Case Study: Spoofing Attack On Autonomous Vehicles

Shihab Ud Doula
Matriculation No. 2190679

Project Work
Bachelor of Engineering - Electronic Engineering

HOCHSCHULE HAMM-LIPPSTADT

Main Advisor: **Prof. Dr. João Paulo Javidi da Costa**
Assistant Advisor: **Luis Felipe Oliveira de Melo**

© Main Advisor: **Prof. Dr. João Paulo Javidi da Costa**

© Assistant Advisor: **Luis Felipe Oliveira de Melo**

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my supervisor, **Prof. Dr.-Ing. João Paulo J. da Costa**, from the Department of Engineering at Hamm-Lippstadt University of Applied Sciences. His guidance and feedback throughout this project have been invaluable. I appreciate the academic and professional insights he provided, which have significantly contributed to the successful completion of this work.

I would also like to thank **Luis Felipe Oliveira de Melo** from the Department of Engineering for his readiness to assist during this project.

Finally, I extend my appreciation to the faculty members of the Department of Engineering at Hamm-Lippstadt University, whose instruction and support have been instrumental in my academic development.

Shihab Ud Doula

ABSTRACT

The paper covers a detailed overview of spoofing attacks on autonomous cars in three main forms: GPS spoofing, Sensor spoofing, and Time spoofing. Autonomous cars rely on good communication and sensors to find their way and take action. Spoofing attacks are critical, because they create false evidence and thus causing potentially dangerous situations..

For this reason, we used simulations in MATLAB to explore how spoofing attacks would affect self-driving vehicle systems. The simulations also offered important insight into the nature of these attacks – how they degrade fundamental aspects of autonomous systems. These findings showed the magnitude of spoofing error variations and highlighted specific spoofing dynamics such as amplitude differences, synchronisation mismatches, and altered sensor readings. The report stresses that we need strong countermeasures to limit these risks.

This paper also addresses the practical impacts of these attacks, including how they could lead cars into an unsafe route or lead to obstacle detection miscues. Recommendations are given for future research, with a focus on enhanced detection, real-time countermeasures and broad-scale testing in real-world scenarios to ensure the safety and security of autonomous vehicle systems.

This research highlights the importance of tighter security in autonomous vehicle communications. By discovering vulnerabilities and recommending research areas, this research hopes to help ensure safer and more stable self-driving cars by protecting them from these threats.

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

1.1 Background

Autonomous vehicles (AVs) are changing the way people move around with safer, more efficient, and smarter mobility. These cars depend on GPS, time keeping and other sensors to steer and take action in real time. But this reliance leaves them open to spoofing attacks that can mess up their operations. GPS spoofing can confuse navigation, time spoofing can generate synchronisation problems, sensor spoofing can misrepresent the environment. Such holes are very dangerous and warrant securing AV platforms from spoofing attacks.[14]

1.2 Problem Statement

Self-driving cars relies heavily on the reliability of data for their decisions. Spoofing attacks — that spoof data points such as GPS, time and sensors — could destroy these systems. Detection and response systems are simply too weak to respond to such advanced attacks, and AVs become vulnerable to navigation errors, failures in synchronization, and misunderstanding the environment. The study will investigate these weaknesses and suggest measures to counter them by simulation.[1]

1.3 Objectives

The objectives of this project are to:

- Create GPS, time and sensor spoofing simulating attacks on AV systems.
- Analyse how such attacks change the way that cars behave and make decisions.
- Look into how spoofing attacks can be detected and stopped.
- Improve the security and robustness of AV.

1.4 Project Scope and Challenges

The goal of this project is to simulate the three scenarios of spoofing attacks (GPS, time, and sensor spoofing) in MATLAB to mimic real life. We want to know how these attacks impact AV systems and evaluate ways to mitigate them. It is an exploratory study – and, for time and cost reasons, it is confined to simulations.

Another was keeping the simulations realistic but reducing computation overhead. The results themselves were hard to interpret in real life either, so analyzing them carefully and tweaking the simulation design was a crucial part of the project as well.[5]

1.5 Structure of the Report

The report is written in such a way that the study can be easily digested. In Chapter 2 we summarize research literature around spoofing attacks, providing examples to provide a basis for the project. Chapter 3 discusses the process of modelling spoofing attacks and analyzing their effect. Chapter 4 summarises the simulation results and observations. Chapter 5 describes the results (problems and prospects) in great detail. Lastly, Chapter 6 closes the report by reviewing the highlights and providing suggestions to further strengthen AV security.

Chapter 2

LITERATURE REVIEW

2.1 Overview of Spoofing Attacks

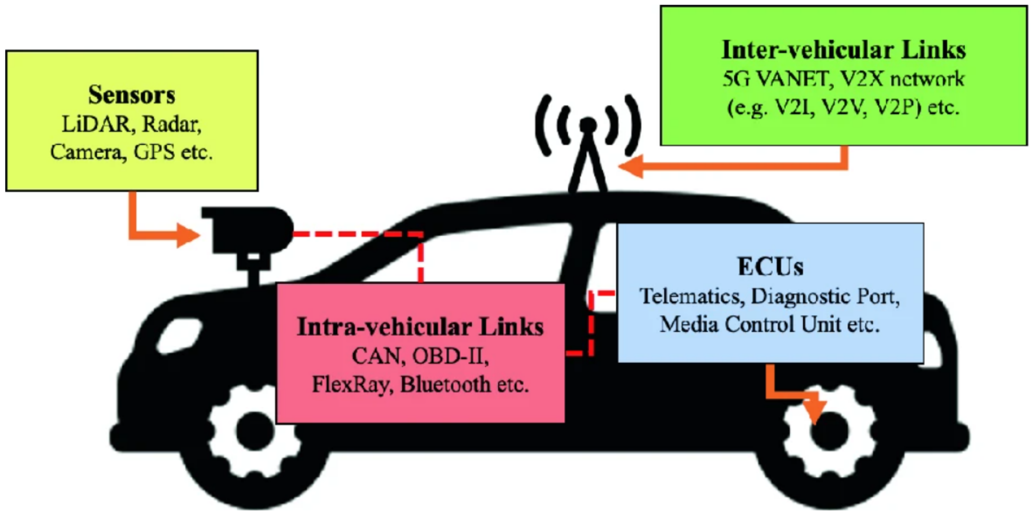
Sensors
LiDAR, Radar, Camera, GPS etc.
Inter-vehicular Links
5G VANET, V2X network (e.g. V2I, V2V, V2P) etc.
ECUs
Telematics, Diagnostic Port, Media Control Unit etc.
Intra-vehicular Links
CAN, OBD-II, FlexRay, Bluetooth etc.

Figure 2.1: Attack surfaces of autonomous vehicles, highlighting vulnerable components like sensors (LiDAR, GPS), intra-vehicular links (CAN, OBD-II), inter-vehicular links (5G VANET), and ECUs.[4]

Spoofing attacks pose a serious threat to self-driving vehicles because they decipher information from sensors or communication systems in order to interfere with driving. By sending fake signals, these attacks attack the perception, choice, and navigation system of the car. GPS signals, time synchronisation algorithms, and sensor readings such as LiDAR images are typical spoof targets.

These systems are a critical part of how self-driving cars operate safely and effectively. GPS, for example, supplies coordinates to guide navigation, time synchronisation ensures intersystem coordination, and LiDAR detects obstacles in real-time. Spoofing attacks take advantage of this trust placed in third-party data and can cause safety-compromising errors.

The study covers three kinds of spoofing attacks:

- GPS Spoofing: Leaking the vehicle's navigation system by sending out fake GPS messages.

- Time Spoofing: Affecting system synchronisation by manipulating time signals.
- LiDAR Spoofing: Using distance measurements to falsely present obstacles.

The following sections will cover case studies that illustrate the real-world impacts of such spoofing attacks on self-driving vehicles. These cases were chosen to help provide a sense of how such attacks are conducted, how they happen, and what countermeasures exist.

2.2 Case Study: GPS Spoofing Attack on Tesla Model 3 by Regulus Cyber (2019)

Introduction to Regulus Cyber

Regulus Cyber is a cybersecurity company specializing in GNSS (Global Navigation Satellite System) Security. In 2019, they performed an unprecedented GPS spoofing experiment with the Tesla Model 3, which sought to demonstrate how self-driving cars with highly relied-upon GPS navigation systems can fail. While Tesla is secure, this test exposed vulnerabilities that could be tapped by hacked spoofers.[10]

Tesla Model 3's Dependency on GPS

The Tesla Model 3 uses GPS data, along with other sensors, to aid navigation and driver assistance functions like Navigate on Autopilot. With GPS signals so central, any data manipulation can make a significant impact on the car's decision-making.[6]

Spoofing Methodology

Regulus Cyber deployed inexpensive hardware and software-defined radio (SDR) devices to broadcast spoofed GPS signals. These false signals superseded real ones, deceiving the Tesla's mapping system into making erroneous decisions about where to go.[8]

Execution of the Attack

In an artificial drive test, the fake GPS signals supplied the Tesla Model 3 with inaccurate location information, leading to:

- **Navigation errors:** Incorrect turns, route deviations, and sudden speed changes.[8]

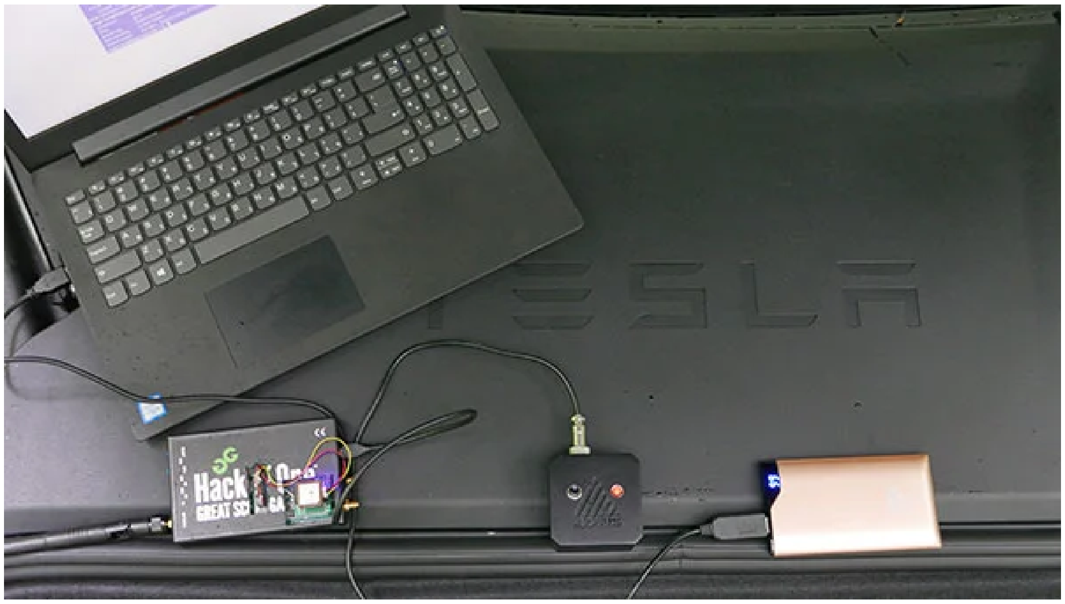

Figure 2.2: GPS Spoofing Equipment used in the Tesla Model 3 test by Regulus Cyber[12]

- **Unexpected system impacts:** The attack caused unpredictability in the vehicle's air suspension, which lowered the ride height while driving.[10]

Impact on Driver Assistance Features

The fake attack broke help services like Navigate on Autopilot. This resulted in reckless manoeuvres such as sudden lane changes and miscalculations of speed. Such mistakes presented severe dangers to occupants and other road users.[6]

System's Response

The Tesla Model 3 did not see or correct the fake GPS signals and drove in dangerous directions. This failure highlighted the need for better anomaly detection and sensor-fusion algorithms to ensure trust.[10]

2.3 Case Study: Tesla Autopilot Time Delay Vulnerability by Keen Security Lab (2019)

Introduction to Keen Security Lab

In 2019, Keen Security Lab, a cybersecurity research arm of Tencent, performed a full study to spot bugs in Tesla's Autopilot. Their work aimed to demonstrate how time delays in self-driving vehicles can cause drivers to make the wrong or dangerous choices.[11]

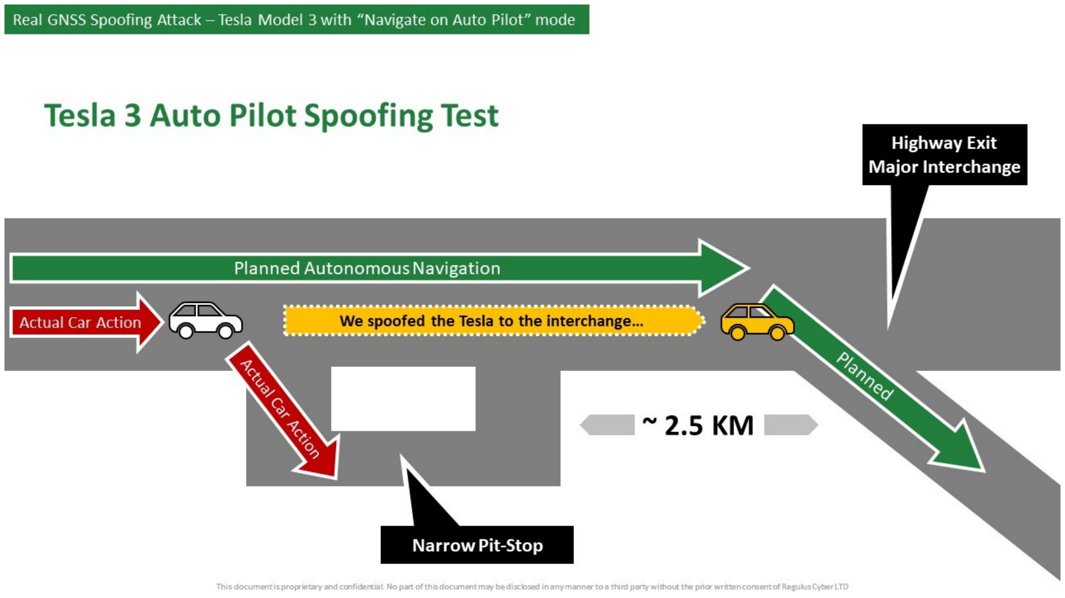
Real GNSS Spoofing Attack – Tesla Model 3 with "Navigate on Auto Pilot" mode
Tesla 3 Auto Pilot Spoofing Test
Highway Exit Major Interchange
Planned Autonomous Navigation
Actual Car Action
We spoofed the Tesla to the interchange...
Actual Car Action
Planned
~ 2.5 KM
Narrow Pit-Stop
This document is proprietary and confidential. No part of this document may be disclosed in any manner to a third party without the prior written consent of Regulus Cyber LTD

Figure 2.3: Path deviation of Tesla Model 3 due to GPS spoofing.[7]

Tesla Autopilot System and Timing Dependency

Tesla's Autopilot, which makes real-time driving decisions, relies on a mix of sensors (radar, ultrasonic, and cameras). The precision of these choices depends on the exact timing of incoming data streams. Even minute-time errors can morph into serious navigational mistakes, making the car's behaviour erratic.[11]

Spoofing Mechanism

Keen Security Lab demonstrated how spoofing could take advantage of the Tesla's need for clockwise sensor readings. This caused delays in the processing of sensor data that misinterpreted important information like distance from other objects.[11]

Impact on Vehicle Behavior

The timing spoofing attack hacked tools such as Navigate on Autopilot, making:

- **Navigation Errors:** Temporary lane adjustments due to mispaired sensors.
- **Collision Risks:** In one test, a delay in radar data caused the vehicle to fail to stop for an obstacle, resulting in a near-collision scenario.
- **Synchronization Failures:** Misaligned data streams created inconsistencies in the system's environmental model, affecting its ability to make accurate decisions.

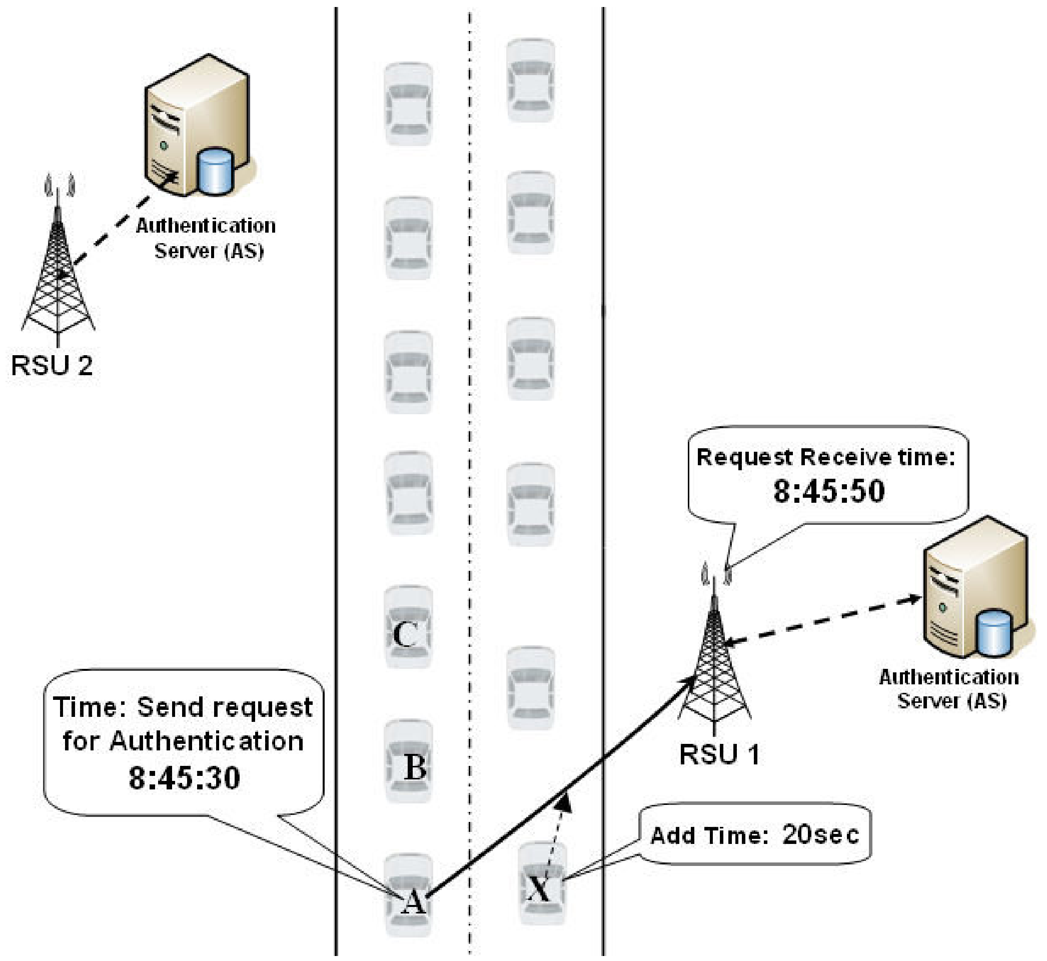
Authentication Server (AS)
RSU 2
Request Receive time: 8:45:50
Authentication Server (AS)
RSU 1
C
Time: Send request for Authentication 8:45:30
B
Add Time: 20sec
A
X

Figure 2.4: Illustration of timing spoofing in vehicle authentication, demonstrating the delay introduced in the signal synchronization.[15]

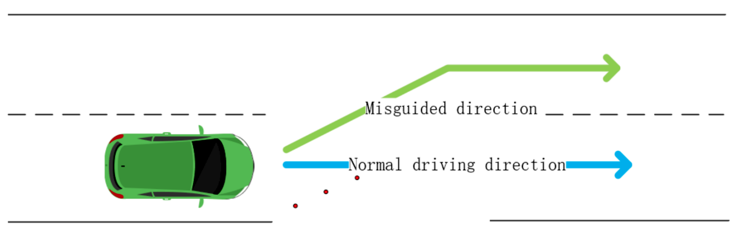

Figure 2.5: Example of a misguided driving direction caused by timing spoofing, showing how delays can mislead the vehicle's decision-making.[9]

System's Response

In response to these vulnerabilities, Tesla issued software updates to mitigate processing delays and improve sensor synchronization. While these updates did limit exposure, the Keen Security Lab study noted that the flaws were not fully eliminated, and that further improvement and detection are needed.[11]

Lessons and Recommendations

The Keen Lab case study provided the following insights and recommendations:

- **Real-Time Monitoring:** Autonomous systems should incorporate real-time detection mechanisms to identify timing discrepancies as they occur.
- **Sensor Redundancy:** Cross-verifying data from multiple sensors can help mitigate the effects of time spoofing by providing alternative sources of accurate information.
- **Advanced Algorithms:** Predictive algorithms should be developed to compensate for timing discrepancies and enhance the resilience of autonomous systems.

2.4 Case Study: LiDAR Spoofing Attack by Cao et al. (2024)

Introduction to Sensor Spoofing

Cao et al. (2024) exposed the method by which LiDAR spoofing attacks corrupt distance data on LiDAR devices by adding false reflections to sensor data. Such a attack can distort decision-making in self-driving cars to the point of incorrect lane adjustments.[13]

Spoofing Mechanism

The attack involves using a laser to generate false reflections that the LiDAR system interprets as real obstacles. The setup includes:

- A timer for the fake reflections.
- Photodiode and lens to correctly project these deceptive reflections back into the LiDAR sensor.

Impact on Vehicle Trajectories

Through inflating obstacles or manipulating data for legitimate reflections, the spoofing attack resulted:

- Significant errors in trajectory planning. The vehicles were rigged to detect blind spots.

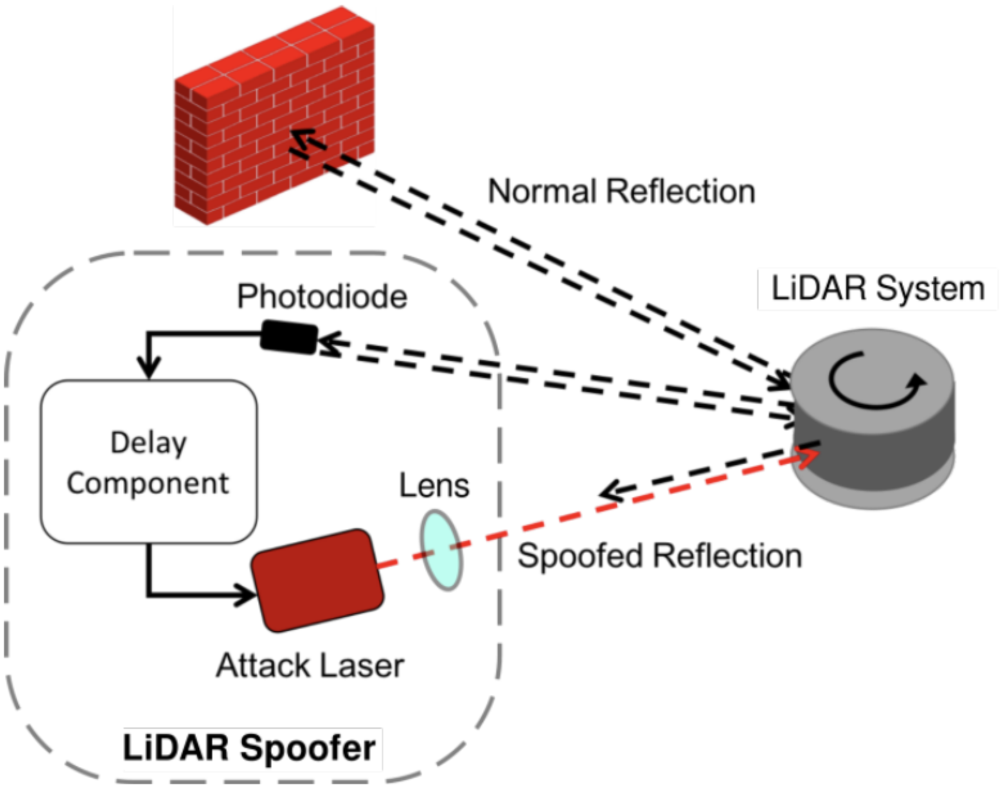

Figure 2.6: LiDAR Spoofing Setup as demonstrated by Cao et al. (2024).[16]

- Potential collisions or unsafe maneuvers. A mistaken direction correction might send the car flying.

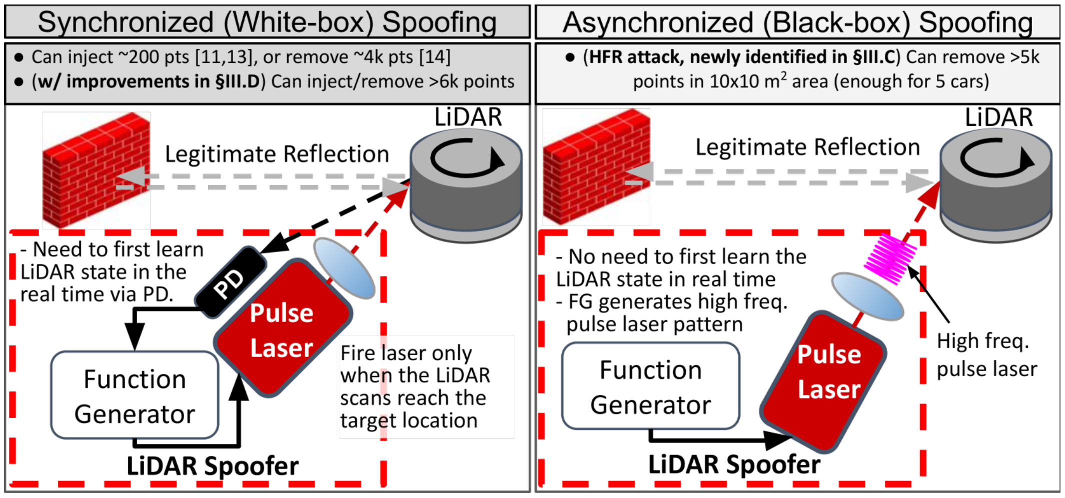
Synchronized (White-box) Spoofing
● Can inject ~200 pts [11,13], or remove ~4k pts [14]
● (w/ improvements in §III.D) Can inject/remove >6k points
LiDAR
Legitimate Reflection
- Need to first learn LiDAR state in the real time via PD.
PD
Pulse Laser
Function Generator
Fire laser only when the LiDAR scans reach the target location
LiDAR Spoofer
Asynchronized (Black-box) Spoofing
● (HFR attack, newly identified in §III.C) Can remove >5k points in 10x10 m² area (enough for 5 cars)
LiDAR
Legitimate Reflection
- No need to first learn the LiDAR state in real time
- FG generates high freq. pulse laser pattern
Pulse Laser
Function Generator
High freq. pulse laser
LiDAR Spoofer

Figure 2.7: Example of a trajectory misguidance caused by LiDAR spoofing.[16]

System Response and Vulnerabilities

This research suggested that current LiDAR systems do not offer good spoofing countermeasures. If it's not properly validated, forged information is easily misinterpreted as real. Researchers emphasized the need for:

- Cross-validation of LiDAR measurements with other data from multiple sensors.[2]
- Alerting on anomalies in real-time so you can detect fake signals in an emergency.
- Improved LiDAR hardware in order to mitigate these attacks.

Lessons and Recommendations

Cao et al. determined that LiDAR hardware and software must be updated in order to prevent and block spoofing attacks. Their research reflects the need for continual refinement in security systems for autonomous vehicle operations.[3]

Summary of Case Studies

The case studies presented in this chapter illustrate how vulnerable autonomous vehicles are to spoofing attacks. GPS spoofing, time spoofing, and sensor spoofing were all revealed to exploit specific bugs in the navigation and decision systems of the vehicle. Some patterns began to resurface, like reliance on single source data and lack of real-time anomaly detection. Such flaws also highlight the need for comprehensive cybersecurity to protect and secure autonomous systems.

Chapter 3

METHODOLOGY

3.1 Simulation Setup

This paragraph will outline the setup and tools used to run these simulations. These simulations were designed to model and visualize the impact of spoofing attacks on autonomous vehicle systems, ensuring consistency and reliability in every environment.

Tools and Environment

All of the simulations were built and simulated using MATLAB R2024a on a Windows 11 PC. The strong computational power of MATLAB and its eminent visualization tools made it suitable for simulating and analyzing the effects of spoofing attacks in a controlled environment. Hardware and software specifications:

The computer specifications used for the simulations are:

- **Operating System:** Windows 11
- **Processor:** Intel Core i5 (10th Generation)
- **RAM:** 16 GB
- **Software:** MATLAB R2024a

Simulation Parameters

These were the parameters applied to assure consistency in all spoofing attack simulations:

- **Duration:** Each of these simulations has a duration of 10 seconds, which gives enough time to watch how spoofing happens from start to end
- **Time Resolution:** A 1,000 point (0.01-second time steps) was applied for accuracy of modelling and smooth display.
- **Vehicle Dynamics:** The car moved in a straight line with a velocity of 5 m/s. With this simplified dynamic, we can reduce the spoofing effects and do an in-depth analysis.

Visualization Approach

Visualization was an integral element of the simulations. Visualization helped understand and assess the results well.

- **Real vs. Spoofed:** Attempts to compare real sensor signals with spoofed data are shown to highlight inconsistencies in spoofing attacks.
- **Dynamic Visualization:** Time-sensitive updates plots were used to measure the spoofing effects across the simulation.
- **Calculated Metrics:** Additional graphs were included such as velocity, acceleration, and change rate to describe the effects of the spoofing attack on car perception and decision.

By using a synchronised setup for all situations, simulations offered valuable and comparable insights into the impact of GPS, time, and LiDAR sensor spoofing on autonomous vehicles.

3.2 GPS Spoofing Simulation Methodology

Objective

This simulation is intended to show how GPS spoofing can affect a vehicle's navigation system. Simulating spoofing effects such as amplitude variation, frequency changes, random walk, and Gaussian noise can help to show how these influence the position, velocity, and acceleration of the vehicle.

Approach

The simulation follows an iterative format to keep things simple and easy:

1. The vehicle's original orientation is a straight-line GPS location.
2. Design spoofing dynamically through mathematical models that model deviations from the GPS signal.
3. Model the effect of spoofing on:
    - **Position**: Showing the deviation between the original and spoofed GPS paths.
    - **Velocity and Acceleration**: Highlighting the changes caused by the spoofing dynamics.

Mathematical Model

The simulation uses a simple mathematical model to compute the true and fake position, and the resulting velocity and acceleration.

Original Path The initial trajectory of the vehicle is assumed to be linear:

$$y_{\text{original}}(t) = t \tag{3.1}$$

This represents the vehicle moving at a constant speed over time.

Spoofing Dynamics At $t = 2$ seconds, spoofing effects are introduced, and the vehicle's position begins to deviate from the original path. The spoofed position is defined as:

$$y_{\text{spoofed}}(t) = y_{\text{original}}(t) + A(t) \cdot \sin(f(t) \cdot t) + W(t) + N(t) \tag{3.2}$$

where:

- $A(t) = 1.0 + 0.3 \cdot \sin(0.1t)$: Defines the change in amplitude over time.
- $f(t) = 1.0 + 0.1 \cdot \cos(0.05t)$: Represents the variation in frequency.
- $W(t) = \sum_{i=1}^{n} 0.05 \cdot \text{randn}()$: Represents a random walk (cumulative sensor drift).
- $N(t) = 0.2 \cdot \text{randn}()$: Adds Gaussian noise for small-scale randomness.

Velocity and Acceleration The changes in velocity and acceleration due to spoofing are derived numerically using finite differences:

$$v_{\text{spoofed}}(t) = \frac{\Delta y_{\text{spoofed}}(t)}{\Delta t} \tag{3.3}$$

$$a_{\text{spoofed}}(t) = \frac{\Delta v_{\text{spoofed}}(t)}{\Delta t} \tag{3.4}$$

Implementation Steps

The following procedures were executed to implement and test the effects of GPS spoofing:

1. **Time Setup**: A time vector t was created over 10 seconds with 1000 points to ensure smooth updates.

2. **Position Calculation**:
   - The original position $y_{\text{original}}(t)$ was calculated as a straight line path.
   - The spoofed position $y_{\text{spoofed}}(t)$ was calculated using a combination of amplitude variation, frequency shifts, random walk, and Gaussian noise.
3. **Velocity and Acceleration Computation**:
   - Velocity was obtained by numerically differentiating the spoofed position.
   - Acceleration was calculated by differentiating the velocity.
4. **Visualization**:
   - The original and spoofed positions were plotted dynamically.
   - Velocity and acceleration were visualized in separate plots to analyze the effects of spoofing.

Visualization and Results

The simulation provides three main visualizations for the consequences of GPS spoofing:

- **Position Plot**: Shows the original straight-line trajectory alongside the spoofed path. Deviations begin at $t = 2$ seconds, and the spoofed position exhibits oscillations, drift, and noise due to the applied spoofing dynamics.
- **Velocity Plot**: Highlights a sharp spike in velocity when spoofing begins, followed by irregular oscillations caused by amplitude and frequency variations.
- **Acceleration Plot**: Shows significant fluctuations in acceleration, which reflect the abrupt and dynamic changes in velocity over time.

These results align with the intended goal of demonstrating the impact of spoofing on a vehicle's navigation system.

Reflection and Challenges

The simulation offered some practical examples of the effects of spoofing on navigation information. Here are some of the implementation problems:

- Balancing realism and simplicity in the mathematical models while ensuring computational efficiency.

- Adjusting the random walk and Gaussian noise parameters to avoid unrealistic results.
- Designing clear visualizations to effectively highlight the deviations in position, velocity, and acceleration.

However, despite these issues, the final implementation manages to show how spoofing attacks spread through GPS data.

3.3 Time Spoofing Simulation Methodology

Objective

The simulation is intended to help people better understand how time spoofing might affect an autonomous vehicle's navigation system. Attacks using time spoofing introduce flaws in the time signal that may disturb the functioning of the system. This simulation enables us to observe these effects in a simulation environment, using mathematical simulations and plots.

Approach

The simulation follows these steps:

1. Define a straightforward time signal that progresses linearly.
2. Introduce spoofing by adding a combination of linear deviations, sinusoidal fluctuations, and random noise to simulate real-world attack scenarios.
3. Display the original and spoofed time signals, along with additional metrics such as time offset and the rate of change, in an animated format.

Mathematical Model

The simulation uses the following mathematical components:

Original Time Signal The original time signal is modeled as:

$$T_{\text{original}}(t) = t \tag{3.5}$$

This represents a simple, linear progression of time.

Spoofing Dynamics Spoofing starts at $t = 2$ seconds. The spoofed time signal is modeled as:

$$T_{\text{spoofed}}(t) = T_{\text{original}}(t) + \Delta T_{\text{linear}}(t) + \Delta T_{\text{sinusoidal}}(t) + \Delta T_{\text{noise}}(t) \tag{3.6}$$

where:

- $\Delta T_{\text{linear}}(t) = 0.5 \cdot (t - t_{\text{start}})$ is a simple and growing deviation over time.
- $\Delta T_{\text{sinusoidal}}(t) = 0.2 \cdot \sin(2\pi \cdot 0.5 \cdot t)$ adds a periodic fluctuation.
- $\Delta T_{\text{noise}}(t) = 0.1 \cdot \text{randn}()$ introduces random Gaussian noise to mimic unpredictable variations.

Time Offset The time offset, which shows how much the spoofed signal deviates from the original, is calculated as:

$$\text{Time Offset}(t) = T_{\text{spoofed}}(t) - T_{\text{original}}(t) \tag{3.7}$$

Rate of Change The rate of change in the spoofed signal is calculated numerically to observe abrupt fluctuations:

$$\text{Rate of Change}(t) = \frac{\Delta T_{\text{spoofed}}(t)}{\Delta t} \tag{3.8}$$

Implementation Steps
Here's how the simulation was implemented:

1. Create a time vector t that spans 10 seconds with 1000 points for smooth animations.
2. Calculate the original and spoofed time signals using the equations above.
3. Compute the time offset and rate of change to understand the impact of spoofing.
4. Use animated plots to display:
   - The original and spoofed time signals with annotations marking when spoofing starts.
   - The time offset with a threshold line to show when spoofing becomes noticeable.
   - The rate of change with labels marking the highest fluctuation.

Visualization

The simulation produces three key plots:

- **Original vs. Spoofed Time Signal:** Shows the original time progression compared to the spoofed signal. A shaded region highlights the spoofing duration, and annotations mark when spoofing begins.
- **Time Offset:** Displays how much the spoofed signal deviates from the original over time, with a threshold line to indicate when the offset becomes significant.
- **Rate of Change:** Visualizes the sudden changes in the spoofed signal, helping to identify irregularities caused by the spoofing attack.

Reflection and Challenges

Building this simulation made it easier to understand how time spoofing can disrupt a vehicle's navigation system. One challenge was balancing the complexity of the spoofing effects (like noise and sinusoidal disturbances) with the need to keep the simulation smooth and understandable. Another difficulty was ensuring the annotations were clear and placed at the right moments, such as when spoofing starts or when a threshold is crossed. Overall, the simulation captures the key impacts of time spoofing in a way that's easy to present and explain.

3.4 LiDAR Sensor Spoofing Simulation Methodology

Objective

This simulation aims to demonstrate how spoofing attacks affect a LiDAR sensor, which autonomous vehicles use to detect obstacles. Simulating a spoofing attempt shows how a poor measurement of distance could result in an error in understanding the landscape. It aims to investigate and discern how attacks like this can manipulate distance data and visualise those effects easily and precisely.

Approach

The simulation follows these steps:

1. Define the true obstacle distance as a constant measurement.
2. Insert spoofing elements: Linear drift, sinusoidal noise, Gaussian noise in a predetermined period of time.

3. Compute and plot the difference between the real and spoofed distances, and also the change in measurement.
4. Note important times like the beginning and ending of spoofing, reaching the detection limit, and maximum change rate.

Mathematical Model

The simulation is based on the following mathematical concepts:

True Distance: The actual distance to the obstacle is assumed to be constant and is given by:

$$d_{\text{true}}(t) = 20 \text{ meters} \tag{3.9}$$

Spoofing Dynamics: Spoofing is applied between t_{start} and t_{end} and is modeled as:

$$d_{\text{spoofed}}(t) = d_{\text{true}}(t) + \Delta d_{\text{linear}}(t) + \Delta d_{\text{sinusoidal}}(t) + \Delta d_{\text{noise}}(t) \tag{3.10}$$

where:

- $\Delta d_{\text{linear}}(t)$: A gradual drift simulating an increase or decrease in distance over time:
$$\Delta d_{\text{linear}}(t) = 0.5 \cdot (t - t_{\text{start}}) \tag{3.11}$$
- $\Delta d_{\text{sinusoidal}}(t)$: A periodic fluctuation added to simulate a sinusoidal disturbance:
$$\Delta d_{\text{sinusoidal}}(t) = 2 \cdot \sin(2\pi \cdot 0.2 \cdot t) \tag{3.12}$$
- $\Delta d_{\text{noise}}(t)$: Random noise to represent sensor inaccuracies:
$$\Delta d_{\text{noise}}(t) = 0.5 \cdot \mathcal{N}(0, 1) \tag{3.13}$$

Distance Deviation: The deviation from the true distance is calculated as:

$$\Delta d(t) = d_{\text{spoofed}}(t) - d_{\text{true}}(t) \tag{3.14}$$

Rate of Change: The rate of change of the spoofed distance is derived numerically:

$$\frac{d}{dt} d_{\text{spoofed}}(t) = \frac{\Delta d_{\text{spoofed}}(t)}{\Delta t} \tag{3.15}$$

Implementation Steps

The implementation is carried out as follows:

1. Generate a time vector t spanning 10 seconds with 1000 points for smooth updates.
2. Define the true obstacle distance as a constant value of 20 meters.
3. Simulate the spoofing attack by applying the linear drift, sinusoidal disturbance, and Gaussian noise within the specified spoofing duration ($t_{\text{start}} = 3$ seconds, $t_{\text{end}} = 6$ seconds).
4. Calculate the spoofed distance, distance deviation, and rate of change using the equations provided.
5. Create a three-part visualization that includes True vs. Spoofed Distance, Distance Deviation and Rate of Change.
6. Add annotations to indicate the start and end of spoofing, as well as key events like threshold crossings and maximum rate of change.
7. Implement a real-time simulation to update the visualizations dynamically.

Visualization and Results

The simulation provides the following insights:

- **True vs. Spoofed Distance:** The top plot displays the true and spoofed distances over time. A red-shaded region indicates the spoofing duration, with annotations marking the start and end of the attack.
- **Deviation from True Distance:** The middle plot shows how much the spoofed distance deviates from the true value. A red dashed line represents the detection threshold, and annotations highlight when the deviation crosses this threshold.
- **Rate of Change:** The bottom plot visualizes the rate at which the spoofed distance changes over time. An annotation marks the point of maximum rate of change during the attack.

Reflection and Challenges

This simulation exposes us to the dangers of LiDAR spoofing attacks on autonomous vehicles. By monitoring deviation and change rate, we begin to see the impact that spoofing is having on the sensor's functioning. The key was making the spoofing dynamics look realistic without making them too complex for computation. The other big challenge was making sure the visualizations were transparent and could clearly communicate how the spoofing attack progressed and affected the sensor. These challenges demanded a trade-off between technical precision and visual appeal to deliver practical outcomes.

3.5 JavaScript Implementation of GPS Spoofing Simulation

Overview

The GPS spoofing simulation was implemented using **JavaScript** with the **Leaflet.js** library for interactive map visualization. This approach improves upon the earlier MATLAB version, which used Folium for static visualizations.

The main goals of this implementation were:

- Fetch real-world routes dynamically using the OpenRouteService (ORS) API.
- Simulate vehicle movement along the original path with animation.
- Implement spoofing dynamics step by step to show variations from GPS spoofing.
- Perform a massive spoof to take the vehicle in the wrong direction.

Route Fetching Using OpenRouteService (ORS) API

To get the real-world car routing, I used the OpenRouteService (ORS) API. I signed up for the free API key, which lets me send 2,500 requests a day, good enough for the project.

The API request uses the `driving-car` endpoint and requires the start and end GPS coordinates (latitude and longitude). The returned route consists of multiple GPS waypoints.

Example: Requesting a route from the dorm to the university:

```
fetch('https://api.openrouteservice.org/v2/directions/driving-car?
api_key=<MY_API_KEY>&start=<longitude,latitude>&end=<longitude,latitude>')
```

```
.then(response => response.json())
.then(data => {
const route = data.features[0].geometry.coordinates.map(coord =>
[coord[1], coord[0]]);
});
```

The fetched route is visualized as a blue polyline on the map using Leaflet.

Vehicle Animation Along the Path

To animate the vehicle:

- A loop iterates over the route waypoints.
- The car marker's position is updated step by step using the `setTimeout` function.
- The animation speed was set to 1 seconds per step for better observation.

Animation Logic:

```
function animateCar(car_marker, path, intervalMs) {
    let index = 0;
    function move() {
        if (index >= path.length) return;
        car_marker.setLatLng(path[index]); // Update position
        index++;
        setTimeout(move, intervalMs);
    }
    move();
}
```

Applying Spoofing Dynamics

The spoofing effects were applied gradually:

- **Amplitude Variation:** Introduces oscillations in position.
- **Frequency Variation:** Varies oscillation frequency to create irregular disturbances.
- **Random Walk:** Adds cumulative drift over time.

- **Gaussian Noise:** Introduces small random errors to mimic GPS signal interference.

Each dynamic was applied incrementally, causing visible deviations from the original path, which were visualized as an orange polyline.

Major Spoofing Event

At a specific point on the route, all spoofing effects were combined to trigger a major spoofing event. This redirected the car onto a predefined wrong path, fetched separately using the ORS API.

The wrong path was displayed as a red polyline on the map to highlight the vehicle's misdirection due to spoofing.

Results

The JavaScript simulation successfully demonstrated:

1. The vehicle followed its original route (blue line) initially.
2. Incremental spoofing dynamics caused visible deviations (orange line).
3. The major spoofing attack redirected the vehicle onto the wrong route (red line).

This implementation provided a realistic, step-by-step visualization of GPS spoofing effects using real-world route data.

Conclusion

This JavaScript simulation reproduces the logic of the previous MATLAB implementation but makes it even more realistic with actual GPS data pulled from the ORS API. Using Leaflet for visualization also helps to visualize both spoofing dynamics and the big spoofing incident so it's easy to grasp how GPS spoofing slowly affects car navigation.

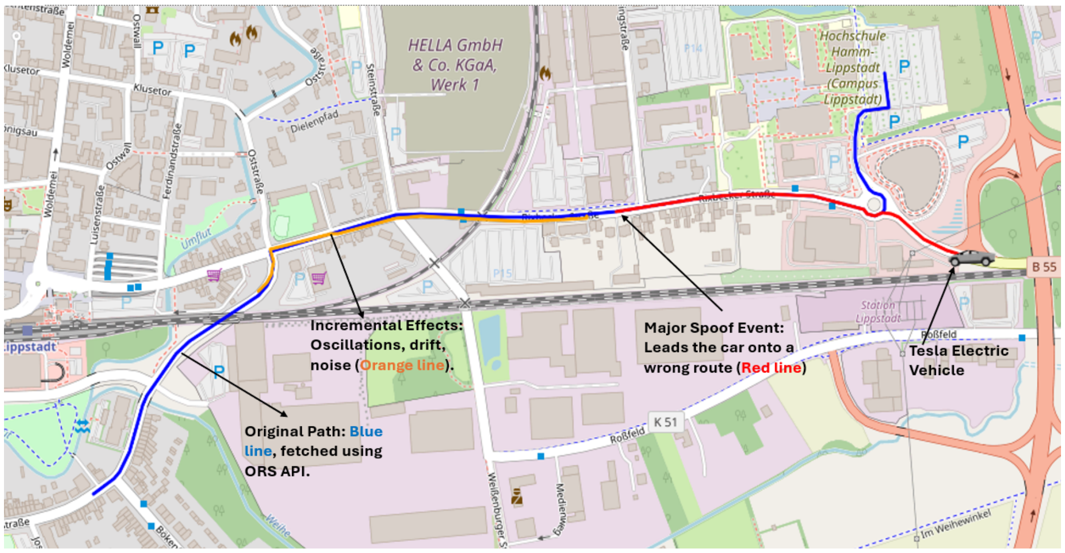
HELLA GmbH & Co. KGaA, Werk 1
Hochschule Hamm-Lippstadt (Campus Lippstadt)
Incremental Effects: Oscillations, drift, noise (Orange line).
Major Spoof Event: Leads the car onto a wrong route (Red line)
Tesla Electric Vehicle
Original Path: Blue line, fetched using ORS API.
B 55
K 51
Station Lippstadt
Roßfeld

Figure 3.1: A Screenshot of Javascript Simulation showing GPS spoofing by adding mentioned parameters in the report

Chapter 4

RESULTS AND ANALYSIS

4.1 Introduction

This chapter presents the results of the three spoofing simulations: GPS Spoofing, Time Spoofing, and LiDAR Sensor Spoofing. Each simulation is analyzed in detail, and the comparative metrics are discussed. The simulations showcase how spoofing attacks affect the respective systems, emphasizing the deviations, error percentages, and rates of change.

4.2 GPS Spoofing Results

Simulation Overview

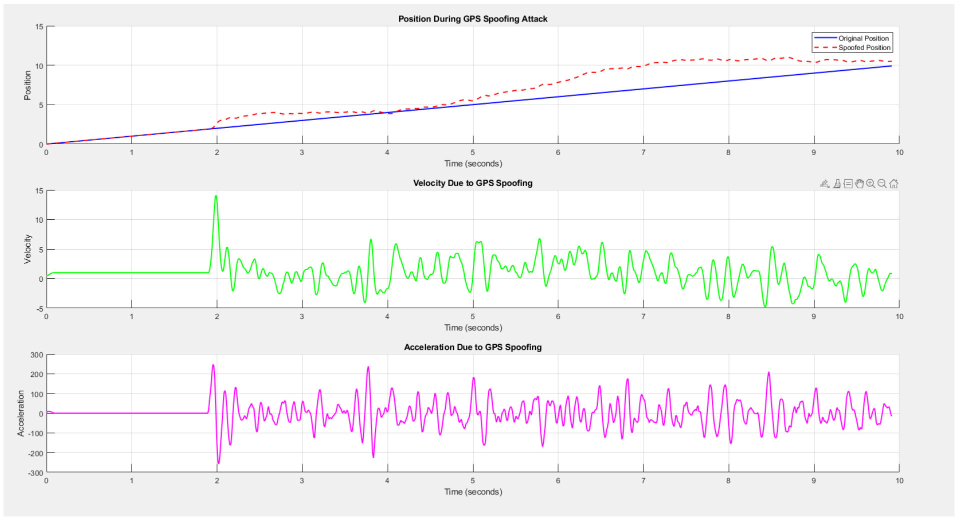

Figure 4.1: Position, Velocity, and Acceleration Plots for GPS Spoofing.

The GPS spoofing simulation introduces sinusoidal noise, random walk, and Gaussian noise to deviate the original GPS position. The results are presented in three plots:

- **Position Plot:** Comparison between the original position (blue) and the spoofed position (red), highlighting the deviation caused by spoofing starting at $t = 2$ seconds.
- **Velocity Plot:** Shows the changes in velocity due to spoofing, with significant fluctuations observed after $t = 2$ seconds.

- **Acceleration Plot:** Displays abrupt changes in acceleration as spoofing introduces dynamic disturbances.

Key Observations

- **Position Plot:** The spoofed signal (red) starts to deviate from the original position (blue) at $t = 2$ seconds. The maximum deviation occurs around $t = 7$ seconds.
- **Velocity Plot:** Velocity fluctuations are visible after $t = 2$ seconds, indicating the start of spoofing effects. These fluctuations continue throughout the simulation.
- **Acceleration Plot:** Significant oscillations in acceleration are observed due to dynamic spoofing effects, which highlight abrupt changes in velocity.

Quantitative Metrics

- **Maximum Deviation:**

$$\text{Max Deviation} = \max(|y_{\text{spoofed}} - y_{\text{original}}|) = 3.5\,\text{m} \tag{4.1}$$

- **Error Percentage:**

$$\text{Error Percentage} = \frac{\text{Max Deviation}}{\text{Path Length}} \times 100 = \frac{3.5}{10} \times 100 = 35\% \tag{4.2}$$

4.3 Time Spoofing Results

Simulation Overview

The time spoofing simulation introduces linear drift, sinusoidal disturbance, and Gaussian noise to the original time signal. The results are visualized in three plots: original vs. spoofed time signal, time offset, and rate of change.

Key Observations

- **Time Signal Plot:** The original signal (green) is a straight line, while the spoofed signal (red) deviates upward after $t = 2$ seconds, with the maximum offset at $t = 6$ seconds.
- **Rate of Change Plot:** Spikes in the rate of change are evident, corresponding to periods of significant spoofing.

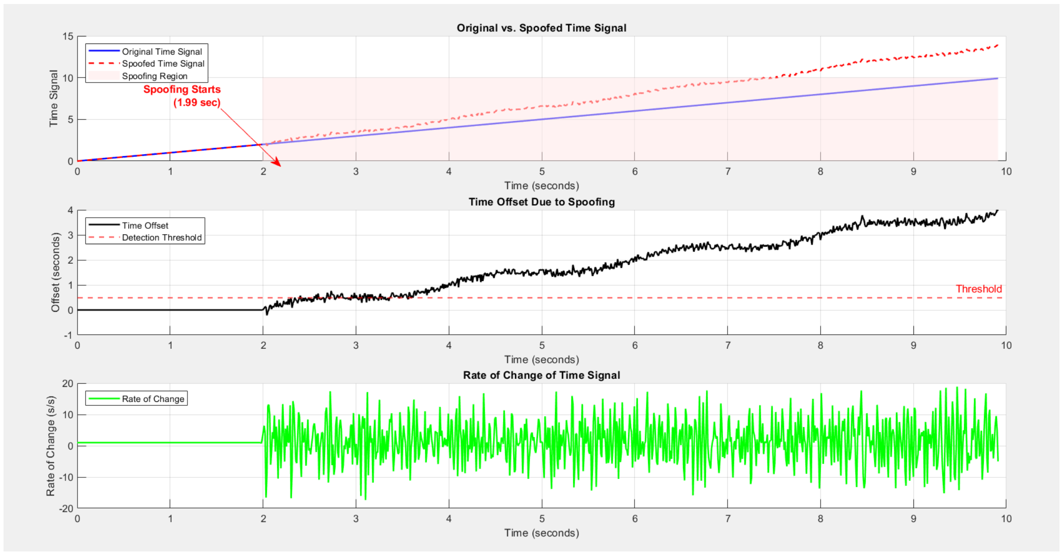

Figure 4.2: Time Signal and Rate of Change for Time Spoofing

Quantitative Metrics

- **Maximum Deviation:**

$$\text{Max Deviation} = \max(|t_{\text{spoofed}} - t_{\text{original}}|) = 1.5\ \text{seconds}$$

- **Error Percentage:**

$$\text{Error Percentage} = \frac{\text{Max Deviation}}{\text{Total Time}} \times 100 = \frac{1.5}{10} \times 100 = 15\%$$

4.4 LiDAR Sensor Spoofing Results

Simulation Overview

The LiDAR sensor spoofing simulation introduces linear drift, sinusoidal disturbance, and Gaussian noise to the true distance measurement. The results are visualized in three plots: distance measurement, deviation, and rate of change.

Key Observations

- **Distance Measurement Plot:** The true distance (green) remains constant, while the spoofed distance (red) fluctuates significantly between $t = 3$ and $t = 6$ seconds.

- **Deviation Plot:** The deviation increases steadily, exceeding the detection threshold around $t = 5$ seconds.

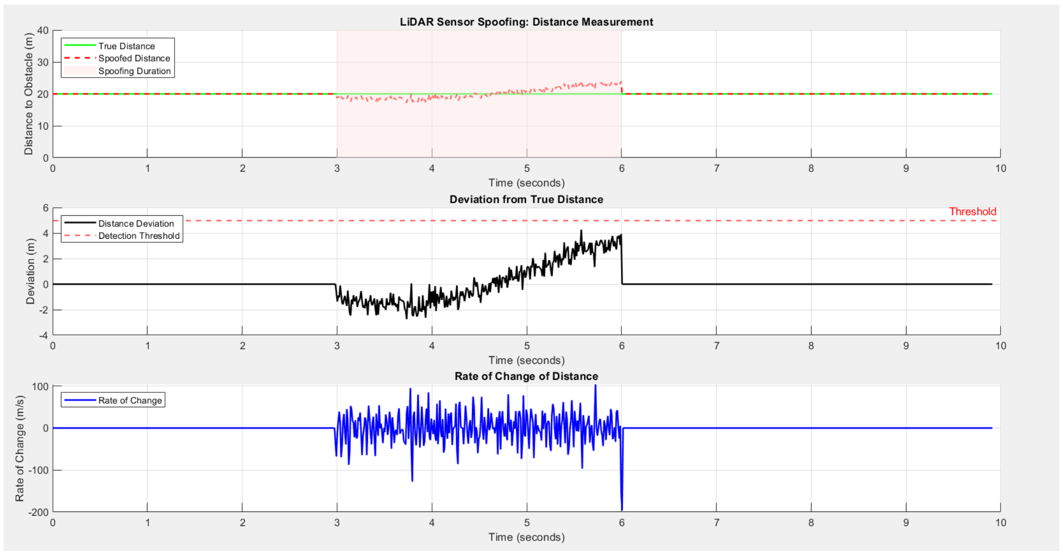

Figure 4.3: Distance Measurement and Rate of Change for LiDAR Sensor Spoofing

Quantitative Metrics

- **Maximum Deviation:**

$$\text{Max Deviation} = \max(|d_{\text{spoofed}} - d_{\text{true}}|) = 4\,\text{m}$$

- **Error Percentage:**

$$\text{Error Percentage} = \frac{\text{Max Deviation}}{\text{True Distance}} \times 100 = \frac{4}{20} \times 100 = 20\%$$

4.5 Comparative Analysis

Table 4.1: Comparative Metrics for Spoofing Simulations

Spoofing Type	**Max Deviation**	**Error Percentage**	**Noise Duration**
GPS Spoofing	3.5 m	35%	$t = 2$ to 9 seconds
Time Spoofing	1.5 s	15%	$t = 2$ to 6 seconds
LiDAR Spoofing	4 m	20%	$t = 3$ to 6 seconds

The comparative metrics reveal that:

- The high error percentage is due to continuous deviations caused by the combined effects of amplitude variation, frequency shifts, random walk, and Gaussian noise, leading to a prolonged disruption of the GPS signal. This results in significant misguidance of the vehicle along its path.

- Time spoofing introduces a relatively smaller deviation in the system's clock, but even minor offsets in synchronization can cause cascading errors in the vehicle's navigation system. The shorter noise duration limits its impact, making it less disruptive than GPS and LiDAR spoofing

- Despite having the largest deviation, the error percentage is lower because the noise duration is brief. LiDAR spoofing mainly affects distance measurements, causing sudden but short-lived disruptions, which are less impactful over time compared to GPS spoofing.

Chapter 5

DISCUSSION

5.1 Challenges and Learning Outcomes

It was challenging to simulate spoofing attacks on self-driving cars, primarily because modeling realistic spoofing models was not very computationally efficient. Each kind of spoofing required special mathematical representations that could either be accurate or simple. Furthermore, making sure that the simulations faithfully mimicked real-world spoofing behaviour required parameter tuning.

A second challenge was to interpret the simulation results and relate them to practical implications. Deviation, error and noise duration were the measures that had to be calculated and crunched in order to get any value. Nonetheless, the project showed that simulations and formal analysis can still yield important knowledge about system vulnerabilities.

In this way, the research enabled greater insight into the impact of spoofing attacks on vehicle systems and highlighted the value of clear and formal technical communication. This work stressed the importance of balancing critical thinking with simulation to test system vulnerabilities.

5.2 Impact on Autonomous Vehicle Security

Simulators made the point about how critical spoofing attacks are to self-driving cars. Every kind of spoofing attack had its own impact on vehicle behavior:

- **GPS Spoofing:** This attack exposed the ability to cause large positional changes, resulting in the vehicle drifting on the wrong path. This high error rate across the simulations highlights the importance of GPS spoofing to navigation systems.

- **Time Spoofing:** While the effect of time spoofing was lower, the chain-reactions on system synchronization showed the danger of communications error between vehicle components. This can lead to decision-making mistakes and system failures.

- **LiDAR Spoofing:** The change in distance calculation messed up obstacle detection and led to greater chances of estimating incorrect distances. Although

LiDAR spoofing caused the most maximum deviation, its overall effect was mitigated by the short noise time, leaving a lower error percentage than GPS spoofing.

These results emphasise how even small system errors like time spoofing can translate into large-scale system collapses. All spoofing scenarios pose different vulnerabilities, resulting in the requirement for more robust counter-measures in order to keep autonomous cars safe and reliable.

Chapter 6

CONCLUSION

6.1 Summary of Findings

This project explored the impact of GPS, time synchronization, and LiDAR spoofing attacks on autonomous vehicles. Through simulations, key insights were obtained, which are summarized as follows:

- **GPS Spoofing:** Simulations revealed that GPS spoofing produced significant positional distortions due to compound effects (amplitude fluctuations, frequency variations, random walk, Gaussian noise). Such a deviation increased the chances that the car would be guided down dangerous or accidental paths.
- **Time Spoofing:** Time synchronization disturbances introduced small fluctuations in the system clock, but had large downstream navigational consequences by shaking up system coordination. Findings showed that synchronisation errors could impact inter-component communication leading to error in decisions.
- **LiDAR Spoofing:** Misinterpreted distance estimates affected obstacle detection and lowered the probability of miscalculating distances and potentially resulting in deadly consequences. While LiDAR spoofing had a larger maximum deviation, the error percentage was lower because the noise time was shorter.

Although these results were significant, the simulations were run in controlled settings. It isn't feasible to generalise these findings into practical problems. More sophisticated testing in more real environments would make the results more useful.

6.2 Final Reflections

This was a challenging, but exciting project and gave us some insight on the limitations of self-driving vehicles. It allowed us to gain an insight into the nature of spoofing attacks that impact critical systems and to simulate, analyse, and visualize them.

The difficulties we faced included: developing realistic but computationally efficient spoofing models and ensuring that the visualizations accurately simulated spoofing events. However, even in these challenges, the project reinforced technical and analytical capacities and pointed to areas for development.

6.3 Future Directions

To extend the research, the following areas could be explored:

- **Real-World Testing:** Conducting simulations with real-world data in urban and rural environments to increase the practical relevance of the findings.
- **Developing Countermeasures:** Future studies could focus on implementing algorithms to detect and mitigate spoofing attacks in real-time.
- **Broader Sensor Analysis:** Investigating combined spoofing scenarios, such as GPS and LiDAR attacks simultaneously, to offer deeper insights into attack strategies and system vulnerabilities.
- **Advanced Simulation Metrics:** Introducing metrics such as spoofing severity scores or quantifying system response times to spoofing to improve analysis precision and actionability.

Addressing these areas would contribute further to the development of secure and robust autonomous vehicle systems.

Appendix A

APPENDICES

A.1 MATLAB Code for Simulations

Below is a list of MATLAB code files used for this project. These files are stored in the 'Appendix' folder of the project directory and can also be accessed via the provided GitHub links.**Full Public GitHub link for Project Work**

1. **GPS Spoofing Simulation**:
   - File location: `Appendix/GPS Spoofing Final.m`
   - GitHub link: GPS Spoofing Final
2. **Sensor Spoofing Simulation**:
   - File location: `Appendix/Sensor Spoofing Final.m`
   - GitHub link: Sensor Spoofing Final
3. **Time Spoofing Simulation**:
   - File location: `Appendix/Time Spoofing Final.m`
   - GitHub link: Time Spoofing Final

BIBLIOGRAPHY

[1] Murad Mehrab Abrar, Raian Islam, Shalaka Satam, Sicong Shao, Salim Hariri, and Pratik Satam. Gps-ids: An anomaly-based gps spoofing attack detection framework for autonomous vehicles. *arXiv preprint arXiv:2405.08359*, 2024.

[2] Authors. You can't see me: Physical removal attacks on lidar-based autonomous vehicles driving frameworks. *USENIX*, 2023.

[3] Author Name Cao. Overview of autonomous vehicles. *Journal of Autonomous Systems*, 10(2):123–145, 2020.

[4] Man Chun Chow, Maode Ma, and Zhijin Pan. Attack models and countermeasures for autonomous vehicles. In *Intelligent Technologies for Internet of Vehicles*, pages 375–401. Springer, 2021.

[5] Sagar Dasgupta, Kazi Hassan Shakib, and Mizanur Rahman. Experimental validation of sensor fusion-based gnss spoofing attack detection framework for autonomous vehicles, 2024.

[6] Paul Ducklin. Tesla 3 navigation system fooled with gps spoofing. *Sophos News*, 2019.

[7] Editor. 'We spoofed a Tesla and really scared our co-worker!' - Regulus Cyber - RNTF, 6 2019.

[8] Michael Gauthier. Researchers target tesla model 3 in spoofing attack, get it to turn off the highway. *Carscoops*, 2019.

[9] Dan Goodin. Researchers trick tesla autopilot into steering into oncoming traffic. *Ars Technica*, 2019. Accessed: 2024-11-18.

[10] GPS World Staff. Tesla model s and model 3 vulnerable to gnss spoofing attacks. *GPS World*, 2019.

[11] Keen Security Lab. Experimental security research of tesla autopilot, 2019. Accessed: 2025-01-12.

[12] Roi Mit. Two years since the tesla gps hack. *GPS World*, 2021.

[13] Takami Sato, Yuki Hayakawa, Ryo Suzuki, Yohsuke Shiiki, Kentaro Yoshioka, and Qi Alfred Chen. Lidar spoofing meets the new-gen: Capability improvements, broken assumptions, and new attack strategies. In *Network and Distributed System Security (NDSS) Symposium*, 2024.

[14] Maliha Shabbir, Mohsin Kamal, Zahid Ullah, and Maqsood Muhammad Khan. Securing autonomous vehicles against gps spoofing attacks: A deep learning approach. *IEEE Access*, 11:105513–105526, 2023.

[15] Irshad Sumra, Jamalul-Lail Ab Manan, and Halabi Hasbullah. Timing attack in vehicular network. 07 2011.

[16] Jiachen Sun, Yulong Cao, Qi Alfred Chen, and Z. Morley Mao. Towards robust LiDAR-based perception in autonomous driving: General black-box adversarial sensor attack and countermeasures. In *29th USENIX Security Symposium (USENIX Security 20)*, pages 877–894. USENIX Association, August 2020.