Secure IoT Medical Data Transmission Using Elliptic Curve Cryptography and Video Steganography with Firefly Optimization

Software Requirements Specification Report

submitted by

ALAN SHAIJU KURIAN

Reg. No. MAC21CD003

ANNA ANN MATHEW

Reg. No. MAC21CS012

KEVIN PAULSON

Reg. No. MAC21CD031

KIRAN PAULSON

Reg. No. MAC21CD032

to

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

in partial fulfilment of the requirements for the award of the Degree

of

BACHELOR OF TECHNOLOGY (HONOURS)

in

COMPUTER SCIENCE AND ENGINEERING

Department of Computer Science & Engineering

Mar Athanasius College of Engineering (Autonomous)

Kothamangalam, Kerala, India 686 666

MARCH 2025

Secure IoT Medical Data Transmission Using Elliptic Curve Cryptography and Video Steganography with Firefly Optimization

Software Requirements Specification Report

submitted by

ALAN SHAIJU KURIAN

Reg. No. MAC21CD003

ANNA ANN MATHEW

Reg. No. MAC21CS012

KEVIN PAULSON

Reg. No. MAC21CD031

KIRAN PAULSON

Reg. No. MAC21CD032

to

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

in partial fulfilment of the requirements for the award of the Degree

of

BACHELOR OF TECHNOLOGY (HONOURS)

in

COMPUTER SCIENCE AND ENGINEERING

Department of Computer Science & Engineering

Mar Athanasius College of Engineering (Autonomous)

Kothamangalam, Kerala, India 686 666

MARCH 2025

DECLARATION

We, the undersigned, hereby declare that the project report ”Secure IoT Medical Data Transmission Using Elliptic Curve Cryptography and Video Steganography with Firefly Optimization”, submitted for partial fulfillment of the requirements for the award of degree of Bachelor of Technology (Honours) of the APJ Abdul Kalam Technological University, Kerala, is a bonafide work done by us under the supervision of Prof. Aswathy M V and Prof. Shabiya M I. This submission represents our ideas in our own words and where ideas or words of others have been included, we have adequately and accurately cited and referenced the original sources. We also declare that we have adhered to the ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in our submission. We understand that any violation of the above will be a cause for disciplinary action by the institute and/or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not previously formed the basis for the award of any degree, diploma, or similar title of any other University.

Place: Kothamangalam

Date:

Anna Ann Mathew (MAC21CS012)
Alan Shaiju Kurian (MAC21CD003)
Kevin Paulson (MAC21CD031)
Kiran Paulson (MAC21CD032)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

MAR ATHANASIUS COLLEGE OF ENGINEERING

(AUTONOMOUS)

KOTHAMANGALAM

CERTIFICATE

This is to certify that the report entitled **Secure IoT Medical Data Transmission Using Elliptic Curve Cryptography and Video Steganography with Firefly Optimization** *submitted by* **Mr. ALAN SHAIJU KURIAN (MAC21CD003), Ms. ANNA ANN MATHEW (MAC21CS012), Mr. KEVIN PAULSON (MAC21CD031), Mr. KIRAN PAULSON (MAC21CD032)** *towards partial fulfillment of the requirement for the award of Degree of Bachelor of Technology in Computer Science and Engineering (Honours) from APJ Abdul Kalam Technological University for March 2025 is a bonafide record of the project carried out by them under our supervision and guidance.*

Prof. Aswathy M V	**Prof. Shabiya M I**	**Prof. Joby George**
Project Guide	Project Coordinator	Head of the Department

Internal Examiner(s) External Examiner(s)

Date: Dept. Seal

ACKNOWLEDGEMENT

First and foremost, we sincerely thank the 'God Almighty' for his grace for the successful and timely completion of the project.

We express our sincere gratitude and thanks to ***Dr. Bos Mathew Jos****, Principal and* ***Prof. Joby George****, Head of the Department for providing the necessary facilities and their encouragement and support.*

We owe special thanks to our project guide ***Prof. Aswathy M V*** *and project coordinator* ***Prof. Shabiya M I*** *for their corrections, suggestions and sincere efforts to co-ordinate the project under a tight schedule.*

We express our sincere thanks to staff members in the Department of Computer Science and Engineering who have taken sincere efforts in guiding and correcting us in conducting this project.

Finally, we would like to acknowledge the heartfelt efforts, comments, criticisms, co-operation and tremendous support given to us by our dear friends during the preparation of the project and also during the presentation without which this work would have been all the more difficult to accomplish.

Table of Contents

List of Abbreviations

ECC	Elliptic Curve Cryptography
IoT	Internet of Things
PSNR	Peak Signal-to-Noise Ratio
FPS	Frames Per Second
MP4V	MPEG-4 Video Codec

Chapter 1
Introduction

1.1 Purpose

The purpose of this project is to tackle the critical challenge of securing sensitive medical data in Internet of Things (IoT) healthcare systems, where patient privacy and data integrity are paramount. By integrating advanced cryptographic and steganographic techniques—specifically Elliptic Curve Cryptography (ECC) with the secp256r1 curve, video steganography using a Matrix XOR method, and the Firefly Optimization Algorithm—the project aims to develop a robust system for the secure and covert transmission of medical information over vulnerable IoT networks.

The project seeks to demonstrate the power of ECC in providing lightweight, strong encryption suitable for resource-constrained IoT devices, ensuring confidentiality of patient vitals such as heart rates, blood pressure, and temperatures. Coupled with this, video steganography embeds the encrypted data into MP4 video frames with minimal visual distortion, leveraging the Firefly Optimization Algorithm to select optimal embedding locations for maximum stealth. This dual-layered approach—encryption for security and steganography for concealment—offers a cutting-edge solution tailored to IoT healthcare needs.

Furthermore, the project aims to showcase how these advanced techniques can enhance traditional data protection methods in IoT environments. By encrypting a synthetic medical dataset, verifying its integrity, and hiding it within video streams, the system achieves high security, imperceptibility (measured via Peak Signal-to-Noise Ratio, PSNR), and efficiency compared to standalone encryption approaches.

Overall, the purpose of this project is to contribute to patient privacy and data protection in IoT healthcare by providing a novel, scalable solution for secure data transmission. Through the development of this system, we aim to address the urgent need for discreet and reliable medical data management, empowering healthcare providers and IoT developers to safeguard sensitive information in real-world scenarios like telemedicine or remote monitoring.

1.2 Document Conventions

1.3 Intended Audience and Reading Suggestions

This Software Requirements Specification (SRS) is intended for the following audiences:

- The team members including project guide, coders, developers, testers, documentation writers—Professors of the college
- Users of the project
- Maintenance Engineers

1.4 Product Scope

The scope of this project is to develop an efficient system for secure IoT medical data transmission using Elliptic Curve Cryptography (ECC), video steganography, and the Firefly Optimization Algorithm. The project addresses the challenge of protecting patient data in IoT healthcare by encrypting it with ECC and concealing it within video files, ensuring both confidentiality and covertness across diverse transmission scenarios.

The project utilizes a synthetic dataset of 100 medical records, stored in `medical_data.csv`, comprising patient IDs, timestamps (starting March 10, 2025), heart rates (60–100 BPM), blood pressure (100–140/60–90 mmHg), and temperatures (36.0–37.5°C). This dataset mimics outputs from IoT healthcare devices like wearables or sensors, curated to test the system's encryption and embedding capabilities. Additionally, five MP4 videos (`video1.mp4` to `video5.mp4`) serve as carriers, sourced from the `videos` subdirectory of the `iot_security_dataset` structure.

To prepare the data, ECC encryption with the secp256r1 curve transforms each record into 192-bit triplets (`C1.x`, `C1.y`, `C2`), stored in `encrypted_medical_data.csv`. A verification step ensures the integrity of videos and datasets, checking frame counts, FPS, resolutions, and record counts. The steganography process embeds the 19,200-bit payload into video frames (10 bits per frame across 1,920 frames total), guided by the Firefly Algorithm to select bright 8x8 blocks, and outputs stego-videos (`stego_video1.mp4` to `stego_video5.mp4`) in the `stego_videos` subdirectory.

The system leverages three Python scripts: `encrypt_data.py` for ECC encryption, `finalize_dataset.py` for dataset verification, and `implement_model.py` for steganography and PSNR evaluation. The Firefly Algorithm optimizes embedding locations, ensuring minimal visual impact (targeting PSNR above 40 dB), while the Matrix XOR method embeds bits efficiently into least significant bits (LSBs) of pixel values.

Performance is evaluated through metrics like encryption speed, embedding capacity (19,200 bits across five videos), and PSNR, demonstrating the system's effectiveness in securing and hiding medical data with imperceptible changes to video quality. The scope focuses on a proof-of-concept, excluding real-time IoT integration or data extraction, but lays a foundation for scalable healthcare applications.

Overall, this project aims to deliver a robust system for secure IoT medical data transmission. By combining ECC, video steganography, and Firefly optimization with a synthetic dataset and video carriers, it offers a practical solution for patient data protection, enhancing privacy and security in IoT healthcare environments like telemedicine or remote patient monitoring.

1.5 References

1.5.1 Websites:

- https://ieeexplore.ieee.org/document/8675777

1.5.2 Reference Books and Articles:

- M. Khari, A. K. Garg, A. H. Gandomi, R. Gupta, R. Patan and B. Balusamy, ”Securing Data in Internet of Things (IoT) Using Cryptography and Steganography Techniques,” in IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 50, no. 1, pp. 73-80, Jan. 2020, doi: 10.1109/TSMC.2019.2903785.

- S. K. Mousavi, A. Ghaffari, S. Besharat, and H. Afshari, ”Security of Internet of Things Based on Cryptographic Algorithms: A Survey,” Wireless Networks, vol. 27, no. 2, pp. 1515–1555, 2021.

- S. Nokhwal, M. Chandrasekharan, and A. Chaudhary, ”Secure Information Embedding in Images with Hybrid Firefly Algorithm,” Neural Computing and Applications, 2024.

- Manal M. Khayyat, Mashael M. Khayyat, S. Abdel-Khalek, and R. F. Mansour, ”Blockchain Enabled Optimal Hopfield Chaotic Neural Network Based Secure Encryption Technique for Industrial Internet of Things Environment,” Alexandria Engineering Journal, vol. 61, pp. 11377–11389, 2022.

- S. Ghosh, A. Saha, T. Pal, and A. K. Jha, ”A Comparative Analysis of Chaos Theory Based Medical Image Steganography to Enhance Data Security,” International Conference on Machine Learning and Data Engineering (ICMLDE 2023), 2023.

Chapter 2
Overall Description

2.1 Product Perspective

The product perspective of this project is to develop a secure, covert system for transmitting sensitive medical data in Internet of Things (IoT) healthcare environments using Elliptic Curve Cryptography (ECC), video steganography, and the Firefly Optimization Algorithm. The system serves as a valuable tool for healthcare providers, IoT developers, and privacy advocates to protect patient vitals—such as heart rates, blood pressure, and temperatures—across vulnerable networks like public Wi-Fi or cellular channels.

The system is designed as a standalone application, capable of processing synthetic medical data and embedding it into MP4 video files sourced from devices like telemedicine cameras or IoT monitoring systems. It leverages ECC with the secp256r1 curve for lightweight, robust encryption, and employs a Matrix XOR steganography method guided by Firefly optimization to conceal the encrypted data within video frames, ensuring both security and imperceptibility.

The system provides a modular pipeline that includes data encryption, dataset verification, and steganographic embedding. It supports loading and processing CSV files (e.g., `medical_data.csv`) and MP4 videos (e.g., `video1.mp4` to `video5.mp4`), with outputs saved as encrypted CSVs (`encrypted_medical_data.csv`) and stego-videos (`stego_video1.mp4` to `stego_video5.mp4`). While lacking a graphical user interface in this proof-of-concept, it is script-driven via Python, with potential for future UI enhancements.

The ECC module encrypts a synthetic dataset of 100 medical records into a 19,200-bit payload, which is then hidden across five 300-frame videos (10 bits per frame) using steganography. The Firefly Algorithm optimizes embedding locations for minimal visual impact, validated by Peak Signal-to-Noise Ratio (PSNR) metrics. Performance is assessed through encryption speed, embedding capacity, and video quality, highlighting the system's potential for IoT healthcare applications like secure telemedicine feeds.

Overall, the system's purpose is to enable secure, discreet transmission of medical data, complementing existing IoT security practices. It aims to preserve patient privacy and data integrity in real-world healthcare scenarios, with a design focused on scalability and adaptability for future integration with IoT workflows and extraction tools.

2.2 Product Functions

The proposed system for secure IoT medical data transmission encompasses several key product functions to achieve its objectives effectively. These functions include:

- Data Loading and Encryption: Capability to load synthetic medical data from CSV files (e.g., `medical_data.csv`) and encrypt it using ECC with the secp256r1 curve. The system generates a private-public key pair and produces encrypted triplets (`C1.x`, `C1.y`, `C2`) stored in `encrypted_medical_data.csv`, ensuring confidentiality with 128-bit security.

- Dataset Verification: The system verifies the integrity of input files—five MP4 videos and two CSV datasets—checking video readability, frame counts, FPS, resolutions, and record counts (100 each). This ensures all assets are valid before embedding, logging results for transparency.

- Steganographic Embedding: The core function is to embed the 19,200-bit encrypted payload into video frames using a Matrix XOR method. The Firefly Optimization Algorithm selects 10 optimal 8x8 blocks per frame based on brightness, embedding one bit per block's least significant bit (LSB), producing stego-videos with the mp4v codec.

- Quality Evaluation: The system calculates PSNR to assess the visual impact of embedding, comparing original and stego-video frames (e.g., targeting 45 dB). This ensures the stego-videos remain imperceptible, suitable for practical use in healthcare settings.
- File Management: The system organizes inputs and outputs within the `iot_security_dataset` directory—raw data in `data`, videos in `videos`, and stego-videos in `stego_videos`—facilitating a structured workflow and traceability.
- Extensibility and Integration: Designed with modularity, the system allows future enhancements like real-time IoT feeds, multi-bit embedding, or integration with decryption modules and healthcare systems for seamless data exchange.
- Security and Privacy: The system prioritizes data security through ECC's cryptographic strength and steganography's covertness. It avoids storing the private key on disk (using `secrets.randbelow`), safeguarding confidentiality during execution.

2.2.1 User Classes and Characteristics

- Healthcare Providers: Require secure, discreet data transmission for patient vitals (e.g., telemedicine feeds).
- IoT Developers: Focus on integrating the system with low-power devices and optimizing performance.
- Privacy Advocates: Emphasize confidentiality and covertness to meet regulations like HIPAA.

2.2.2 Operating Environment

Operating Environment for Secure IoT Medical Data Transmission:

- Python: A versatile language for cryptography and computer vision, supporting libraries like `tinyec`, `secrets`, and `OpenCV` for ECC, random number generation, and video processing.
- OpenCV (cv2): Handles video frame processing, grayscale conversion, and stego-video writing with the mp4v codec, essential for steganography and verification.
- tinyec: Implements ECC with the secp256r1 curve, enabling lightweight encryption suitable for IoT devices.
- Operating Systems: Runs on Windows, Linux, or macOS, with Windows-compatible paths (e.g., `os.path.join`) used in scripts.
- Hardware: Requires moderate CPU power for encryption and embedding; GPU acceleration (e.g., via CUDA) is optional but not utilized here, fitting IoT's resource constraints.
- Development Environment: Scripts are developed and tested in environments like PyCharm or VS Code, supporting Python 3.x with library dependencies installed via pip.

2.3 Design and Implementation Constraints

- Computational Resources: ECC and video processing demand moderate CPU resources; embedding 19,200 bits across 1,920 frames may strain low-end IoT devices without optimization.
- Dataset Size: Limited to 100 synthetic records and five 300-frame videos, constraining scalability testing for larger datasets or real-time feeds.
- Processing Time: Encryption and frame-by-frame embedding may slow execution on resource-constrained devices, impacting real-time applicability.
- Steganography Detectability: Basic LSB embedding could be vulnerable to steganalysis (e.g., histogram shifts), though Firefly's sparse selection mitigates this.

- Video Quality: MP4 compression or dark-heavy footage might disrupt LSB fidelity or reduce usable blocks, affecting capacity or PSNR.
- Key Management: Lack of secure key storage or distribution limits practical deployment beyond this proof-of-concept.

2.4 User Documentation

The user documentation for this system includes:

- Detailed comments within `encrypt_data.py`, `finalize_dataset.py`, and `implement_model.py` explaining functionality and usage.
- Console logs (e.g., ”Encrypted data saved”, ”Wrote 300 frames”) for run-time feedback.

2.5 Assumptions and Dependencies

2.5.1 Assumptions

- Videos have sufficient frames (e.g., 300 each) and bright areas for embedding.
- Synthetic data accurately represents IoT medical outputs.
- Execution environment has required libraries installed (e.g., `tinyec`, `opencv-python`).

2.5.2 Dependencies

- Video quality and frame consistency affect embedding success.
- ECC security relies on the secrecy of the private key, managed ephemerally in memory.
- Python libraries (`tinyec`, `opencv-python`, `numpy`) must be installed and compatible.

Chapter 3
System Features

3.1 System Features I

Data Encryption

The system should utilize Elliptic Curve Cryptography (ECC) with the secp256r1 curve to encrypt sensitive medical data from `medical_data.csv`. It should generate a private-public key pair and produce encrypted triplets (`C1.x`, `C1.y`, `C2`) using a simplified ECC scheme, ensuring confidentiality with 128-bit security suitable for IoT healthcare environments.

3.2 System Features II

Dataset Verification

The system should have the capability to verify the integrity of input files, including five MP4 videos (`video1.mp4` to `video5.mp4`) and two CSV datasets (`medical_data.csv`, `encrypted_medical_data.csv`). It should check video readability, extract metadata (frame count, FPS, resolution), and count records (100 each), ensuring all assets are valid before steganographic embedding.

3.3 System Features III

Steganographic Embedding

The system should provide efficient embedding of the 19,200-bit encrypted payload into video frames using a Matrix XOR method. Guided by the Firefly

Optimization Algorithm, it should select 10 optimal 8x8 blocks per frame based on brightness, embedding one bit per block's least significant bit (LSB), and output stego-videos (`stego_video1.mp4` to `stego_video5.mp4`) with minimal visual distortion.

3.4 System Features IV

Quality Assessment

The system should support evaluation of stego-video quality using Peak Signal-to-Noise Ratio (PSNR). It should compare original and stego-video frames (e.g., `video1.mp4` vs. `stego_video1.mp4`), targeting a PSNR above 40 dB to ensure imperceptibility, facilitating practical use in healthcare applications like telemedicine.

3.5 System Features V

File Management

The system should organize inputs and outputs within the `iot_security_dataset` directory structure. It should handle raw data in `data`, original videos in `videos`, and stego-videos in `stego_videos`, providing a modular and traceable workflow for secure data transmission.

3.6 System Features VI

Processing Efficiency

The system should be optimized for efficient processing on resource-constrained IoT devices. It should encrypt 100 records, verify datasets, and embed 19,200 bits across 1,920 frames (five 300-frame videos) swiftly, leveraging lightweight ECC and sparse embedding (10 bits per frame) to minimize computational overhead.

3.7 System Features VII

Scalability and Extensibility

The system should be designed with scalability and extensibility in mind, allowing for future enhancements such as real-time IoT integration, multi-bit embedding per frame, or extraction/decryption modules. It should support potential integration with healthcare systems for seamless data exchange and collaboration.

3.8 System Features VIII

Security and Covertness

To enhance security and prevent detection, the system should combine ECC's cryptographic strength with steganography's covertness. It should distribute the payload across five videos, requiring all for full reconstruction, and use Firefly's brightness-based block selection to reduce statistical detectability, ensuring patient data remains both secure and hidden.

Chapter 4
Other Non-Functional Requirements

4.1 Performance Requirements

- Processing Efficiency: The system should encrypt medical data, verify datasets, and embed the 19,200-bit payload into five MP4 videos efficiently, delivering fast results suitable for IoT devices with limited computational resources.
- Embedding Speed: The system should provide rapid steganographic embedding capabilities, processing 10 bits per frame across 1,920 frames (five 300-frame videos) swiftly, enabling timely data transmission in healthcare scenarios.
- Scalability: The system should be scalable to handle larger datasets (beyond 100 records) and additional videos, maintaining performance and video quality as the volume of medical data or video carriers increases.
- Reliability: The system should exhibit high reliability in encrypting, verifying, and embedding data, ensuring consistent performance across varying video resolutions, frame rates, and data sizes without errors or loss.
- Security: The system should prioritize data security and confidentiality, leveraging ECC's 128-bit strength and steganography's covertness to protect against unauthorized access, with the private key managed ephemerally

in memory.

- Usability: The system should feature clear script execution and console logging (e.g., "Encrypted data saved", "Wrote 300 frames"), providing straightforward feedback for developers and testers during operation.
- Maintenance: The system should be implemented in a modular manner with comprehensive code comments, facilitating future updates such as real-time integration or enhanced embedding techniques.

4.2 Safety Requirements

- Data Privacy: The system should prioritize the privacy and confidentiality of medical data, implementing ECC encryption to safeguard patient vitals and adhering to healthcare regulations like HIPAA. Secure key generation via `secrets.randbelow` ensures no sensitive data is exposed on disk.
- Ethical Considerations: The project should adhere to ethical guidelines, using synthetic data to avoid real patient information risks, ensuring respect for privacy and well-being in IoT healthcare applications.
- System Stability: The system should be designed to ensure stability, minimizing risks of crashes or data corruption during encryption, verification, or embedding. Rigorous testing of scripts should identify and mitigate potential failures.
- Documentation and Training: Clear documentation within `encrypt_data.py`, `finalize_dataset.py`, and `implement_model.py` should be provided, explaining functionality and usage to developers and healthcare IT staff, reducing misuse risks.
- Emergency Protocols: The system should include fallback procedures (e.g., logging errors like "Could not open video") to handle unexpected issues such as missing files or corrupted videos, ensuring safe operation and data recovery options.

- Stakeholder Communication: Transparent communication with stakeholders—healthcare providers, IoT developers, and privacy advocates—should be maintained, providing updates on system capabilities and addressing security or safety concerns promptly.

4.3 Software Quality Attributes

- Accuracy: The system should demonstrate high accuracy in encrypting medical data and embedding it into videos. The ECC module should produce consistent triplets (`C1.x`, `C1.y`, `C2`), and steganography should embed bits precisely, verifiable via extraction tests.

- Performance: The system should exhibit strong performance in terms of speed and efficiency. It should encrypt 100 records, verify five videos, and embed 19,200 bits within a reasonable timeframe, optimized for IoT's low-power constraints.

- Reliability: The system should be robust, handling varying video qualities (e.g., 1920x1080, 30 FPS) and dataset sizes without failures. It should consistently produce stego-videos with PSNR above 40 dB, ensuring dependable operation.

- Usability: The system should provide intuitive script-based interaction for developers, with clear console outputs (e.g., "PSNR: 45.2 dB") and structured file management (`data`, `videos`, `stego_videos`) to simplify usage and troubleshooting.

- Scalability: The system should scale to accommodate larger medical datasets or additional videos, maintaining encryption strength and embedding quality as inputs grow, ensuring adaptability for broader IoT healthcare deployments.

Chapter 5
Other Requirements

- Testability: The software should be designed for easy testing to verify its accuracy and performance in encrypting, verifying, and embedding medical data. It should support logging (e.g., console outputs like "Encrypted data saved") and debugging capabilities within `encrypt_data.py`, `finalize_dataset.py`, and `implement_model.py` for effective validation.

- Compliance: The software should comply with relevant healthcare industry standards and privacy regulations (e.g., HIPAA) applicable to IoT medical data transmission. It should adhere to security and ethical guidelines to ensure responsible deployment in healthcare settings.

- System Reliability and Stability: The system should prioritize reliability and stability during operation, with fail-safe mechanisms (e.g., error messages like "Could not open video") to handle file absences or processing interruptions, maintaining data integrity throughout the pipeline.

- Documentation and Traceability: The system should be accompanied by comprehensive documentation within the scripts, including comments on functionality, file structure (`iot_security_dataset`), and workflow steps. This promotes transparency and facilitates issue resolution or future enhancements.

- Integration with Existing Systems: The system should support integration with existing IoT healthcare platforms, such as telemedicine systems or patient monitoring databases. It should enable data exchange (e.g., stego-videos as carriers) and interoperability to streamline secure data transmission processes.

- Security and Privacy: The system should implement strict security measures to protect sensitive medical data, using ECC's 128-bit encryption and steganography's covertness. It should comply with privacy standards, ensuring confidentiality and secure handling of data without storing the private key on disk.
- User-friendly Interface: The system should feature an intuitive script-based interface for developers, allowing easy execution of encryption, verification, and embedding tasks. Console logs (e.g., "PSNR: 45.2 dB") should provide clear feedback to aid in secure data management decisions.
- Error Handling and Alerting: The system should include robust error handling mechanisms to address issues like missing files or video corruption effectively. It should provide timely alerts (e.g., "Error: medical_data.csv is missing") to users, ensuring awareness of potential risks during processing.

Chapter 6
Appendix

6.1 Appendix A: Glossary

- ECC - Elliptic Curve Cryptography
- IoT - Internet of Things
- PSNR - Peak Signal-to-Noise Ratio
- LSB - Least Significant Bit
- MP4 - MPEG-4 Part 14 (video format)

6.2 Appendix B: Dataflow Diagram

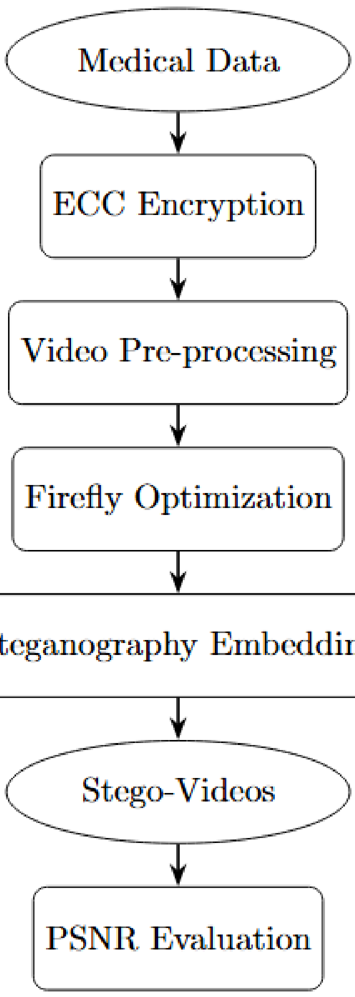

Figure 6.1: Dataflow Diagram for Secure IoT Medical Data Transmission