific organizations. The ledger structure uses composite keys to enable efficient querying of model history and relationships between local and global models.

In an ideal scenario, we would perform a comprehensive validation of submitted global models that includes ZKP verification by endorsement peers. This would require implementing a custom endorsement plugin to handle the additional validation logic. However, Hyperledger Fabric's chaincode execution environment imposes strict limitations to ensure deterministic execution, preventing the direct integration of third-party code or remote/API calls within smart contracts.

While EZKL provides ZKP functionality through a Solidity contract implementation (using Rust-based WebAssembly), adapting this for Hyperledger Fabric requires additional engineering effort. A production-ready solution would involve adjusting the Rust code, compiling it to WebAssembly, and creating a Go environment capable of executing this WASM code within an endorsement plugin. This approach would enable proper on-chain verification of ZKPs during the endorsement process.

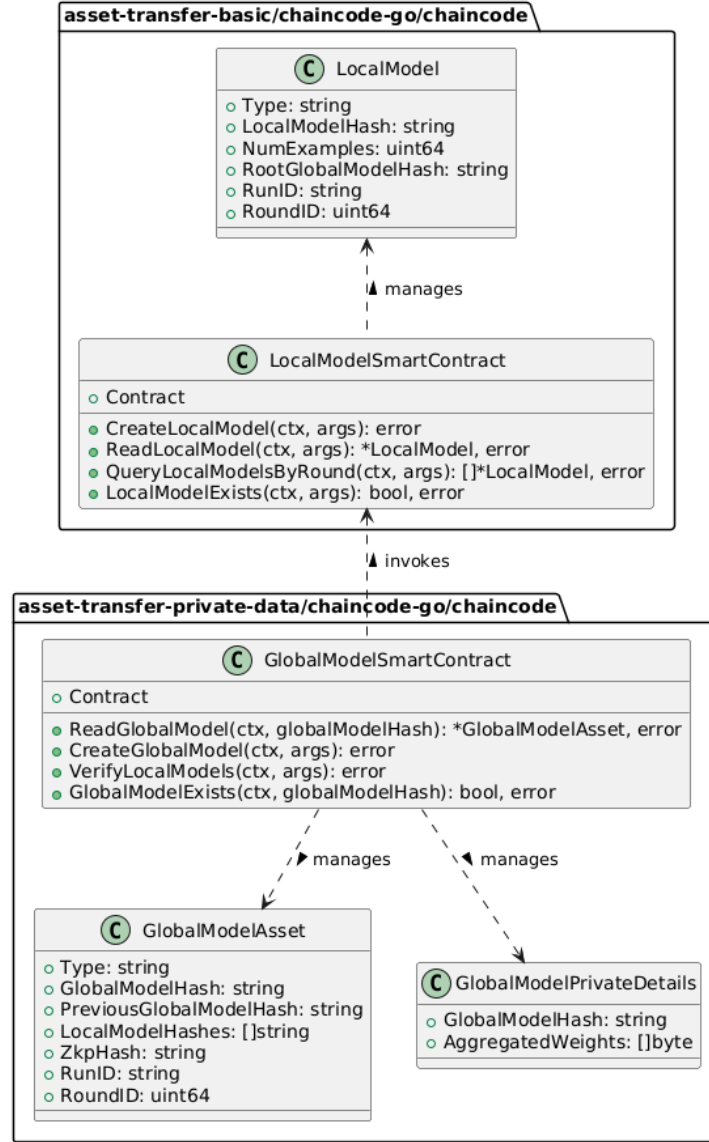


Figure 17. Class diagram for chaincodes implementation

However, for our proof-of-concept implementation, we adopt a simplified approach where ZKP validation occurs off-chain on the client side. This design decision allows us to demonstrate the core functionality of our blockchain-based FL system while avoiding the complexity of implementing custom endorsement plugins. Clients independently verify the ZKPs before accepting and training on new global models, maintaining security guarantees without requiring modifications to the Fabric endorsement process.

A REST API service, implemented in Go, provides a clean interface for interaction

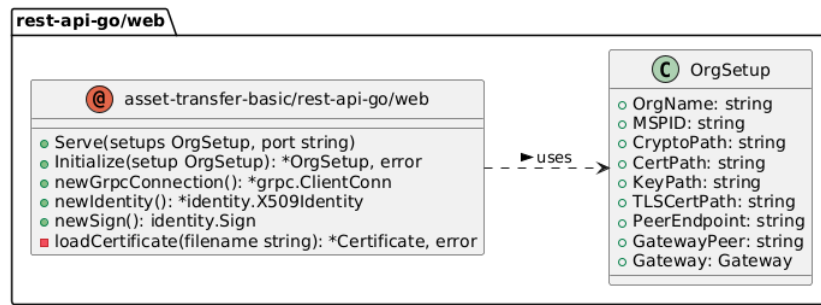


Figure 18. Class diagram for gateway REST API implementation

with the blockchain network. The API exposes endpoints for model submission, retrieval, and verification. For model submission, the API handles the creation of transaction proposals, collection of endorsements, and orderer submission. The verification endpoints allow clients to validate the authenticity of received models by checking model metadata against the ledger records. The class diagram for the REST API is depicted on Figure 18.

Access control is implemented through a combination of Fabric's built-in MSP, chaincode policies, collections policies, and custom chaincode-level checks. The MSP manages identity through X.509 certificates, while chaincode logic and policies enforce fine-grained permissions based on organizational roles and training round status. This dual-layer approach ensures that only authorized participants can contribute to the FL process.

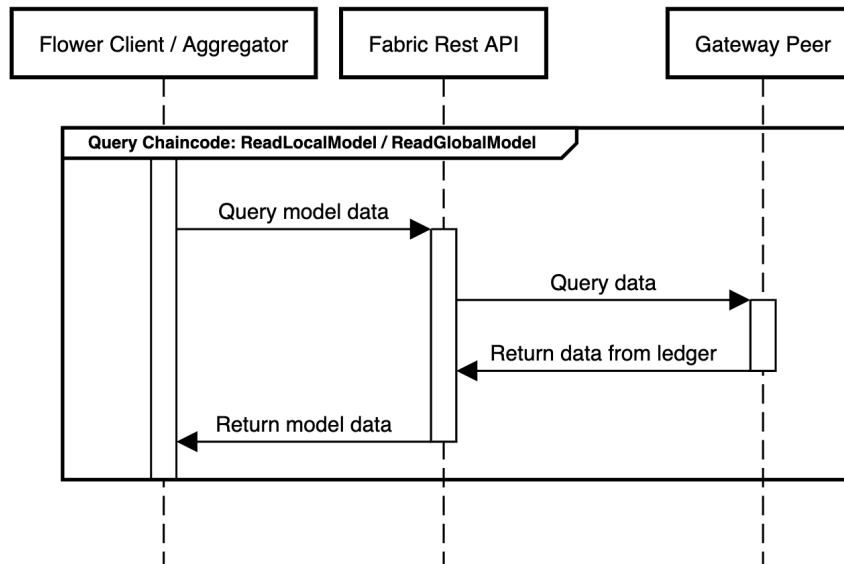


Figure 19. Sequence diagram for the chaincode querying

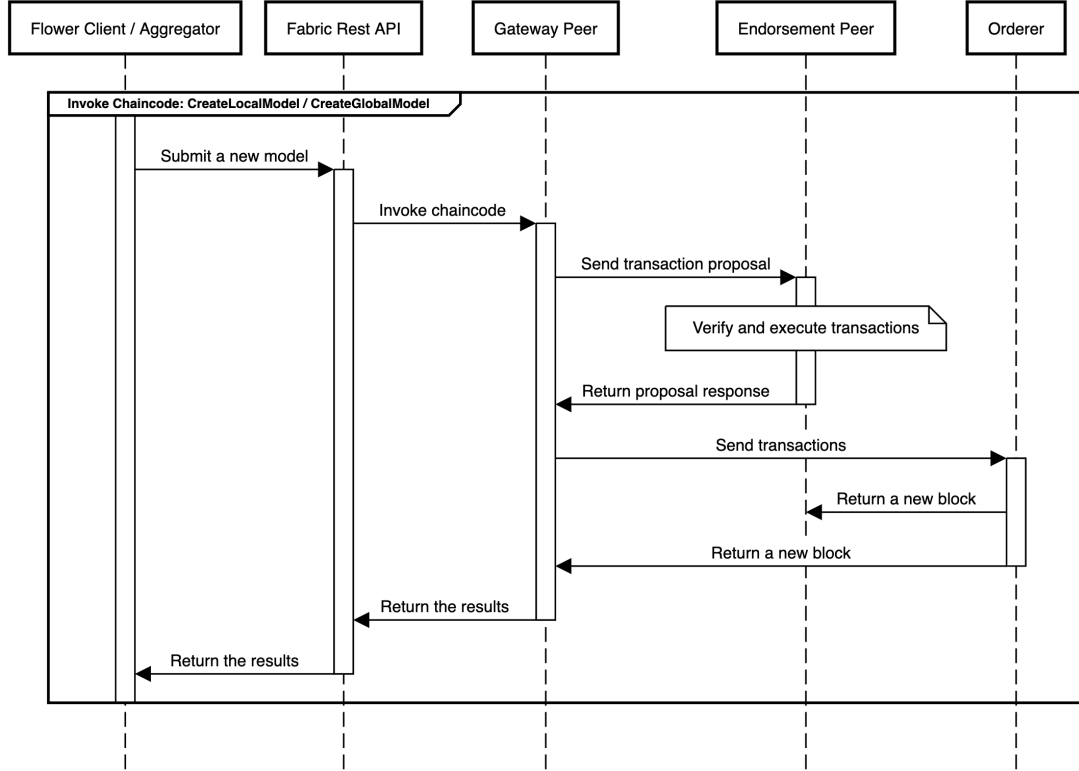


Figure 20. Sequence diagram for the chaincode invocation

Finally, we will cover the implementation of blockchain interactions through our REST API service. Figure 20 shows the chaincode invocation flow, where model submissions from Flower clients or aggregation servers trigger a multi-step transaction process. The REST API service creates and submits transaction proposals to gateway peers, which are forwarded to endorsing peers for verification and execution. After collecting endorsements, transactions are ordered into blocks and distributed back through the network. Figure 19 demonstrates the query flow used to retrieve model data, where read requests are processed directly by gateway peers accessing their local ledger state, so there is no need for full consensus. These security mechanisms ensure that all model updates are properly validated and recorded on the blockchain.

6.2.4 Deployment

The blockchain-based FL system development encompasses a production-like architecture design and its simplified proof-of-concept implementation. The first one presents a design vision for enterprise deployment in dynamic IoV networks, where the system must accommodate constantly changing numbers of vehicles, RSUs, and network peers

with robust scalability and security features. For this thesis work, we implemented a simplified proof-of-concept version that focuses on demonstrating and validating the fundamental interactions between key system components in a controlled environment.

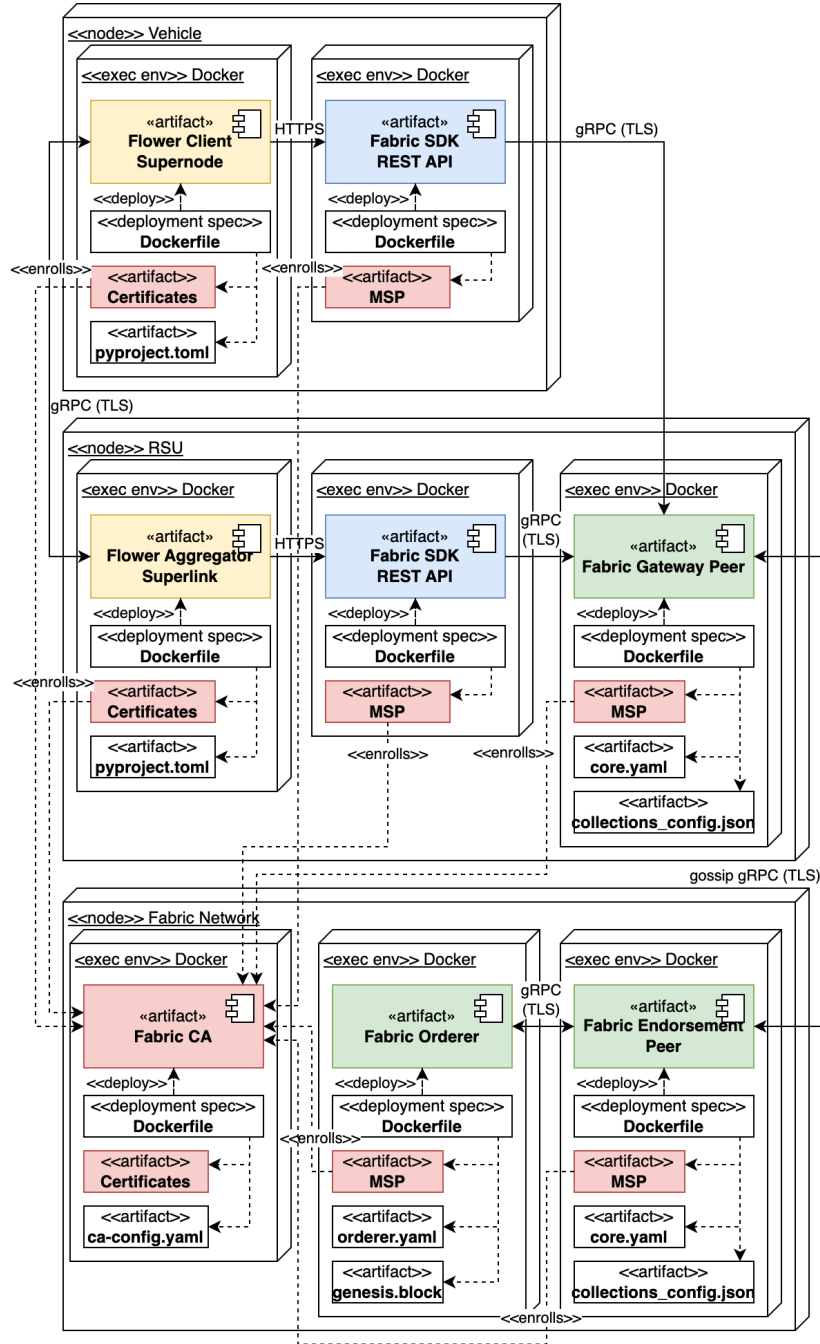


Figure 21. Deployment diagram for production-like system

Production-like Deployment. The deployment diagram, shown in Figure 21, illustrates a production-like blockchain-based FL system organized into three distinct levels: Vehicle nodes, RSU nodes, and the Fabric Network infrastructure. Each level employs Docker containers for component isolation and standardized deployment.

At the top level, Vehicle nodes operate two primary Docker containers. The first container hosts the Flower Client Supernode, which manages FL operations and relies on configurations specified in `pyproject.toml`. Rather than functioning as full blockchain nodes, vehicles interact with the blockchain through the second container with Fabric SDK REST API, authenticating via MSP client credentials. These containers share access to common artifacts. The internal communication between components operates with HTTPS protocol in the isolated network environment. The Vehicle nodes communicate with RSUs through the gRPC (TLS) protocol through direct FL messages and indirect blockchain requests.

The middle level consists of RSU nodes running three Docker containers. The Flower Aggregator Superlink container handles FL model aggregation and ZKP logic. A Fabric Gateway Peer container enables full participation in the blockchain network. The Fabric SDK REST API container manages the interaction between FL logic and blockchain interaction. Like Vehicle nodes, RSUs maintain their MSP certificates and configuration files, including `pyproject.toml` for FL settings and `core.yaml` for peer configuration. RSUs communicate with other FL clients using and Fabric peers with gRPC (TLS) protocol.

The bottom level represents the Fabric Network infrastructure. The core Fabric network infrastructure consists of several essential components. The Certificate Authority manages the public key infrastructure and digital identities, while the Ordering Service handles transaction ordering and block creation. Endorsement peers execute and validate chaincode operations. The architecture shows the minimum required elements, while the actual deployment can scale through channels and organizations partitioning to follow the IoV dynamic nature.

The deployment utilizes Docker Compose to orchestrate the container ecosystem. Each Docker container is defined by its own Dockerfile and deployment specifications, ensuring consistent deployment across the network. Admins maintain core network configurations with initial deployment details and network policies through YAML files: `ca-config.yaml` for the CA, `orderer.yaml` for the ordering service, `core.yaml` for the peers, and `collections-config.json` for private data collection policies.

Communication between components is secured through multiple mechanisms. gRPC with TLS handles secure communication between Fabric components, while HTTPS protects the REST API interactions. The gossip protocol enables peer-to-peer communication between Fabric nodes for ledger synchronization and private data dissemination. There could be many CAs in production that issue certificates for both Fabric MSPs and Flower encrypted connections. Ideally, Flower should automatically manage the authentication list by synchronizing it with Fabric MSP.

This deployment architecture provides the foundation for a production-ready system that can scale to accommodate additional vehicles, RSUs, and Fabric network components as needed, while maintaining security and performance requirements through its containerized structure and well-defined communication protocols.

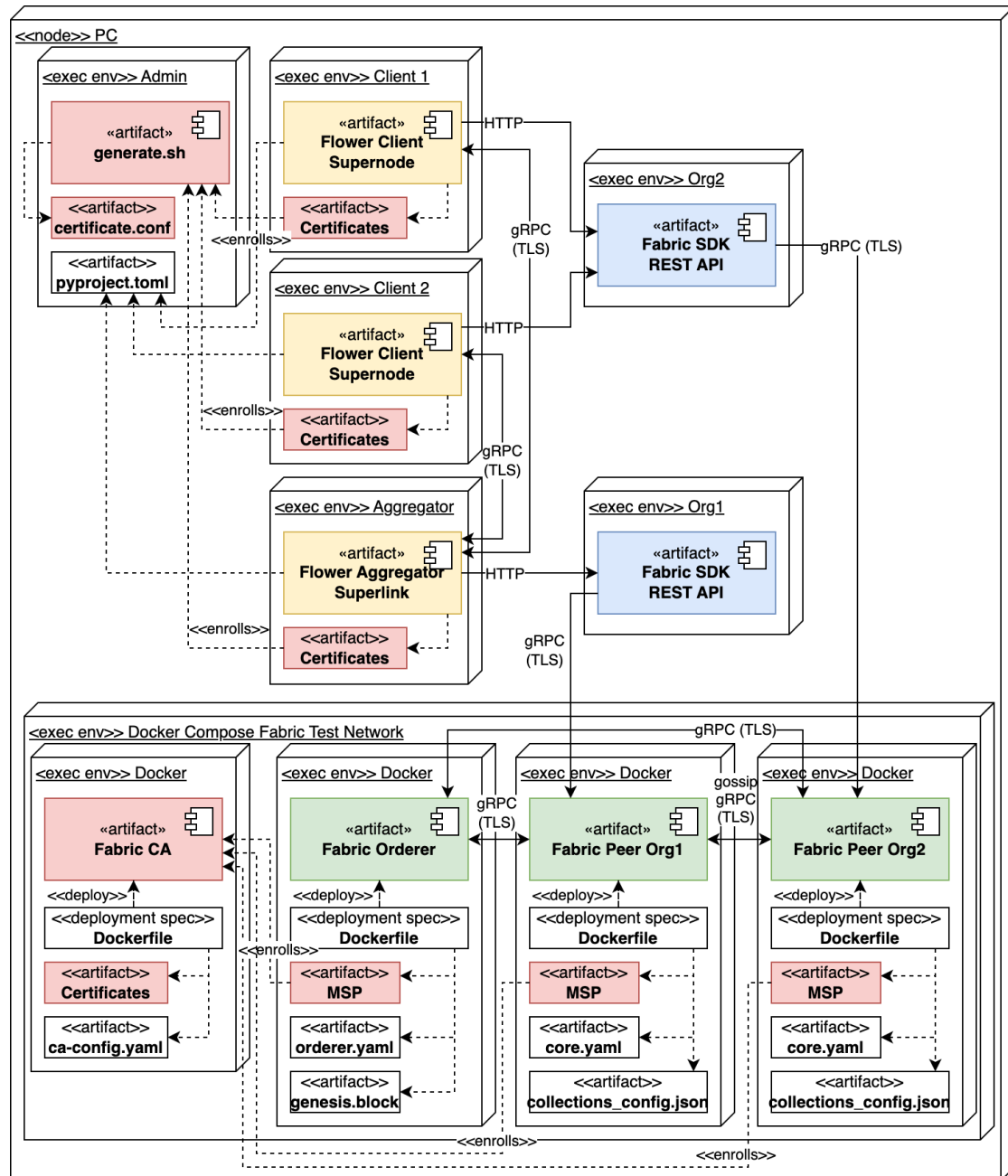


Figure 22. Deployment diagram for proof of concept implementation

Proof-of-Concept Deployment. The proof-of-concept implementation, depicted in Figure 22, simplifies the architecture by consolidating all components onto a single PC. Instead of containerized Vehicle nodes, the system runs multiple Flower Client Supernodes directly in the Python execution environment, sharing a common `pyproject.toml` configuration. An Admin environment handles certificate generation through `generate.sh` scripts, centralizing the security setup process. Similarly, the Aggregator runs as a Python process rather than in a container, demonstrating the core FL functionality in a simplified context.

The Fabric network is implemented as a Docker Compose test network with minimal components, where Org1 represents RSUs and Org2 represents vehicles. Each organization has its own Fabric SDK REST API instance to handle blockchain interactions. The test network maintains essential components: one CA, one Orderer, and two peers (one for each organization), using a genesis block and simplified MSP configuration.

The two implementations serve complementary purposes in validating and deploying the system. While the production architecture handles the complexities of real-world IoV environments with their dynamic topology and scaling requirements, the proof-of-concept implementation provides a controlled environment for validating core system interactions. By demonstrating these interactions in a simplified setting - from FL clients to blockchain consensus - we can verify the system’s fundamental operational principles before scaling to full production deployment. This approach ensures that the essential security and functionality requirements are met in a manageable development environment while maintaining a clear path to production-scale deployment in dynamic vehicular networks.

6.3 Experiment

For our experimental validation, we focused on two critical attack scenarios identified in Chapter 5.3: Unreliable Communication Attacks (SR4) and Model Aggregation Poisoning Attacks (SR5). Our goal was to evaluate how our implementation, combining HLF private network setup and ZKP aggregation verification, addresses the vulnerabilities present in centralized FL solutions.

We used CIFAR-10 as our test dataset, a standard benchmark in computer vision consisting of 60,000 32x32 RGB images across 10 object classes. While our research is task-agnostic and focuses primarily on the FL process itself, CIFAR-10 serves as a reasonable proxy for IoV scenarios where vehicles might need to perform visual recognition tasks (e.g., road sign classification). Each FL client uses a simple CNN model with 62,000 trainable parameters for local training. The training data is partitioned between two clients in our federated setting. The deployment process began with establishing our proof-of-concept blockchain test network using Docker-Compose, creating two organizations: one for vehicle clients and one for the RSU aggregation server. Each organization maintains its own ledger and PDCs. We then initiated two gateway REST

API servers to enable FL-blockchain interaction via HTTP, generated TLS certificates and authentication lists for FL clients, and performed ZKP setup to generate essential prover and verifier keys. Finally, we launched two Flower Supernodes representing clients and one Flower Superlink for the server in separate terminals.

To validate our implementation against Unreliable Communication Attacks (SR4), we conducted comprehensive API testing to verify chaincode integrity checks. This included testing the system's response to various invalid inputs, such as incorrect FL run IDs, invalid round IDs, mismatched global model hashes, and wrong local model hashes. The system successfully detected and rejected these invalid communications, demonstrating its resilience against tampering attempts.

To test Model Aggregation Poisoning Attacks (SR5), we simulated malicious server behavior by injecting fake ZKP witnesses during alternating rounds. During the first round, the system operated normally with a valid global model and legitimate FL steps. In the second round, the server submitted a precomputed malicious model with fake ZKP. When clients attempted to validate this model in the third round, they detected the invalid ZKP, causing the round to fail and preventing the poisoned model from entering the training process.

A complete demonstration of the implementation and attack scenarios can be viewed in our video demo⁸. The full source code, including configuration files and deployment scripts, is publicly available in our GitHub repository⁹.

Our implementation successfully addressed the core vulnerabilities in both attack scenarios through robust HLF security features, ZKP-based verification mechanisms, and comprehensive access controls. The system fully implements 24 security requirements and partially implements 15 security controls, with particularly strong coverage in communication security, access control, and server-side aggregation integrity. However, 14 security requirements remain unaddressed, primarily in areas such as local model training verification, reverse-engineering robustness, and monitoring and anomaly detection practices. For a detailed breakdown of our security implementation, readers can refer to Appendix III, which contains the complete list of security requirements, and Appendix IV, which outlines the specific security controls implemented in our system, including their current implementation status.

6.4 Summary

This chapter presented a comprehensive blockchain-based FL solution for secure data sharing in IoV environments. The solution combines Hyperledger Fabric's enterprise blockchain capabilities with ZKPs to address the security vulnerabilities identified in centralized FL systems. Key accomplishments include:

⁸<https://youtu.be/RxbJgq6MiQE>

⁹<https://github.com/luzanikita/fl-chain-data-sharing>

Multiple design artifacts were produced to document the solution architecture: detailed BPMN diagrams mapping the complete data flow for network initialization, model distribution, local training, and aggregation processes; UML class diagrams illustrating the relationships between FL components, ZKP modules, and blockchain elements; sequence diagrams showing the precise interactions between system components; and deployment diagrams for both production and proof-of-concept implementations.

The design established a robust security architecture featuring distributed RSU nodes, multi-channel organization, and fine-grained access control through MSP and chaincode policies. The implementation leveraged three main components: Flower for FL coordination, EZKL for ZKP generation and verification, and Hyperledger Fabric SDK for secure blockchain infrastructure.

The solution implemented multiple security layers, including TLS-encrypted communication, digital signatures for model verification, private data collections for sensitive information, and ZKP-based verification of model aggregation. While the current implementation has some limitations, particularly around ZKP circuit size and verification of local training, it successfully demonstrates the feasibility of blockchain-based security enhancements for FL systems.

Experimental validation focused on two critical attack scenarios - Unreliable Communication Attacks (SR4) and Model Aggregation Poisoning Attacks (SR5). The implementation fully addressed 24 security requirements and partially implemented 15 security controls, with particularly strong coverage in communication security, access control, and aggregation integrity verification. The proof-of-concept demonstrated the system's ability to detect and prevent both invalid communications and poisoned model submissions by a malicious aggregation server through its multi-layered security approach.

7 Concluding Remarks

This thesis has addressed the critical need for systematic security analysis of FL in specific application contexts, focusing on the IoV as an illustrative use case. While existing research has extensively documented attacks and defenses for FL systems in general, our work contributes by examining how these security challenges manifest in a concrete IoV data sharing scenario, leading to targeted blockchain-based solutions.

The research makes three main contributions. First, we conducted a comprehensive literature review and security risk analysis combining STRIDE methodology with the ISSRM domain model to systematically identify and evaluate security risks specific to IoV-based FL. Second, we demonstrated how blockchain technology and ZKPs can effectively address the identified vulnerabilities while maintaining essential privacy properties. Third, we validated our theoretical framework through a concrete implementation combining Hyperledger Fabric with ZKPs, successfully implementing 24 security requirements and partially addressing 15 more. The findings contribute to bridging the gap between theoretical security models and their practical implementation in specific application contexts.

7.1 Limitations

The development and implementation of our blockchain-based FL solution encountered several significant challenges and limitations across different aspects of the system. At a fundamental level, the decentralized nature of FL in IoV environments presents inherent challenges in monitoring and validating participant actions. Vehicles operate as independent entities that prioritize privacy, making it particularly difficult to verify the integrity of their training processes and detect malicious behaviors without compromising their autonomy. The challenge of identifying the root cause of poisoning attacks becomes especially complex when compromised data indirectly affects model performance after training.

Our research scope consciously excluded several important aspects of FL security. We did not implement specific defenses against poisoning attacks from vehicles or model reverse-engineering attacks from servers, as these would require extensive control over vehicle data and actions, potentially compromising the privacy guarantees fundamental to FL. While sophisticated defense mechanisms like differential privacy, secure aggregation, and anomaly detection exist, their significant computational overhead and complexity make them challenging to implement in resource-constrained IoV environments. Our focus remained on permissioned blockchain and ZKP-based aggregation verification as practical, implementable solutions. Due to time constraints, we haven't performed a comprehensive security analysis of the proposed blockchain-based solution and a comparative analysis with the centralized approach, which would be valuable to explore in future research.

The implementation of ZKPs presented particular technical challenges. While our ZKP integration allows verification of aggregation correctness without exposing individual contributions, the circuit size grows linearly with both model parameters and participating clients, creating substantial computational overhead. In our implementation, prover key files reached sizes up to 5GB, and proof generation could take several minutes, raising concerns about scalability in production environments. Moreover, while the proof verifies mathematical correctness, it cannot independently verify input data authenticity. The current EZKL implementation lacks support for efficient ZKPs of cryptographic hash functions due to ONNX data type conversion limitations, preventing the implementation of a more comprehensive end-to-end verification system.

Our proof-of-concept implementation made several simplifications to demonstrate core functionality. Instead of a fully distributed system, all components were consolidated onto a single PC, with Flower Client Supernodes running directly in the Python environment rather than in containers. The Fabric network was implemented as a minimal test network with basic components. Critical aspects like ZKP storage were simplified to on-device storage rather than implementing more robust solutions like PDC or IPFS, leading to missed opportunities for certain integrity checks. While these simplifications allowed us to validate core system interactions, they leave several challenges to be addressed in a production-ready deployment, particularly around scalability, resource management, and robust storage solutions.

Security remains a multifaceted challenge, with breaches possible across physical, network, and application levels. The difficulty in tracing server actions and verifying operations performed during validation, aggregation, and system management creates opportunities for unauthorized modifications that could propagate throughout the IoV system. The fundamental challenge lies in balancing robust security measures with privacy protection while maintaining acceptable model performance and system efficiency. These limitations and challenges point to several areas requiring further research and development before such a system could be deployed in production environments.

7.2 Answers to Research Questions

The thesis explored how blockchain technology could improve security in FL data sharing for the IoV. Following design science guidelines, we approached each research question systematically.

RQ1: What are the security risks of centralized FL data sharing? Following guidelines G2 and G5, we conducted a thorough literature review using snowball sampling (Chapter 4) and security analysis (Chapter 5) to identify major risks in centralized FL systems. The analysis revealed client-side risks like data/model poisoning, server-side risks from compromised aggregators, and man-in-the-middle attacks targeting communication

and model privacy.

RQ2: How to build a threat model using blockchain technology to mitigate the security risks of centralized FL data sharing in the context of IoV? Addressing guidelines G3, G4 and G5, our security assessment in Chapter 5 developed several key artifacts. We identified critical assets, mapped vulnerabilities and threats using STRIDE methodology, and created ISSRM and CIA models. This systematic analysis led to proposing blockchain-based countermeasures including distributed ledger, smart contracts, and ZKPs.

RQ3: How to design and implement secure data sharing using blockchain-based FL? Following guidelines G4, G6 and G7, we designed and implemented a comprehensive solution detailed in Chapter 6. We integrated Hyperledger Fabric with FL coordination through the Flower framework, developed ZKPs for verifiable aggregation, and created supporting artifacts like UML diagrams for architecture. The implementation was validated against key attack scenarios, demonstrating how blockchain technology enhances security while maintaining privacy and efficiency requirements of FL in IoV environments.

This structured approach, guided by design science principles, enabled us to systematically address the security challenges in FL-based data sharing while ensuring both academic rigor and practical applicability. The research shows that blockchain technology, particularly when combined with ZKPs and proper identity management, can enhance the security of FL systems in IoV environments. The implemented solution successfully addresses the main vulnerabilities of centralized architectures while maintaining the privacy and efficiency requirements of FL.

7.3 Future Work

Several promising directions for future research emerge from our work on blockchain-based FL for IoV environments. A primary area for investigation is the enhancement of our ZKP implementation to achieve comprehensive end-to-end verification. This would involve developing efficient ZKP circuits for cryptographic hash functions, enabling verification of both aggregation correctness and input data authenticity.

Practical deployment considerations also warrant further investigation. Implementation of differential privacy mechanisms, secure aggregation protocols, and robust anomaly detection systems would enhance the system's resistance to various attacks. Development of efficient storage solutions for ZKPs, possibly utilizing IPFS or other distributed storage systems, would be crucial for production environments. Addition-

ally, comprehensive benchmarking and optimization studies would help understand and improve system performance under realistic IoV conditions.

Additionally, a comprehensive security analysis of the proposed blockchain-based architecture would be valuable. This analysis should be complemented by a detailed comparative study with centralized approaches, evaluating security guarantees, computational overhead, scalability, and overall system performance across different deployment scenarios. Such research would provide valuable insights into the trade-offs between centralized and decentralized approaches, helping organizations make informed decisions about FL architectures based on their specific security requirements and resource constraints.

Defense mechanisms identified in our literature review but not implemented in our current system deserve exploration. These include robust training techniques for poison detection, advanced membership inference attack prevention, and sophisticated free-rider detection mechanisms. Integration of these defenses while maintaining system efficiency and scalability presents significant research opportunities. Furthermore, development of incentive mechanisms to encourage honest participation and high-quality contributions could enhance system reliability and effectiveness.

The exploration of alternative blockchain platforms presents another valuable research direction. While Hyperledger Fabric proved suitable for our proof-of-concept, platforms like NEAR Protocol, with its AI-centric architecture and improved scalability, could offer advantages for FL applications. Similarly, Polkadot's Substrate framework provides extensive customization options and WASM support, potentially enabling more efficient on-chain verification of ZKPs and smarter contract execution.

A valuable enhancement for blockchain-based FL would be integrating incentive and reputation mechanisms. Using blockchain's asset management capabilities, a token-based system could reward quality contributions while reputation scores based on factors like model quality and participation history could influence aggregation weights. Smart contracts would automate reward distribution according to verifiable metrics. This would create a self-regulating system that encourages honest participation by giving more influence to reliable participants while making attacks more costly.

Emerging cryptographic technologies offer promising avenues for enhancing security and privacy. Trusted Execution Environments (TEE) could provide hardware-based guarantees for the integrity of both server-side aggregation and client-side training processes. Fully Homomorphic Encryption (FHE) could enable secure computation on encrypted data, potentially revolutionizing how model updates are processed and aggregated while maintaining privacy. These technologies could be particularly valuable in addressing the poisoning and model inversion attacks we identified but did not fully address in our current implementation.

The rapid growth of Large Language Models (LLMs) presents intriguing opportunities for FL research. As organizations seek to collaboratively train and improve these

massive models while protecting proprietary data, our blockchain-based framework could be adapted to support secure, verifiable training of LLMs. This would require addressing new challenges around model size, computational requirements, and data privacy at an unprecedented scale.

The continuing evolution of IoV environments and their security requirements suggests the need for adaptive security measures. Future research could explore dynamic security policies that adjust based on threat levels, traffic conditions, and system resources. Investigation of cross-chain interoperability could enable FL across different vehicular networks while maintaining security guarantees. These advancements would move us closer to realizing the full potential of secure, efficient, and privacy-preserving FL in real-world IoV deployments.

List of References

- [1] Lu Y, Huang X, Zhang K, Maharjan S, Zhang Y. Blockchain Empowered Asynchronous Federated Learning for Secure Data Sharing in Internet of Vehicles. *IEEE Transactions on Vehicular Technology*. 2020 Apr;69(4):4298-311. Available from: <https://ieeexplore.ieee.org/document/8998397/>.
- [2] Khattak HA, Farman H, Jan B, Din IU. Toward Integrating Vehicular Clouds with IoT for Smart City Services. *IEEE Network*. 2019 Mar;33(2):65-71. Available from: <https://ieeexplore.ieee.org/document/8675174/>.
- [3] Du Z, Wu C, Yoshinaga T, Yau KLA, Ji Y, Li J. Federated Learning for Vehicular Internet of Things: Recent Advances and Open Issues. *IEEE Open Journal of the Computer Society*. 2020;1:45-61. Available from: <https://ieeexplore.ieee.org/document/9086790/>.
- [4] Lu Y, Huang X, Dai Y, Maharjan S, Zhang Y. Blockchain and Federated Learning for Privacy-Preserved Data Sharing in Industrial IoT. *IEEE Transactions on Industrial Informatics*. 2020 Jun;16(6):4177-86. Available from: <https://ieeexplore.ieee.org/document/8843900/>.
- [5] Nguyen DC, Ding M, Pham QV, Pathirana PN, Le LB, Seneviratne A, et al.. Federated Learning Meets Blockchain in Edge Computing: Opportunities and Challenges. *arXiv*; 2021. ArXiv:2104.01776 [cs, eess]. Available from: <http://arxiv.org/abs/2104.01776>.
- [6] Danba S, Bao J, Han G, Guleng S, Wu C. Toward Collaborative Intelligence in IoV Systems: Recent Advances and Open Issues. *Sensors*. 2022 Sep;22(18):6995. Available from: <https://www.mdpi.com/1424-8220/22/18/6995>.
- [7] Hevner AR, March ST, Park J, Ram S. Design science in information systems research. *MIS Q*. 2004 Mar;28(1):75–105.
- [8] Wieringa RJ. Design Science Methodology for Information Systems and Software Engineering. Berlin, Heidelberg: Springer Berlin Heidelberg; 2014. Available from: <https://link.springer.com/10.1007/978-3-662-43839-8>.
- [9] Ferrag MA, Friha O, Maglaras L, Janicke H, Shu L. Federated Deep Learning for Cyber Security in the Internet of Things: Concepts, Applications, and Experimental Analysis. *IEEE Access*. 2021;9:138509-42. Available from: <https://ieeexplore.ieee.org/document/9562531/>.
- [10] Wang Z, Hu Q. Blockchain-based Federated Learning: A Comprehensive Survey. *arXiv*; 2021. ArXiv:2110.02182 [cs]. Available from: <http://arxiv.org/abs/2110.02182>.

- [11] McMahan HB, Moore E, Ramage D, Hampson S, Arcas B. Communication-Efficient Learning of Deep Networks from Decentralized Data. arXiv; 2023. ArXiv:1602.05629 [cs]. Available from: <http://arxiv.org/abs/1602.05629>.
- [12] Agrawal S, Sarkar S, Alazab M, Maddikunta PKR, Gadekallu TR, Pham QV. Genetic CFL: Hyperparameter Optimization in Clustered Federated Learning. Computational Intelligence and Neuroscience. 2021 Nov;2021:1-10. Available from: <https://www.hindawi.com/journals/cin/2021/7156420/>.
- [13] Chhetri B, Gopali S, Olapojoye R, Dehbash S, Namin AS. A Survey on Blockchain-Based Federated Learning and Data Privacy. arXiv; 2023. ArXiv:2306.17338 [cs]. Available from: <http://arxiv.org/abs/2306.17338>.
- [14] Kang J, Yu R, Huang X, Wu M, Maharjan S, Xie S, et al. Blockchain for Secure and Efficient Data Sharing in Vehicular Edge Computing and Networks. IEEE Internet of Things Journal. 2019 Jun;6(3):4660-70. Available from: <https://ieeexplore.ieee.org/document/8489897/>.
- [15] Iqbal M. Reference Framework for Managing Security Risks Using Blockchain;. Accessed: 2024-01-07. Available from: https://www.researchgate.net/publication/371874161_Reference_Framework_for_Managing_Security_Risks_Using_Blockchain.
- [16] Dubois Heymans P, Mayer N, Matulevičius R. A Systematic Approach to Define the Domain of Information System Security Risk Management. In: Nurcan S, Salinesi C, Souveyet C, Ralyté J, editors. Intentional Perspectives on Information Systems Engineering. Berlin, Heidelberg: Springer Berlin Heidelberg; 2010. p. 289-306. Available from: http://link.springer.com/10.1007/978-3-642-12544-7_16.
- [17] Matulevičius R. Fundamentals of Secure System Modelling; 2017. Journal Abbreviation: Fundamentals of Secure System Modelling Publication Title: Fundamentals of Secure System Modelling.
- [18] Haddaji A, Ayed S, Fourati LC. Artificial Intelligence techniques to mitigate cyber-attacks within vehicular networks: Survey. Computers and Electrical Engineering. 2022 Dec;104:108460. Available from: <https://linkinghub.elsevier.com/retrieve/pii/S0045790622006759>.
- [19] Esposito C, Sperli G, Moscato V, Zhao Z. On Attacks To Federated Learning and a Blockchain-empowered Protection. In: 2022 IEEE 19th Annual Consumer Communications & Networking Conference (CCNC). Las Vegas, NV, USA: IEEE; 2022. p. 1-6. Available from: <https://ieeexplore.ieee.org/document/9700723/>.

- [20] Takahashi H. AIJack: Let's Hijack AI! Security and Privacy Risk Simulator for Machine Learning. arXiv; 2024. ArXiv:2312.17667 [cs]. Available from: <http://arxiv.org/abs/2312.17667>.
- [21] Cao X, Gong NZ. MPAF: Model Poisoning Attacks to Federated Learning based on Fake Clients. arXiv; 2022. ArXiv:2203.08669 [cs]. Available from: <http://arxiv.org/abs/2203.08669>.
- [22] OWASP Machine Learning Security Top Ten 2023 | ML02:2023 Data Poisoning Attack | OWASP Foundation;. Accessed: 2024-07-02. Available from: https://owasp.org/www-project-machine-learning-security-top-10/docs/ML02_2023-Data_Poisoning_Attack.html.
- [23] OWASP Machine Learning Security Top Ten 2023 | ML10:2023 Model Poisoning | OWASP Foundation;. Accessed: 2024-07-02. Available from: https://owasp.org/www-project-machine-learning-security-top-10/docs/ML10_2023-Model_Poisoning.html.
- [24] OWASP Machine Learning Security Top Ten 2023 | ML08:2023 Model Skewing | OWASP Foundation;. Accessed: 2024-07-02. Available from: https://owasp.org/www-project-machine-learning-security-top-10/docs/ML08_2023-Model_Skewing.html.
- [25] OWASP Machine Learning Security Top Ten 2023 | ML07:2023 Transfer Learning Attack | OWASP Foundation;. Accessed: 2024-07-02. Available from: https://owasp.org/www-project-machine-learning-security-top-10/docs/ML07_2023-Transfer_Learning_Attack.html.
- [26] Xie C, Huang K, Chen PY, Li B. DBA: Distributed Backdoor Attacks against Federated Learning; 2019. Available from: <https://openreview.net/forum?id=rkgyS0VFvr>.
- [27] Bagdasaryan E, Veit A, Hua Y, Estrin D, Shmatikov V. How To Backdoor Federated Learning. arXiv; 2019. ArXiv:1807.00459 [cs]. Available from: <http://arxiv.org/abs/1807.00459>.
- [28] Lin J, Du M, Liu J. Free-riders in Federated Learning: Attacks and Defenses. arXiv; 2019. ArXiv:1911.12560 [cs, stat]. Available from: <http://arxiv.org/abs/1911.12560>.
- [29] OWASP Machine Learning Security Top Ten 2023 | ML03:2023 Model Inversion Attack | OWASP Foundation;. Accessed: 2024-07-03. Available from: https://owasp.org/www-project-machine-learning-security-top-10/docs/ML03_2023-Model_Inversion_Attack.html.

- [30] Takahashi H, Liu J, Liu Y. Breaching FedMD: Image Recovery via Paired-Logits Inversion Attack. arXiv; 2023. ArXiv:2304.11436 [cs]. Available from: <http://arxiv.org/abs/2304.11436>.
- [31] Geiping J, Bauermeister H, Dröge H, Moeller M. Inverting Gradients - How easy is it to break privacy in federated learning? In: Advances in Neural Information Processing Systems. vol. 33. Curran Associates, Inc.; 2020. p. 16937-47. Available from: <https://proceedings.neurips.cc/paper/2020/hash/c4ede56bbd98819ae6112b20ac6bf145-Abstract.html>.
- [32] Hitaj B, Ateniese G, Perez-Cruz F. Deep Models Under the GAN: Information Leakage from Collaborative Deep Learning. arXiv; 2017. ArXiv:1702.07464 [cs, stat]. Available from: <http://arxiv.org/abs/1702.07464>.
- [33] Zhu L, Liu Z, Han S. Deep Leakage from Gradients. arXiv; 2019. ArXiv:1906.08935 [cs, stat]. Available from: <http://arxiv.org/abs/1906.08935>.
- [34] Zhao B, Mopuri KR, Bilen H. iDLG: Improved Deep Leakage from Gradients. arXiv; 2020. ArXiv:2001.02610 [cs, stat]. Available from: <http://arxiv.org/abs/2001.02610>.
- [35] Yin H, Mallya A, Vahdat A, Alvarez JM, Kautz J, Molchanov P. See through Gradients: Image Batch Recovery via GradInversion. In: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Nashville, TN, USA: IEEE; 2021. p. 16332-41. Available from: <https://ieeexplore.ieee.org/document/9577731/>.
- [36] Wei W, Liu L, Loper M, Chow KH, Gursoy ME, Truex S, et al.. A Framework for Evaluating Gradient Leakage Attacks in Federated Learning. arXiv; 2020. ArXiv:2004.10397 [cs, stat]. Available from: <http://arxiv.org/abs/2004.10397>.
- [37] Li O, Sun J, Yang X, Gao W, Zhang H, Xie J, et al.. Label Leakage and Protection in Two-party Split Learning. arXiv; 2022. ArXiv:2102.08504 [cs]. Available from: <http://arxiv.org/abs/2102.08504>.
- [38] OWASP Machine Learning Security Top Ten 2023 | ML04:2023 Membership Inference Attack | OWASP Foundation;. Available from: https://owasp.org/www-project-machine-learning-security-top-10/docs/ML04_2023-Membership_Inference_Attack.html.

- [39] Shokri R, Stronati M, Song C, Shmatikov V. Membership Inference Attacks against Machine Learning Models. arXiv; 2017. ArXiv:1610.05820 [cs, stat]. Available from: <http://arxiv.org/abs/1610.05820>.
- [40] Gu Y, Bai Y, Xu S. CS-MIA: Membership inference attack based on prediction confidence series in federated learning. Journal of Information Security and Applications. 2022 Jun;67:103201. Available from: <https://linkinghub.elsevier.com/retrieve/pii/S2214212622000801>.
- [41] Suri A, Kanani P, Marathe VJ, Peterson DW. Subject Membership Inference Attacks in Federated Learning. arXiv; 2023. ArXiv:2206.03317 [cs]. Available from: <http://arxiv.org/abs/2206.03317>.
- [42] OWASP Machine Learning Security Top Ten 2023 | ML09:2023 Output Integrity Attack | OWASP Foundation;. Accessed: 2024-07-02. Available from: https://owasp.org/www-project-machine-learning-security-top-10/docs/ML09_2023-Output_Integrity_Attack.html.
- [43] OWASP Machine Learning Security Top Ten 2023 | ML06:2023 AI Supply Chain Attacks | OWASP Foundation;. Accessed: 2024-06-17. Available from: https://owasp.org/www-project-machine-learning-security-top-10/docs/ML06_2023-AI_Supply_Chain_Attacks.html.
- [44] Exploitation for Client Execution, Technique T1203 - Enterprise | MITRE ATT&CK®;. Accessed: 2024-06-17. Available from: <https://attack.mitre.org/techniques/T1203/>.
- [45] Wang X, Zhu H, Ning Z, Guo L, Zhang Y. Blockchain Intelligence for Internet of Vehicles: Challenges and Solutions. IEEE Communications Surveys & Tutorials. 2023;25(4):2325-55. Conference Name: IEEE Communications Surveys & Tutorials. Available from: <https://ieeexplore-ieee-org.ezproxy.utlib.ut.ee/document/10217209>.
- [46] Song L, Sun G, Yu H, Du X, Guizani M. FBIA: A Fog-Based Identity Authentication Scheme for Privacy Preservation in Internet of Vehicles. IEEE Transactions on Vehicular Technology. 2020 May;69(5):5403-15. Conference Name: IEEE Transactions on Vehicular Technology. Available from: <https://ieeexplore-ieee-org.ezproxy.utlib.ut.ee/document/9020102>.
- [47] Tian Jf, Ni R. An identity authentication and key agreement protocol for the internet of vehicles based on trusted cloud-edge-terminal architecture. Vehicular Communications. 2024 Oct;49:100825. Available from: <https://www.sciencedirect.com/science/article/pii/S2214209624001001>.

- [48] Huang A, Liu Y, Chen T, Zhou Y, Sun Q, Chai H, et al. StarFL: Hybrid Federated Learning Architecture for Smart Urban Computing. *ACM Transactions on Intelligent Systems and Technology*. 2021 Aug;12(4):1-23. Available from: <https://dl.acm.org/doi/10.1145/3467956>.
- [49] Zhu J, Cao J, Saxena D, Jiang S, Ferradi H. Blockchain-empowered Federated Learning: Challenges, Solutions, and Future Directions. *ACM Computing Surveys*. 2023 Nov;55(11):1-31. Available from: <https://dl.acm.org/doi/10.1145/3570953>.
- [50] Oktian YE, Lee SG. Blockchain-Based Federated Learning System: A Survey on Design Choices. *Sensors*. 2023 Jun;23(12):5658. Available from: <https://www.mdpi.com/1424-8220/23/12/5658>.
- [51] Qu Y, Uddin MP, Gan C, Xiang Y, Gao L, Yearwood J. Blockchain-enabled Federated Learning: A Survey. *ACM Computing Surveys*. 2023 Apr;55(4):1-35. Available from: <https://dl.acm.org/doi/10.1145/3524104>.
- [52] Ramanan P, Nakayama K. BAFFLE : Blockchain Based Aggregator Free Federated Learning. In: 2020 IEEE International Conference on Blockchain (Blockchain); 2020. p. 72-81. Available from: <https://ieeexplore.ieee.org/document/9284684>.
- [53] Zhang Y, Tang Y, Zhang Z, Li M, Li Z, Khan S, et al. Blockchain-Based Practical and Privacy-Preserving Federated Learning with Verifiable Fairness. *Mathematics*. 2023 Jan;11(5):1091. Number: 5 Publisher: Multidisciplinary Digital Publishing Institute. Available from: <https://www.mdpi.com/2227-7390/11/5/1091>.
- [54] Mugunthan V, Rahman R, Kagal L. BlockFLow: An Accountable and Privacy-Preserving Solution for Federated Learning. *arXiv*; 2020. ArXiv:2007.03856 [cs, stat]. Available from: <http://arxiv.org/abs/2007.03856>.
- [55] Li Y, Chen C, Liu N, Huang H, Zheng Z, Yan Q. A Blockchain-Based Decentralized Federated Learning Framework with Committee Consensus. *IEEE Network*. 2021 Jan;35(1):234-41. Available from: <https://ieeexplore.ieee.org/document/9293091/>.
- [56] Ahmadi M, Nourmohammadi R. zkDFL: An efficient and privacy-preserving decentralized federated learning with zero-knowledge proof. *arXiv*; 2024. ArXiv:2312.04579 [cs]. Available from: <http://arxiv.org/abs/2312.04579>.
- [57] Nguyen T, Thai MT. Preserving Privacy and Security in Federated Learning. *IEEE/ACM Transactions on Networking*. 2024 Feb;32(1):833-43. ArXiv:2202.03402 [cs]. Available from: <http://arxiv.org/abs/2202.03402>.

- [58] Angel S, Blumberg AJ, Ioannidis E, Woods J. Efficient Representation of Numerical Optimization Problems for SNARKs; 2021. Publication info: Published elsewhere. Minor revision. USENIX Security 2022. Available from: <https://eprint.iacr.org/2021/1436>.
- [59] Smahi A, Li H, Yang Y, Yang X, Lu P, Zhong Y, et al. BV-ICVs: A privacy-preserving and verifiable federated learning framework for V2X environments using blockchain and zkSNARKs. *Journal of King Saud University - Computer and Information Sciences*. 2023 Jun;35(6):101542. Available from: <https://linkinghub.elsevier.com/retrieve/pii/S1319157823000885>.
- [60] Abadi M, Chu A, Goodfellow I, McMahan HB, Mironov I, Talwar K, et al. Deep Learning with Differential Privacy. In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*; 2016. p. 308-18. ArXiv:1607.00133 [cs, stat]. Available from: <http://arxiv.org/abs/1607.00133>.
- [61] Li T, Zaheer M, Reddi SJ, Smith V. Private Adaptive Optimization with Side Information. arXiv; 2022. ArXiv:2202.05963 [cs, stat]. Available from: <http://arxiv.org/abs/2202.05963>.
- [62] Wang W, Wang T, Wang L, Luo N, Zhou P, Song D, et al.. DPlis: Boosting Utility of Differentially Private Deep Learning via Randomized Smoothing. arXiv; 2021. ArXiv:2103.01496 [cs, stat]. Available from: <http://arxiv.org/abs/2103.01496>.
- [63] Ma Y, Woods J, Angel S, Polychroniadou A, Rabin T. Flamingo: Multi-Round Single-Server Secure Aggregation with Applications to Private Federated Learning; 2023. Publication info: Published elsewhere. Minor revision. IEEE S&P (Oakland) 2023. Available from: <https://eprint.iacr.org/2023/486>.
- [64] Guan Z, Zhao Y, Wan Z, Han J. OPSA: Efficient and Verifiable One-Pass Secure Aggregation with TEE for Federated Learning; 2024. Publication info: Preprint. Available from: <https://eprint.iacr.org/2024/476>.
- [65] Taiello R, Önen M, Gritti C, Lorenzi M. Let Them Drop: Scalable and Efficient Federated Learning Solutions Agnostic to Client Stragglers; 2024. Publication info: Preprint. Available from: <https://eprint.iacr.org/2024/942>.
- [66] Lecuyer M, Atlidakis V, Geambasu R, Hsu D, Jana S. Certified Robustness to Adversarial Examples with Differential Privacy. arXiv; 2019. ArXiv:1802.03471 [cs, stat]. Available from: <http://arxiv.org/abs/1802.03471>.

- [67] Chen Y, Wang S, Jiang W, Cidon A, Jana S. Cost-Aware Robust Tree Ensembles for Security Applications. arXiv; 2021. ArXiv:1912.01149 [cs]. Available from: <http://arxiv.org/abs/1912.01149>.
- [68] Xie C, Koyejo S, Gupta I. Zeno: Distributed Stochastic Gradient Descent with Suspicion-based Fault-tolerance. In: Proceedings of the 36th International Conference on Machine Learning. PMLR; 2019. p. 6893-901. ISSN: 2640-3498. Available from: <https://proceedings.mlr.press/v97/xie19b.html>.
- [69] Zong B, Song Q, Min MR, Cheng W, Lumezanu C, Cho D, et al. Deep Autoencoding Gaussian Mixture Model for Unsupervised Anomaly Detection; 2018. Available from: <https://openreview.net/forum?id=BJJLHbb0->.
- [70] Wang T, Zhang Y, Jia R. Improving Robustness to Model Inversion Attacks via Mutual Information Regularization. arXiv; 2020. ArXiv:2009.05241 [cs]. Available from: <http://arxiv.org/abs/2009.05241>.
- [71] Sun J, Li A, Wang B, Yang H, Li H, Chen Y. Soteria: Provable Defense against Privacy Leakage in Federated Learning from Representation Perspective. In: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Nashville, TN, USA: IEEE; 2021. p. 9307-15. Available from: <https://ieeexplore.ieee.org/document/9578192/>.
- [72] Fung C, Yoon CJM, Beschastnikh I. Mitigating Sybils in Federated Learning Poisoning. arXiv; 2020. ArXiv:1808.04866 [cs, stat]. Available from: <http://arxiv.org/abs/1808.04866>.
- [73] CAPEC - CAPEC-151: Identity Spoofing (Version 3.9);. Accessed: 2024-10-04. Available from: <https://capec.mitre.org/data/definitions/151.html>.
- [74] Valid Accounts, Technique T1078 - Enterprise | MITRE ATT&CK®;. Accessed: 2024-10-04. Available from: <https://attack.mitre.org/techniques/T1078/>.
- [75] CAPEC - CAPEC-165: File Manipulation (Version 3.9);. Accessed: 2024-10-04. Available from: <https://capec.mitre.org/data/definitions/165.html>.
- [76] Data Manipulation: Runtime Data Manipulation, Sub-technique T1565.003 - Enterprise | MITRE ATT&CK®;. Accessed: 2024-10-04. Available from: <https://attack.mitre.org/techniques/T1565/003/>.
- [77] Data Manipulation: Transmitted Data Manipulation, Sub-technique T1565.002 - Enterprise | MITRE ATT&CK®;. Accessed: 2024-10-04. Available from: <https://attack.mitre.org/techniques/T1565/002/>.

- [78] Adversary-in-the-Middle, Technique T1557 - Enterprise | MITRE ATT&CK®;. Accessed: 2024-10-04. Available from: <https://attack.mitre.org/techniques/T1557/>.
- [79] Data Manipulation: Stored Data Manipulation, Sub-technique T1565.001 - Enterprise | MITRE ATT&CK®;. Accessed: 2024-10-04. Available from: <https://attack.mitre.org/techniques/T1565/001/>.
- [80] CAPEC - CAPEC-184: Software Integrity Attack (Version 3.9);. Accessed: 2024-10-04. Available from: <https://capec.mitre.org/data/definitions/184.html>.
- [81] Indicator Removal, Technique T1070 - Enterprise | MITRE ATT&CK®;. Accessed: 2024-10-04. Available from: <https://attack.mitre.org/techniques/T1070/>.
- [82] Repudiation Attack | OWASP Foundation;. Accessed: 2024-10-04. Available from: https://owasp.org/www-community/attacks/Repudiation_Attack.
- [83] CAPEC - CAPEC-93: Log Injection-Tampering-Forging (Version 3.9);. Accessed: 2024-10-04. Available from: <https://capec.mitre.org/data/definitions/93.html>.
- [84] CAPEC - CAPEC-180: Exploiting Incorrectly Configured Access Control Security Levels (Version 3.9);. Accessed: 2024-10-04. Available from: <https://capec.mitre.org/data/definitions/180.html>.
- [85] CAPEC - CAPEC-94: Adversary in the Middle (AiTM) (Version 3.9);. Accessed: 2024-10-04. Available from: <https://capec.mitre.org/data/definitions/94.html>.
- [86] Automated Collection, Technique T1119 - Enterprise | MITRE ATT&CK®;. Accessed: 2024-10-04. Available from: <https://attack.mitre.org/techniques/T1119/>.
- [87] Automated Exfiltration, Technique T1020 - Enterprise | MITRE ATT&CK®;. Accessed: 2024-10-04. Available from: <https://attack.mitre.org/techniques/T1020/>.
- [88] CAPEC - CAPEC-125: Flooding (Version 3.9);. Accessed: 2024-10-04. Available from: <https://capec.mitre.org/data/definitions/125.html>.
- [89] Network Denial of Service: Direct Network Flood, Sub-technique T1498.001 - Enterprise | MITRE ATT&CK®;. Accessed: 2024-10-04. Available from: <https://attack.mitre.org/techniques/T1498/001/>.

- [90] Endpoint Denial of Service, Technique T1499 - Enterprise | MITRE ATT&CK®;. Accessed: 2024-10-04. Available from: <https://attack.mitre.org/techniques/T1499/>.
- [91] Resource Hijacking, Technique T1496 - Enterprise | MITRE ATT&CK®;. Accessed: 2024-10-04. Available from: <https://attack.mitre.org/techniques/T1496/>.
- [92] CAPEC - CAPEC-605: Cellular Jamming (Version 3.9);. Accessed: 2024-10-04. Available from: <https://capec.mitre.org/data/definitions/605.html>.
- [93] CAPEC - CAPEC-114: Authentication Abuse (Version 3.9);. Accessed: 2024-10-04. Available from: <https://capec.mitre.org/data/definitions/114.html>.
- [94] Credential Access, Tactic TA0031 - Mobile | MITRE ATT&CK®;. Accessed: 2024-10-04. Available from: <https://attack.mitre.org/tactics/TA0031/>.
- [95] Key Concepts — Hyperledger Fabric Docs main documentation;. Accessed: 2025-01-04. Available from: https://hyperledger-fabric.readthedocs.io/en/release-2.5/key_concepts.html.
- [96] Beutel DJ, Topal T, Mathur A, Qiu X, Fernandez-Marques J, Gao Y, et al.. Flower: A Friendly Federated Learning Research Framework. arXiv; 2022. ArXiv:2007.14390 [cs]. Available from: <http://arxiv.org/abs/2007.14390>.
- [97] zkonduit/ezkl. Zkonduit; 2025. Accessed: 2025-01-04. Available from: <https://github.com/zkonduit/ezkl>.
- [98] Albrecht M, Grassi L, Rechberger C, Roy A, Tiessen T. MiMC: Efficient Encryption and Cryptographic Hashing with Minimal Multiplicative Complexity; 2016. Publication info: A minor revision of an IACR publication in ASIACRYPT 2016. Available from: <https://eprint.iacr.org/2016/492>.
- [99] Grassi L, Khovratovich D, Rechberger C, Roy A, Schofnegger M. Poseidon: A New Hash Function for Zero-Knowledge Proof Systems; 2019. Publication info: Published elsewhere. USENIX Security '21. Available from: <https://eprint.iacr.org/2019/458>.

Appendix

I. Glossary

ACL	Access Control List
AI	Artificial Intelligence
BPMN	Business Process Model and Notation
CA	Certificate Authority
CIA	Confidentiality, Integrity, Availability
CNN	Convolutional Neural Network
DDoS	Distributed Denial of Service
DP	Differential Privacy
DSRC	Dedicated Short-Range Communications
FHE	Fully Homomorphic Encryption
FL	Federated Learning
GAN	Generative Adversarial Network
gRPC	Google Remote Procedure Call
GSM	Global System for Mobile Communications
HLF	Hyperledger Fabric
HTTPS	Hypertext Transfer Protocol Secure
IoT	Internet of Things
IoV	Internet of Vehicles
ISSRM	Information System Security Risk Management
ITS	Intelligent Transportation System
LLM	Large Language Model
MEC	Multi-access Edge Computing
MitM	Man-in-the-Middle
MSP	Membership Service Provider
PDC	Private Data Collections
PKI	Public Key Infrastructure
PoS	Proof-of-Stake
PoW	Proof-of-Work
REST	Representational State Transfer
RSU	Road Side Unit
SA	Secure Aggregation
SDK	Software Development Kit
SNARK	Succinct Non-interactive Argument of Knowledge
TEE	Trusted Execution Environment
TLS	Transport Layer Security
UML	Unified Modeling Language
V2I	Vehicle-to-Infrastructure
V2V	Vehicle-to-Vehicle
V2X	Vehicle-to-Everything
VANET	Vehicular Ad-hoc Network
WASM	WebAssembly
ZKP	Zero-Knowledge Proof

II. Extended Risk-Related Concepts

Client-side Threats

Let's start with the clients who hold the most sensitive information, such as raw traffic data. They are responsible for data collection and local model training. The distributed nature of FL allows both honest and malicious clients to participate, each with different intentions. Legitimate vehicles aim to contribute valuable information in exchange for an improved global model (or other incentives). On the other hand, malicious clients may seek to either unfairly benefit from the ITS or disrupt its operation entirely.

Table 7 outlines the key vulnerabilities in the default FL process that dishonest clients can exploit to attack the ITS:

Table 7. Vulnerabilities in FL exploited by authorized clients

Id	Vulnerability	System Asset	Details
V1	Lack of training process monitoring	SA6	Lack of reliability, traceability, and control in training can lead to undetected malicious behaviors, such as local model tampering and poisoning
V2	Lack of training data quality checks	SA6	The absence of verification for data distribution and the validity of local training data, including benchmarks and quality checks, can compromise the integrity of the training process
V3	Poor model validation algorithms	SA7	Weak validation practices may fail to detect malicious models, allowing them to pass into the aggregation stage
V4	Robustness of model aggregation against outliers	SA8	Different aggregation practices offer varying resilience and robustness against malicious local models. The effectiveness of these practices depends on factors such as model architecture, the number of FL participants, and the distribution of data and weights. Strong aggregation practices should mitigate the impact of outliers on the final results

Lack of Training Process Monitoring (V1). In the context of IoV, the decentralized nature of FL makes it difficult to monitor the training process effectively. Verifying the exact algorithms and methods (SA6) used to train local machine learning models can be challenging. This vulnerability opens the door to undetected malicious behaviors, such as local model tampering or poisoning. A compromised vehicle client could manipulate its local model updates without detection, introducing biases or errors into the model aggregation process, which would then impact the system's performance.

Lack of Training Data Quality Checks (V2). The absence of rigorous verification mechanisms (SA6) for data distribution and the quality of local training data presents a major vulnerability in the FL process. While clients' data must remain private, some form of on-site validation is also required. Without such checks, a malicious vehicle could inject anomalous data into its local training set. If this goes undetected, it could distort the global model's understanding of traffic patterns, undermining the reliability of the entire ITS.

Poor Model Validation Algorithms (V3). Since this compromised data indirectly affects model performance after training, identifying the root cause of the poisoning becomes increasingly difficult later in the process. Model validation algorithms (SA7) can vary in complexity, as well as in their ability to detect and mitigate these outliers, making this an ongoing challenge that requires continuous improvement.

Robustness of Model Aggregation Against Outliers (V4). The FL aggregation process (SA8) is particularly vulnerable to outlier contributions, which can disproportionately impact the global model. If vehicles submit extreme or erroneous local models, they can significantly distort the global model's interpretation of traffic patterns, especially if the aggregation methods fail to account for these outliers effectively. The robustness of aggregation techniques varies, depending on factors such as model architecture, the number of participants, and the distribution of data and weights. To ensure reliable results, robust aggregation methods must be implemented on the RSU server side to mitigate the influence of outliers on the final model before it is utilized by the ITS and propagated to other peers.

All the described vulnerabilities can be exploited by malicious vehicle clients, leading to various attack scenarios. Table 8 presents three main attack methods that direct FL participants can perform in the data sharing process: data poisoning attacks, model poisoning attacks, and free rider attacks. First, we will discuss each scenario and then conclude the common impact of the attacks on our system's business assets.

The initial phase of most IoV scenarios follows a common pattern: a vehicle approaches the RSU and establishes communication. The RSU, acting as a server/manager, registers the vehicle as an authorized participant in the FL process. Subsequent actions of the threat agent vary depending on the specific attack method:

Data Poisoning Attacks (SR1). In data poisoning attacks, adversary vehicles manipulate training data to induce undesirable behavior in the trained and subsequently aggregated models. The attacker can be just a dishonest user or a compromised one (ST1 Identity Spoofing, ET2 Client Software Exploitation, ET3 Authentication Bypass). These attacks (TT1 Local Data Manipulation) exploit the lack of training data validation (V2).

Without proper on-site data quality checks and actions logging (RT2 System Log Falsification), a client can poison the training outcome even while following the legitimate training process protocol (V1). The vehicle then shares the poisoned model, which may bypass (RT1 Participation Denial) poor validation practices (V3) and be included in the aggregation on the RSU server side. The global model integrity impact of poisoning (TT4 Aggregation Result Manipulation) varies depending on the robustness of the specific model aggregation algorithm against outliers (V4), the model architecture, overall data distribution, and specific ML task constraints. A larger number of attackers (ST2 Sybil Attacks) makes detection and filtering more challenging, potentially leading to more significant collective damage (DT2 Resource Exhaustion). The full data poisoning attack scenario is depicted in Figure 23.

Table 8. Security risks in FL caused by authorized clients

Security Risk	Threat agent	Attack method	Threat	Vulnerability	Impact
SR1	An authorized participant of FL process (vehicle client)	Data poisoning attacks	An attacker inserts poisoned data samples into a training dataset and submits this compromised local model for aggregation (ST1, ST2, TT1, RT1, DT2, ET2, ET3)	V2, V3, V4	Integrity: D1, D3, D4 Availability: D1, D3
SR2	An authorized participant of FL process (vehicle client)	Model poisoning attacks	An attacker directly manipulates the model's parameters or updates, causing it to behave undesirably. This manipulated model can then poison the global model during aggregation (ST1, ST2, TT2, RT1, RT2, DT2, ET2, ET3)	V1, V3, V4	Integrity: D1, D3, D4 Availability: D1, D3
SR3	An authorized participant in the FL process (vehicle client)	Free rider attacks	The attacker generates fake model parameters or updates to mimic legitimate users. While the free rider benefits from access to the FL process, their participation negatively impacts the system (ST1, ST2, TT2, RT1, RT2, DT2, ET2, ET3)	V1, V2, V3, V4	Integrity: D1, D3, D4 Availability: D1, D3

Model Poisoning Attacks (SR2). Model poisoning attacks exhibit greater variability in execution and goals compared to data poisoning. The attacker can also be either a dishonest participant or a compromised one (ST1 Identity Spoofing, ET2 Client Software Exploitation, ET3 Authentication Bypass). Malicious vehicles can train faulty models or carefully craft model updates before the actual FL process. They may then either submit entirely fake models or fine-tune them on a previously prepared setup (TT2 V2I Message Tampering). If the FL process lacks a strict training protocol or fails to monitor its execution (V1), the poisonous model can be included in the aggregation (TT4 Aggregation Result Manipulation). It is especially challenging to monitor vehicles' actions because they act as independent and isolated entities that value their privacy (RT1 Participation Denial and RT2 System Log Falsification). Model poisoning subsequently exploits the same server-side vulnerabilities as data poisonings, such as poor model validation (V3) and weak robustness of model aggregation against outliers (V4). In addition, the increased number of malicious actors (ST2 Sybil Attacks) can negatively influence the availability of resources in the system (DT2 Resource Exhaustion). The model poisoning scenario is depicted in Figure 24.

Free Rider Attacks (SR3). Free rider attacks involve malicious clients attempting to benefit from the collaborative learning process without contributing valid data or computational resources. The attacker can also be either a dishonest participant or a compromised one (ST1 Identity Spoofing, ET2 Client Software Exploitation, ET3 Authentication Bypass). During FL training, free riders generate fake model updates to mimic legitimate contributors. In the absence of proper model training protocols (V1), data quality control (V2), or model validation (V3), threat actors can abuse access to the global model for extended periods (RT1 Participation Denial and RT2 System Log Falsification). Free riders typically contribute either completely random or averaged updates from different FL rounds (TT2 V2I Message Tampering). Their primary goal is not to dramatically influence the final model accuracy, but the impact (TT4 Aggregation Result Manipulation) still depends on the aggregation robustness (V4). The free rider attack scenario can be seen as a variation of model poisoning attacks, but it typically has a lesser overall impact. However, the excessive number of free-riders (ST2 Sybil Attacks) can noticeably slow down the system (DT2 Resource Exhaustion).

A successful poisoning attack from a client primarily violates the integrity of ITS assets. During the first FL round, only the global model's integrity (D1) is compromised. However, after consecutive iterations, the poisoned global model (D1*) serves as a basis for training other local models (D3). This poisoning can have a cumulative effect, leading to unreliable predictions (D4) for the infected local models. So, the whole ITS can malfunction and make poor decisions based on these predictions. For example, unreliable predictions of vehicle routes can cause ineffective traffic signal control, leading

to congestion or false alerts about traffic situations. Moreover, if the number of attackers is significant, legitimate users end up participating in meaningless data sharing, wasting time and computational resources. This can be considered a violation of both global (D1) and local (D3) models' availability. So, the ITS may experience delays in the data sharing process, which can be critical for some emergency situations on the roads.

Man-in-the-Middle Threats

Finally, we will explore the vulnerabilities and security risks posed by malicious unauthorized actors. In addition to the previously mentioned client and server-side vulnerabilities, the unauthorized actors exploit the following vulnerabilities to break into the system (check Table 9):

Table 9. Vulnerabilities in FL Man-in-the-Middle scenarios

Id	Vulnerability	System Asset	Details
V11	Lack of encryption	SA1	Weak or absence of encryption can expose sensitive information in the network
V12	Weak identity and access management	SA2	Weak authentication and authorization mechanisms allow unauthorized actors to access data not intended for them

Lack of Encryption (V11). The communication layer (SA1) should employ strong encryption to protect sensitive data transmitted between participants in the FL process. Without robust encryption, unauthorized actors can potentially intercept and access confidential information, including model updates, training data, or system commands. This vulnerability exposes the entire FL system to eavesdropping attacks, potentially compromising the privacy and integrity of the learning process.

Weak Identity and Access Management (V12). The authentication and authorization layer (SA2) must implement robust mechanisms to verify the identity of participants and control their access to system resources. Weak identity and access management can allow unauthorized actors to impersonate legitimate participants or gain access to restricted parts of the system. This vulnerability can lead to various attacks, including data theft, model manipulation, or disruption of the FL process. Proper identity verification and access control are crucial for maintaining the security and trustworthiness of the entire FL ecosystem in the IoV context.

All the described vulnerabilities can be exploited by malicious actors in the middle of communication between vehicles and servers, leading to various attack scenarios. Table

10 presents two main attack methods that unauthorized actors can perform in the FL process: model poisoning attacks and reverse engineering inference attacks.

In a man-in-the-middle scenario, the attackers do not have to be officially part of the FL process. They abuse vulnerabilities both on the client side and on the server side to perform attacks indirectly during V2I communication. The MitM attack scenarios are depicted in Figure 25.

Communication and Model Poisoning Attacks (SR7). By bypassing message encryption (V11) and identity control (V12), the Man-in-the-Middle can capture transmitted data in V2I communication (ST1 Identity Spoofing, ET2 Client Software Exploitation, ET3 Authentication Bypass), violating the confidentiality of global (D1) and local models (D3) (IT1 User Traffic and Identity Disclosure and IT2 Model Update Eavesdropping). The adversary can train faulty models or carefully craft model updates before the actual FL process, as in SR2. Next, without proper data authenticity verification (V8), the attacker may tamper with the transmitted data and add the poisonous model to the aggregation (TT2 V2I Message Tampering). In case of poor model validation (V3) and weak robustness of model aggregation against outliers (V4), the integrity of the global model (D1) and subsequently of related local models (D3) and their predictions (D4) will be violated (TT4 Aggregation Result Manipulation). Without proper logging and action accountability, it can be challenging to identify the root cause of the attack (RT1 Participation Denial and RT2 System Log Falsification). In addition, with a large number of attackers (ST2 Sybil Attacks), the entire system may experience the violation of availability of global (D1) and local models (D3) due to DoS attacks (DT1 Message Flooding, DT2 Resource Exhaustion, DT3 MitM, DT4 Infrastructure Node Attacks).

The impact of such attacks primarily undermines the integrity of ITS assets. The global model (D1) would be compromised first, and after several FL rounds, local models (D3) would become distorted or poisoned, resulting in unreliable predictions (D4). Furthermore, the availability of the global (D1) and local models (D3) could be affected due to disrupted V2I communication of ITS infrastructure.

Communication, Model Inversion, and Membership Inference Attacks (SR8). In this scenario, a curious unauthorized actor exploits vulnerabilities in the communication process to capture sensitive data and attempt to reverse-engineer model parameters, aiming to reconstruct private information from FL participants. This attack leverages multiple vulnerabilities, including insufficient privacy protection mechanisms for local models (V10), weak encryption (V11), and inadequate identity and access management (V12) (ET2 Client Software Exploitation, ET3 Authentication Bypass). The sophistication of this attack lies in its ability to exploit the FL communication process without needing direct access to the server or clients. For instance, an attacker could intercept V2I communications, capturing model updates from various vehicles across multiple FL

rounds (IT2 Model Update eavesdropping). By analyzing these intercepted updates and applying various model inversion techniques, the attacker can potentially reconstruct sensitive training data (IT4 Inference Attacks). Furthermore, the attacker's ability to observe multiple versions of model updates enhances their capability to perform membership inference attacks, potentially identifying specific training samples used by individual vehicles (IT1 User Traffic and Identity Disclosure).

The primary impact of these attacks is on the confidentiality of ITS assets, with the global model (D1), raw traffic data (D2), and local models (D3) all at risk of exposure. This level of insight could reveal detailed information about individual vehicles' movements, preferences, or behaviors, severely compromising user privacy without direct access to the central server or individual clients.

Table 10. Security risks in FL Man-in-the-Middle scenarios

Security Risk	Threat agent	Attack method	Threat	Vulnerability	Impact
SR7	Unauthorised actor	Communication and model poisoning attacks	An unauthorized (man in the middle) can capture sensitive data during FL communication, tamper with and distribute fake ML models (local and global) or FL process instructions. It can also delay responses, slowing down the FL process (ST1, ST2, TT2, TT4, RT1, RT2, IT1, IT2, DT1, DT2, DT3, DT4, ET2, ET3)	V3, V4, V8, V11, V12	Confidentiality: D1, D3 Integrity: D1, D3, D4 Availability: D1, D3
SR8	Unauthorised actor and curious actor	Communication, model inversion and membership inference attacks	A curious unauthorized actor can capture sensitive data during FL communication and reverse-engineer model parameters to reconstruct sensitive data from FL participants (IT1, IT2, IT4, ET2, ET3)	V10, V11, V12	Confidentiality: D1, D2, D3

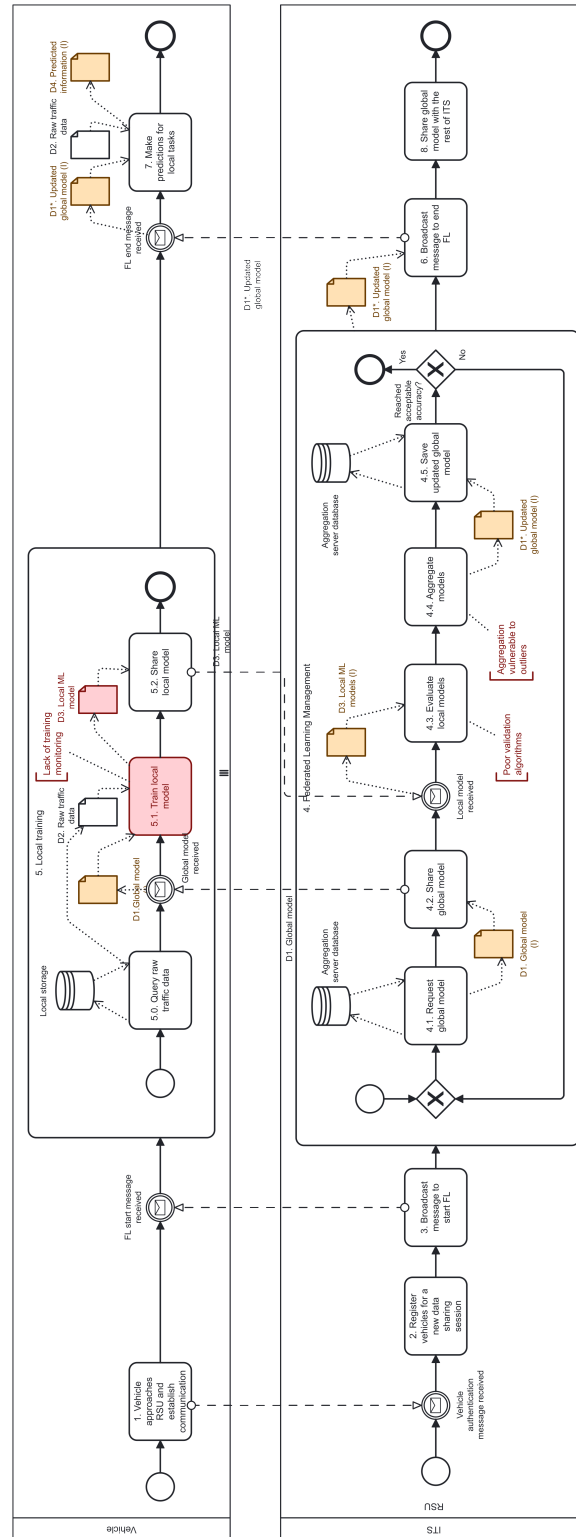


Figure 24. BPMN diagram of model poisoning attacks execution

III. Security Requirements

Security Threat -> Related Risks	
ID	Requirement
<i>ST1. Identity Spoofing -> SR1-SR7</i>	
ST1.R1	The system shall verify the identity of all participants before allowing FL process participation.
ST1.R2	The system shall require multi-factor authentication for critical FL operations.
ST1.R3	The system shall store participant credentials in a tamper-resistant manner.
ST1.R4	The system shall use encrypted protocols for credential management in FL participation.
ST1.R5	The system shall enforce a user management policy with regular credential updates.
ST1.R6	The system shall record all authentication attempts and participant activities in immutable logs.
ST1.R7	The system shall detect anomalies in participant behavior and model updates.
ST1.R8	The system shall verify the authenticity and integrity of participating devices.
ST1.R9	The system shall implement an access control mechanism.
<i>ST2. Sybil Attacks -> SR1, SR2, SR3, SR7</i>	
ST2.R1	The system shall track participant reliability and contributions over time.
ST2.R2	The system shall limit resource allocation to mitigate Sybil attack impacts.
ST2.R3	The system shall limit the number of identities associated with a single authenticated entity.
ST2.R4	The system shall apply constraints based on geographical or network location for identity creation.
ST2.R5	The system shall implement a reputation mechanism for participants.
<i>TT1. Local Data Manipulation -> SR1</i>	
TT1.R1	The system shall validate the integrity of local training data before model training.
TT1.R2	The system shall implement mechanisms to detect anomalies in local model updates.
TT1.R3	The system shall enforce a strict training protocol for all participants.
TT1.R4	The system shall implement robust training process monitoring mechanisms.
<i>TT2. V2I Message Tampering -> SR2, SR3, SR4, SR5, SR7</i>	
TT2.R1	The system shall use secure communication protocols for transmitting model updates.
TT2.R2	The system shall implement message integrity checks for all transmitted data.
TT2.R3	The system shall authenticate the source of all model updates and coordination messages.
<i>TT3. Global Model Modification -> SR4, SR5</i>	
TT3.R1	The system shall implement access controls to restrict modification of the global model.
TT3.R2	The system shall maintain an auditable log of all changes to the global model.
TT3.R3	The system shall use integrity verification mechanisms for the global model storage.
<i>TT4. Aggregation Result Manipulation -> SR1, SR2, SR3, SR7</i>	
TT4.R1	The system shall implement a verifiable aggregation process for model updates.
TT4.R2	The system shall use robust aggregation algorithms that are resistant to outlier influences.
TT4.R3	The system shall validate the integrity of aggregation results before distribution.

<i>RT1. Participation Denial -> SR1, SR2, SR3, SR7</i>	
RT1.R1	The system shall maintain cryptographically signed logs of all participant activities in the FL process.
RT1.R2	The system shall implement non-repudiation mechanisms for all critical FL operations.
<i>RT2. System Log Falsification -> SR2, SR3, SR4, SR5, SR7</i>	
RT2.R1	The system shall use tamper-evident logging mechanisms for all system logs.
<i>RT3. Model Distribution Repudiation -> SR4, SR5</i>	
RT3.R1	The system shall maintain a verifiable record of all model distributions to other FL participants.
RT3.R2	The system shall maintain a verifiable record of all global model updates to the shared storage.
<i>IT1. User Traffic and Identity Disclosure -> SR6, SR7, SR8</i>	
IT1.R1	The system shall encrypt all user traffic in the FL network.
IT1.R2	The system shall implement strong identity protection mechanisms for all participants.
<i>IT2. Model Update Eavesdropping -> SR6, SR7, SR8</i>	
IT2.R1	The system shall use secure channels for transmitting model updates.
IT2.R2	The system shall implement traffic analysis prevention techniques.
<i>IT3. Centralized Collection of Data and Events -> SR6</i>	
IT3.R1	The system shall minimize centralized data collection in the FL process.
IT3.R2	The system shall implement strict access controls for any centrally collected data.
<i>IT4. Inference Attacks -> SR6, SR8</i>	
IT4.R1	The system shall protect trained models from reverse engineering attacks.
IT4.R2	The system shall implement differential privacy techniques in the FL process.
IT4.R3	The system shall limit the number of model versions accessible to any single entity to reduce the risk of temporal analysis attacks.
<i>DT1. Message Flooding -> SR7</i>	
DT1.R1	The system shall implement rate limiting for all types of messages in the FL network.
DT1.R2	The system shall have mechanisms to detect and mitigate abnormal message patterns.
<i>DT2. Resource Exhaustion -> SR1, SR2, SR3, SR4, SR7</i>	
DT2.R1	The system shall implement resource allocation limits for individual participants.
DT2.R2	The system shall monitor and manage computational resources to prevent exhaustion.
<i>DT3. Man-in-the-Middle (MitM) -> SR4, SR7</i>	
DT3.R1	The system shall implement robust authentication and encryption to prevent MitM attacks.
DT3.R2	The system shall verify the authenticity of communication endpoints.
<i>DT4. Infrastructure Node Attacks -> SR4, SR7</i>	
DT4.R1	The system shall implement redundancy and failover mechanisms for critical infrastructure components.
DT4.R2	The system shall have mechanisms to detect and mitigate attacks on infrastructure nodes.
<i>ET1. Server Unauthorized Access -> SR4, SR5, SR6</i>	
ET1.R1	The system shall implement the principle of least privilege for all system components.
ET1.R2	The system shall use multi-factor authentication for accessing critical server resources.
<i>ET2. Client Software Exploitation -> SR1-SR8</i>	
ET2.R1	The system shall implement secure coding practices and regular vulnerability assessments for all client software.
ET2.R2	The system shall have a robust patch management process for all software used in the FL process.
<i>ET3. Authentication Bypass -> SR1-SR8</i>	
ET3.R1	The system shall implement strong and multi-layered authentication mechanisms.
ET3.R2	The system shall monitor and alert on unusual authentication patterns or attempts.

IV. Security Controls

Req. ID	Control Components	Status
Status: + = Fully Implemented, +- = Partially Implemented, - = Not Implemented		
<i>Identity and Authentication Controls</i>		
ST1.R1	X.509 certificate-based identity verification; HLF MSP implementation	+
ST1.R2	Certificate-based authentication with additional identity checks within smart contracts	+
ST1.R3	Basic storage of X.509 credentials; MSP certificate management	+-
ST1.R4	TLS encryption for credential transmission; Certificate-based authentication	+
ST1.R5	Basic credential management without automated rotation	+-
ST1.R6	Limited number of actions are recorded to the ledger	+-
ST1.R7	Transaction monitoring with anomaly detection	-
ST1.R8	X.509 certificates and MSP policies for device authentication and verification	+
ST1.R9	Role-based access control through MSP; Organization-level policies	+
<i>Sybil Attack Prevention</i>		
ST2.R1	FL participants metrics logging	+
ST2.R2	Resource allocation limits based on reputation	-
ST2.R3	Basic identity verification without strict limits	+-
ST2.R4	Channel policies set limits on identities of HLF organizations	+
ST2.R5	Long-term FL participants reputation	-
<i>Data Integrity Controls</i>		
TT1.R1	Data integrity checks through chaincode and ZKP	+-
TT1.R2	Model updates analysis and anomaly detection	-
TT1.R3	Enforcement of training protocol through ZKP or TEE	-
TT1.R4	Blockchain-based logging of training events	+-
<i>Communication Security</i>		
TT2.R1	Flower gRPC with TLS; HLF native TLS	+
TT2.R2	Hash verification for PDC data; Smart contracts integrity checks	+
TT2.R3	Digital signatures verification; Hash verification via smart contracts	+
<i>Model Management</i>		
TT3.R1	Smart contract access control; PDC for limited number of users	+
TT3.R2	Immutable audit log on the ledger	+
TT3.R3	Merkle tree verification; Hash integrity checks via smart contracts	+
<i>Aggregation Security</i>		
TT4.R1	Off-chain ZKP verification of aggregation	+-
TT4.R2	Advanced aggregation strategies	-
TT4.R3	Multi-peer validation; Endorsement policy enforcement	+-
<i>Non-repudiation and Logging</i>		
RT1.R1	Blockchain-based immutable activity logs; Digital signatures for all participant activities	+-
RT1.R2	Smart contract-based non-repudiation; Timestamping of operations	+
RT2.R1	Verifiable executed action and system logs via TEE	-

<i>Privacy Protection</i>		
IT1.R1	Flower gRPC with TLS; HLF native TLS	+
IT1.R2	Basic identity protection through MSP	+/-
IT2.R1	TLS-secured channels; PDC storage with limited access	+
IT2.R2	Weights masking during aggregation; Differential privacy	-
IT3.R1	Decentralized storage using HLF channels and PDC	+
IT3.R2	Channel policies and ACL; PDC access control	+
IT4.R1	Weights masking during aggregation	-
IT4.R2	Differential privacy to protect model weights	-
IT4.R3	Limits on the number of model versions available via smart contracts	+/-
<i>Availability Protection</i>		
DT1.R1	Rate limiting for messages via smart contracts	-
DT1.R2	Anomaly detection of message traffic	-
DT2.R1	Network resources controlled by HLF policies	+/-
DT2.R2	Resource monitoring of blockchain activity	-
DT3.R1	Flower gRPC with TLS; HLF native TLS and MSP	+
DT3.R2	MSP, digital signatures and hash verification	+
DT4.R1	Blockchain-based distributed architecture	+
DT4.R2	Actions verification via smart contracts and ZKP; Endorsement policies	+/-
<i>Privilege Management</i>		
ET1.R1	Channel policies and ACL; PDC access control	+
ET1.R2	Certificate-based authentication with additional identity checks within smart contracts	+
ET2.R1	Packages vulnerability scanning via CI/CD	-
ET2.R2	Chaincode lifecycle management	+/-
ET3.R1	Certificate-based authentication with additional identity checks within smart contracts	+
ET3.R2	Immutable ledger tracking of authentication attempts	+/-

V. Licence

Non-exclusive licence to reproduce thesis and make thesis public

I, **Mykyta Luzan**,

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,

Secure Data Sharing in the Internet of Vehicles Using Blockchain-based Federated Learning,

supervised by Dr. Mubashar Iqbal and Dr. Raimundas Matulevičius.

2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.
3. I am aware of the fact that the author retains the rights specified in p. 1 and 2.
4. I certify that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Mykyta Luzan

05/01/2025