Electric Power Steering control system

Telmo Sauce
DETI
Universidade de Aveiro
Aveiro, Portugal
telmobelasauce@ua.pt

Ricardo Covelo
DETI
Universidade de Aveiro
Aveiro, Portugal
ricardocovelo11@ua.pt

João Almeida
DETI
Universidade de Aveiro
Aveiro, Portugal
boia13@ua.pt

***Abstract*—In the era of electric power steering (EPS), the reliability of the integral sensors has emerged as a critical concern. This study digs into the pivotal issue of sensor dependability within EPS systems and the needed robustness of the system so that fewer EPS-related accidents happen. The objective is to design and implement a comprehensive monitoring system capable of continuously assessing sensor state and calibration. By focusing on this vital aspect, we aim to mitigate potential safety risks, ensure optimal system performance, and bolster the overall security of EPS-equipped vehicles.**

I. Introduction

This research aims to investigate and address the challenges associated with the reliability, robustness, and security of Electric Power Steering (EPS) systems in modern automobiles.

Our study seeks to provide a solution that offers an innovative perspective on these issues. By exploring various approaches, including advanced encryption protocols and sensor calibration techniques, we intend to enhance the security of EPS systems, making them more resilient to potential cyber threats and external disruptions.

Furthermore, our research is poised to explore profoundly the realm of strategies for the enhancement of reliability and robustness in EPS systems. This endeavor may encompass the meticulous examination of redundancy mechanisms and the seamless integration of real-time diagnostics. Such measures are envisaged to significantly mitigate the likelihood of system failures and ensure the unfaltering performance of EPS systems, even under adverse operating conditions.

In summary, this research acknowledges the importance of addressing these challenges in EPS systems. It aims to explore potential solutions, leaving room for flexibility and adaptability as we progress in our investigations. Ultimately, our goal is to contribute to fortifying these systems, rendering them more impervious to potential cyber threats and extraneous interferences.

II. Secure Life Cycle Description

A. Education and Awareness

This phase will be supervised by Project Manager.

To Begin the development of a secure and reliable product, it is essential to gain a comprehensive understanding of our project's scope and the tools that will be used throughout its life cycle.

In terms of programming languages, we have opted to use C/C++ due to their low-level capabilities and high-speed performance, both crucial for achieving a responsive and high-performance application. To enhance our expertise, we plan to enroll in a three-day intermediate-level course.

Regarding sensors, we'll acquire them from Bosch Mobility and receive training from a Bosch Mobility expert to ensure proper integration into a vehicle. We'll invest at least three days in this training.

Upon obtaining proficiency in both sensor technology and C/C++ programming, we will be well-prepared to initiate the project. However, recognizing the need to further enhance our capabilities, we will schedule a one-day MATLAB course. This decision is motivated by our lack of familiarity with MATLAB, a language that will significantly aid in comprehending the data generated by the sensors and streamline the data visualization process.

Finally, it is crucial to provide training that is more connected to security. In this case, there is a need to provide training on secure practices and code standards in order to raise awareness on potential vulnerabilities.

B. Project Inception

This phase will be supervised by Project Manager and Project Architect.

To start, we must acquire an in-depth grasp of industry standards, including third-party tools used in our field, along with cryptographic protocols such as TLS and SSL. Additionally, we must establish secure response time benchmarks for our application.

We also must identify potential vulnerabilities within our project and detect possible points susceptible to attacks. In addition, we should formulate strategies for the mitigation of such vulnerabilities, to maintain our project's integrity.

C. Analysis and Requirements

This phase will be supervised by Developers and Project Manager and Project Architect.

During this phase, our aim is to methodically formulate, capture, and gain approval for both the analysis and subsequent security requirements. We will begin by comprehensively eliciting system requirements through stakeholder interactions and documentation reviews. This process ensures a deep

understanding of functional, performance, and security needs. To capture these requirements, we will employ a dedicated requirements management tool, such as IBM Engineering Requirements Management DOORS or Atlassian Jira, offering version control, traceability, and collaboration features. The captured requirements, encompassing both functional and security aspects, undergo a formal review and approval process. Stakeholder involvement remains integral throughout, with regular validation sessions ensuring the final requirements accurately reflect collective expectations. This meticulous process establishes a robust foundation for the secure design of the system.

D. Architectural and Detailed Design

This phase will be supervised by the Project Architect.

The primary goal of this phase is to establish a functional and secure design for the system. The initiation involves defining comprehensive requirements to serve as a guiding framework for the entire design process. Subsequently, a meticulous risk analysis will be conducted to identify potential weak points and fundamental design vulnerabilities. The plan explicitly outlines the methodology for executing architecture and design, ensuring a precise mapping of the derived architecture and design elements to the pre-defined security requirements. Specific tools, including Enterprise Architect for UML modeling, ThreatModeler for threat identification and mitigation, and Dependency-Check for managing open-source component vulnerabilities, will be utilized to enhance precision and effectiveness throughout the design phase.

E. Implementation and testing

This phase will be supervised by Testing Team Manager.

As this is a technology responsible for safeguarding multiple lives, our system testing will be rigorous and comprehensive. We will use Klocwork, a SAST tool to analyze the source code and detect vulnerabilities early on in the development phase. We chose Klocwork since we are already familiar with the tool and runs on C/C++ languages that we will use on our project. For a finer analysis we will use the DAST tool "Icamtuf", this tool will automate a big part of pen-testing, using pre-build attacks and detecting vulnerabilities that our system may have. Also, we will sub-contract a specialized penetration testing team to help us detect the more complicated attacks that a user may be able to do. For maximum security, the professionals will use white box testing of the system so that even if the attackers get access to the source code, it's very difficult to bypass the security mechanisms. As we will make the system from scratch minimal use of third-party components will be used, limiting the possible attacks that may occur from poorly written code of those components.

F. Release, deployment, and support

This phase will be supervised by Project Manager.

As usual, some vulnerabilities might be found after the launch of the product. To mitigate the problems that might arise from that we will inform authorized car mechanics that a new update to the system will launch and they should update it on all the cars that use our product. That update will be authenticated and only authenticated updates will be able to be installed on the car. That update will be made by a team that will always be ready to fix a bug that might be discovered. Simulations of a vulnerability found will be made from 6 to 6 months to ensure that the team still remembers how the system works and can quickly patch it. Also, we will alert car inspectors to the fact that an unofficial version may be circulating and to be aware and inspect if the version of the system is an official one. We will also inform the authorities of the possibility of those unofficial units circulating and apprehend those for the safety of the users.

III. Security requirements

Since our system is responsible for ensuring the safety of multiple individuals, multiple security requirements were put in place to guarantee the safety of our users. These security requirements outline the necessary measures, protocols, and guidelines that must be followed to protect assets against unauthorized access, theft, or damage.

A. Availability

- AVA-1 **System Availability** - The system must be available at 99.999% up time.
  - The trust bestowed upon our system by the users is rooted in its ability to detect flaws in the EPS system. Our system needs to be more dependable than the EPS system and its sensors, resulting in a need for a very high up time percentage
- AVA-2 **Power Recovery Time** - The system's recovery time after a power outage must not exceed 0.05 seconds once the power is restored.
- AVA-3 **Information Processing** - The system shall be able to process 10G/s bytes of information.
  - In challenging situations, such as extreme weather conditions, the many sensors in a vehicle can generate an abundance of data. Therefore, our system must possess a superior capacity to manage information to avoid any potential system crashes.

B. Confidentiality

- CONF-1 **Secure Communication** - The system's communication shall use TLS/SSL protocol.
  - The utilization of SSL and TLS ensures the safe transmission of sensitive data, effectively preventing unauthorized interception and eavesdropping. Through these highly sophisticated security protocols, critical information is safeguarded, ensuring its integrity.

- CONF-2 **Data Protection** - Log files must not include passwords used to encrypt data. (**Logging**)
- CONF-3 **Access Control** - The system shall have an (RBAC) Role Base Access Control. (**Authentication**)

C. Integrity

- INT-1 **System Accuracy** - The system must have at least 99.95% accuracy.
- INT-2 **Error Rates** - The system must not have more than 0.1% false positives, and 0.01% false negatives.
  - Considering that a false positive is usually not dangerous, but a false negative could result in deadly outcomes, the failure rate of such cases should align with their respective consequences
- INT-3 **Message Integrity** - The system must use integrity control in the messages sent (MAC).
  - The integration of Message Authentication Codes (MACs) into all internal messages of an electric power steering system holds immense significance in preserving data integrity and deterring unauthorized tampering. MACs help the system validate the genuineness of every internally transmitted message, thus minimizing the possibility of data corruption or unauthorized infiltration.
- INT-4 **Anomaly Detection** - The system must detect an anomaly/badly calibrated sensor in a time frame of 2 seconds. (**Performance**)
- INT-5 **CPU Usage** - Our system utilization shall cause a CPU usage increase at a maximum of 20% of the least powerful sensors currently in use. (**Performance**)
  - As our system is integrated with multiple EPS systems, it becomes difficult to customize it to match a specific system. As a result, it's not advisable to have our system disrupt EPS systems with limited information-handling capabilities.
- INT-6 **System Load** - The EPS system should maintain its optimal performance while using the system. (**Performance**)
- INT-7 **Data Logging** - All the information received by the sensors and the decisions made by the system as well as updates to it, must be logged.(**Logging**)
- INT-8 **Log Management** - If the information is older than 6 months or the logs are over 50GB, 7GB of the oldest logs must be deleted. (**Logging**)
  - To avoid excessive use of the system's data storage capabilities, a data deletion point was implemented.
  - To prevent any impact on performance, a significant quantity of logs are removed from the system. This measure ensures that the deletion process does not occur constantly and overwhelms the system.
- INT-9 **Unusual Data Flagging** - Unusual information (The sensors are not calibrated) must have a special flag, logs with that flag cannot be erased until the user goes to a specialist. (**Logging**)
  - It is crucial to ensure the overall health of the system by removing redundant information. However, it is equally important to refrain from eliminating essential data that could be essential for future problem-solving or developing solutions.

D. User

- USR-1 **Secure Updates** - User shall only download and then install updates from our official server (**Authentication)**

E. Development

- DEV-1 **Updated Version** - The latest version of C (C17) shall be used during development.

IV. Security Test Plan

Ensuring the robust security of our Electric Power Steering Control System is paramount. This section outlines a comprehensive Security Test Plan, encompassing assessments and measures to safeguard against vulnerabilities and potential risks.

A. Penetration testing

1) Strategy: As stated before we will sub-contract a specialized pen-testing team. This team will use existing tools to scan our whole environment and system, ensuring that every attack is well accounted for. Also, the team of specialists will make, hand-made attacks for the most complex parts of our system.

2) Explored vulnerabilities:

- **(DOS) Denial of Service**
  - **Reason** As our system has a high flow of information, possibly the most common and easiest type of attack would be a DOS.
  - **Penetration Testing** Our applications are most vulnerable to this type of attack where the information flow is very big. Attacks related to inputting large amounts of non-genuine data may be successful.
  - **Vulnerable Interfaces** The two interfaces that may be vulnerable to this kind of attack are:
    - **Log local database**, which needs to delete information if the size of the logs exceeds 50Gb, if there is too much information to delete in a very short amount of time could result in the malfunctioning of the system
    - **Communication interface between sensors and system** is also vulnerable to this kind of attack since once again if the system gets overloaded with information might result in the malfunctioning of the software.
- **OS command injection**
  - **Reason** Our system has various ways where code handles input.As there are so many instances a badly sanitized input could mean a command injection made by attackers.

  - **Penetration Testing** Our contracted professionals will test every data entry, trying various ways to input malicious code into our software.
  - **Vulnerable Interfaces** Once again our most vulnerable interfaces are where code handles the data, such as in logging reaction and deletion, and code that handles the messages that come from the EPS system and its sensors.
- **Privilege escalation**
  - **Reason** Due to the need for specialists to communicate with the system, it needs to have the possibility to have a command line interface, this interface might be a way for attackers to escalate privileges and access to information and services they are not supposed to have.
  - **Penetration Testing** Connection to this command line, specialists will try to escalate privileges with different tactics, from buffer overflow to badly configured ACLs.
  - **Vulnerable interfaces** As stated before the command line is the interface where this type of attack will occur and will exploit badly configured security options as well as some poorly written code.

B. Fuzz Testing

Fuzz testing ensures software robustness by systematically identifying unexpected behaviors, crucial for reliability. In automotive applications, especially Electric Power Steering, it's vital for safety and responsiveness across diverse conditions.

1) Strategy: For EPS control system testing, a mix of open-source (AFL) and proprietary tools will be used. AFL, known for transparency and community support, is combined with a tailored proprietary fuzzing tool, leveraging AFL's adaptability and the proprietary tool's specialized features for automotive protocols.

2) Attack Vectors: To comprehensively assess the EPS control system, a range of attack vectors tailored to automotive systems are considered:

- **Invalid Messages:**
  - **Input:** Send a message with missing or extra parameters.
  - **Expected behavior:** Graceful handling of the invalid message without system crashes.
- **Out-of-Bounds Values:**
  - **Input:** Provide a steering angle value beyond the specified range.
  - **Expected behavior:** System should handle the out-of-bounds value gracefully, possibly with a warning or by clamping the value.
- **Fuzzing Message Timing:**
  - **Input:** Introduce delays or send messages at irregular intervals.
  - **Expected behavior:** System should remain responsive and recover gracefully from timing variations.
- **Randomized Data:**
  - **Input:** Use random data as input for various parameters.
  - **Expected behavior:** System should handle the randomness without crashing and maintain stability.
- **Protocol Manipulation:**
  - **Input:** Modify the communication protocol mid-session.
  - **Expected behavior:** System should detect and handle protocol changes appropriately, possibly with an error message.
- **Environmental Factors:**
  - **Input:** Introduce simulated noise into the system.
  - **Expected behavior:** System should filter out noise and maintain accurate steering control.
- **Concurrency Issues:**
  - **Input:** Simulate multiple steering requests simultaneously.
  - **Expected behavior:** System should handle concurrent requests gracefully, avoiding conflicts.
- **Power Fluctuations:**
  - **Input:** Mimic fluctuations in power supply.
  - **Expected behavior:** System should continue functioning or enter a safe state during power fluctuations.
- **Memory Corruption:**
  - **Input:** Inject data that could potentially corrupt memory.
  - **Expected behavior:** System should have safeguards against memory corruption, preventing crashes
- **Boundary Value Testing:**
  - **Input:** Test extreme values for steering parameters.
  - **Expected behavior:** System should handle boundary values correctly and safely.

C. Requirement Testing

1) Availability:

- **AVA-1** Using data mass produced from AVA-1 using a script we will check if the values from that data pass the delimited threshold.
- **AVA-2** The power supply will experience periodic interruptions at different stages in the process, occurring at varying time intervals and specific time slots. This ensures that the duration of a power outage remains within acceptable limits and does not exceed the expected time frame.
- **AVA-3** The system must undergo rigorous testing in a virtualized environment, maintaining a constant response time without data loss while consistently handling an influx of 10G/s of data, without violating other requirements.

2) Confidentiality:

- **CONF-1** The code shall be examined to determine whether this protocol is being correctly implemented. The team security shall also do an eavesdrop test and try to read the data, which should be unreadable.
- **CONF-2** After all the security operations that handle sensitive data, review the Log file to verify whether any of that data is being displayed and also check the code responsible for logging for any violation of the log policies.
- **CONF-3** The security team will have an account from every ACL group, with each one attempting to access all the functionalities, and access if the every one that the group has access to works and the ones that the group is not authorized to use doesn't execute.

3) Integrity:

- **INT-1** The test should cover a wide range of scenarios, including high-load and high-volume conditions. The system's output must be rigorously evaluated against labeled data to validate its accuracy. Whenever feasible, simulations or practical tests should be performed to generate a significant volume of data for thorough assessment. This rigorous testing approach aims to guarantee that the system not only achieves but consistently maintains the specified accuracy level, thereby ensuring its reliability in diverse real-world situations.
- **INT-2** In the aforementioned system, it is expected that the output will consist of false-positive and false-negative results with the specified percentages. The evaluation of these cases will be conducted under high load, high volume, and extreme conditions. False positives and false negatives will be assessed by a meticulous analysis of the system's output compared to expected results, utilizing labeled data.
- **INT-3** The testing process will consist of two parts:
  1) Conducting a thorough analysis of the code to ensure that the messages generated possess Integrity Checks.
  2) Intercepting the messages exchanged between components and deliberately modifying them to examine the effectiveness of the integrity control system in detecting these manually altered packets.
- **INT-4** In a period of 48 hours, the system will run in a simulated environment, 25 different simulated anomalies will be given as input to the system at random intervals, and at critical stress points to ensure that even in the worst conditions anomalies can be detected in a timely matter.
- **INT-5** Real-time test shall be done with the least powerful sensors in terms of processing power available on the market. The CPU usage of said sensors shall be monitored and should not exceed 20% of the original CPU usage when the system is subjected to heavy loads.
- **INT-6** Our software will be compared to the 20 most commonly used EPS systems worldwide in a simulated environment. The monitoring of decision-making and their time frames shall be evaluated, and there should be no discernible disparity between EPS using the software and those not using the software.
- **INT-7** The code responsible for logging and verifying the correct logging of predefined parameters should be analyzed. Additionally, a comparison between the logs generated in a simulated environment and the logs calculated from the same environment needs to be conducted.
- **INT-8** The system shall undergo a test to generate log data that fills the storage and exceeds the specified limits. Additionally, another test shall ensure that some logs generated have a time span greater than 6 months, in order to verify proper removal of such logs.
- **INT-9** Attempts will be made to remove flagged logged files to ensure their persistence within the system. Initially, these attempts will be carried out by unauthorized users to ensure they are unable to perform this action. Subsequently, specialists will also attempt to remove this data, and the integrity of the log file must be ensured.

D. User

- **USR-1** In the EPS control system, it is expected to download and install updates only from official servers. To verify this, simulation attempts will be made to download the same update file from a non-official source, confirming the system's rejection of unauthorized servers. Additionally, the system's ability to authenticate the official server will be tested, including the creation of a simulated server to mimic a legitimate one, ensuring that updates are accepted only from the designated source.

E. Development

- **DEV-1** Compilation of all the components must be done without compiling errors using gcc/g++ using version 17.

V. Hazard and Threat analysis

This section is crucial as it involves identifying and evaluating potential risks, hazards, and threats that could impact the system's performance, reliability, and security. The analysis aims to proactively address these concerns by outlining strategies for risk mitigation and system fortification.

- **Denial of Service:** This hazard refers to intentional actions or events that disrupt the normal operation of the Electric Power Steering (EPS) control system, leading to a loss of service or functionality.
  - **Existing safety measures**: Limits on the number of simultaneous connections to the EPS system. **AVA-1**. **AVA-3**.
  - **Likelihood of happening**: Rare.
  - **Severirty**: Minor.
  - **Risk ranking**: Negligible.
- **Eletronic noise:** Electronic noise refers to unwanted electrical signals or interference that can disrupt the normal functioning of the Electric Power Steering (EPS) system.

  - **Existing safety measures**: Shielding and Grounding. **INT-3**. **INT-9**.
  - **Likelihood of happening**: Frequent or almost certain.
  - **Severirty**: Insignificant.
  - **Risk ranking**: Tolerable.
- **Power fluctuations:** Power fluctuations refer to variations in electrical power supply, including sudden surges, drops, or interruptions, which can impact the Electric Power Steering (EPS) system.
  - **Existing safety measures**: Warning to the user. Enter safe mode. **AVA-2**.
  - **Likelihood of happening**: Likely.
  - **Severirty**: Insignificant.
  - **Risk ranking**: Tolerable.
- **Sensors manipulation:** Sensors manipulation involves unauthorized interference with sensors responsible for providing critical data to the Electric Power Steering (EPS) system.
  - **Existing safety measures**: **CONF-1**. **CONF-3**. **INT-9**.
  - **Likelihood of happening**: Rare.
  - **Severirty**: Major.
  - **Risk ranking**: Tolerable.
- **Unauthorized Firmware updates:** Unauthorized firmware updates involve the installation of unapproved or compromised software into the Electric Power Steering (EPS) control unit.
  - **Existing safety measures**: **CONF-1**. **INT-3**. **USR-1**.
  - **Likelihood of happening**: Rare.
  - **Severirty**: Major.
  - **Risk ranking**: Tolerable.

REFERENCES

[1] https://completecar.ca/maintenance/electric-power-steering-failure-symptoms/

[2] https://www.brakeandfrontend.com/electric-power-steering-sensors/