

UNIVERSITY OF PLYMOUTH

**Investigating Privacy and Security of Cloud-Connected
Autonomous Vehicles**

By

Sahand B. Murad

A thesis submitted to the University of Plymouth
in partial fulfilment for the degree of

Research Masters

School of Computing, Electronics, and
Mathematics

March 2020

Copyright Statement

This copy of the thesis has been supplied on condition that anyone who consults it is understood to recognise that its copyright rests with its author and that no quotation from the thesis and no information derived from it may be published without the author's prior consent.

Acknowledgment

I wish to extend my warmest thanks and appreciation to those who have helped me during my thesis work.

Many thanks to Dr. Stavros Shiaeles, for his continued support and guidance during my research work, it was not possible without his guidance and inspiration from the start to the end of this Research degree.

Dr. Asiya Khan and Dr. Giovanni Masala thank you very much for your help in every step, following and guiding me during every step of my test, and testbed setup.

My family, for their continued support during my journey, continues love and always believing in me, my father for making this a living reality for me instead of a dream, without them none of this would have been even possible.

I would like to thank Mr. Charlie Day, Mr Liam McEachen, Mr. Stuart MacVeigh, Mr. John Welsh and Dr. Toby Whitley for their support in building the autonomous vehicle and general electronics.

Declaration

At no time during the registration for the degree of Doctor of Philosophy has the author been registered for any other University award without prior agreement of the Doctoral College Quality Sub-Committee

This is to certify that the candidate, Sahand Barzan Murad Murad carried out the work submitted herewith.

Work submitted for this research degree at the University of Plymouth has not formed part of any other degree either at the University of Plymouth or at another establishment.

The author would like to thank Mr. Charlie Day, Mr Liam McEachen, Mr. Stuart MacVeigh, Mr. John Welsh and Dr. Toby Whitley for their support in building the autonomous vehicle and general electronics.

Word count of main body of thesis: 15996

Candidate's Signature:

Sahand Barzan Murad Murad

Sahand

Date: March 2020

Publications

Accepted and published paper in 2019 IEEE World Congress on Services (SERVICES) conference under 'Data Encryption and Fragmentation in Autonomous Vehicles using Raspberry Pi3' title. Attached to the thesis as appendix 1.

Murad, S. Shiaeles, S. Khan, A. and G. Masala (2019). Data Encryption and Fragmentation in Autonomous Vehicles Using Raspberry Pi 3. 2019 IEEE World Congress on Services (SERVICES), IEEE.

Vol 2642 pp 2012-2016

Sahand Murad

Investigating Privacy and Security of Cloud-Connected Autonomous Vehicles

Abstract

Autonomous cars are intelligent systems that can do Physical tasks without human interaction and are used in Industrial environments, transport, and the military, one of most powerful feature of this technology is that possess intelligent agents that can learn from their environment, furthermore, they have several sensors with connectivity between them. Nowadays most of the car manufacturer use autonomous features like lane-keeping, Adaptive Cruise Control (ACC), advance driver assistance system and automatic parking system resulting in a rapid increase in research of autonomous vehicles e.g. in 2004 and 2005 DARPA challenges for vehicles to autonomously navigate via desert terrain, moreover, the DARPA challenge in 2007 developed and tested cars that independently explored via a mock urban condition amid traffic.

Vehicles have huge potential in improving road safety, providing convenience; reducing emissions and congestion by communicating with another vehicle within the same network furthermore, in case of emergency they can also notify other vehicles of the incident. Much architecture for communication between vehicles is centralized, typically using cloud servers. The security and trust of that communication are paramount.

Therefore, this research aimed to propose a novel method that can insure data security in the cloud by encrypting and fragmenting data to increase the uncertainty for an attacker so as a result, it becomes difficult for hackers to compromise the confidentiality and integrity of data residing in the cloud. This research presents experimental results in terms of time, CPU utilization and size which allowed to determine the most effective method for securing data in the cloud and hence making it difficult for a hacker to reconstruct data.

Splitting and encrypting different size of video and text file or encrypting the whole file shows that less time, CPU usage and size is taken in splitting and encrypting 5KB rather than other sizes or encrypting the whole file, so it saves CPU utilization, time and storage, hence, it is the ideal size as it minimizes the CPU resources and memory as compared to different size

fragments. The privacy of data is at a higher level preventing a hacker from accessing the data as it is shared in multiple clouds, furthermore, the proposed technique also proposed a mechanism which ensures the data integrity and confidentiality by encrypting the data header hence making it almost impossible for hacker to reconstruct the original data even if it been hacked by man in middle attack. Finally, the experimental results shows that this method can overcome the issue of overhead in transmission and as a result, makes it an efficient and effective mechanism to encounter the data security problem.

Contents

List of abbreviation.....	
1 Chapter Introduction	1
1.1 Introduction.....	1
1.2 The motivation of the study	1
1.3 Aims of the project.....	9
1.4 Research questions	9
1.5 Contributions and Structure of the thesis	9
2 Chapter Background	11
2.1 Background.....	11
2.2 Raspberry pi3.....	14
2.3 Data Security.....	16
2.4 Data at Rest	17
2.5 Data Fragmentation	19
2.6 Data Encryption	22
2.7 Public Key Encryption.....	23
2.8 Identity-Based Encryption (IBE)	23
2.9 Attribute-Based Encryption (ABE).....	23
2.10 Fuzzy identity-based encryption	23
3 Chapter Proposed Method.....	25
3.1 Proposed method	25
3.2 Experimental Methodology.....	26
3.3 Connecting the Raspberry Pi 3.....	27
3.4 Connecting Ultrasonic Sensors to the Raspberry Pi 3	28
3.5 Software used for implementation.....	28
3.6 Data encryption and fragmentation	28
3.6.1 AES 256 CBC Encryption.....	29
3.6.2 RSA Encryption	31
4 Chapter Results and discussion	33
4.1 Results and Discussion	33
4.1.1 Data collection phase	33
4.1.2 Data analysis phase	33
4.1.3 Data encryption phase.....	34
4.2 Comparing Symmetric with asymmetric encryption	38
5 Chapter 5.....	42

5.1	Conclusion and Future work.....	42
6	Acknowledgments	43
	References	44

List of figures

Figure 1-1 [1]	1
Figure 1-2 Single-hop Example.	3
Figure 1-3 architecture of autonomous cars.....	4
Figure 1-4 Tracking a rural road at night using a multi-layer	5
Figure 1-5 Autonomous Vehicle Attack Taxonomy.....	6
Figure 1-6 [39].....	8
Figure 2-1 Random pattern fragmentation	12
Figure 2-2 Raspberry Pi 3 Model B	13
Connecting and communicating with Raspberry Pi Figure 2-3.....	15
Ultrasonic sensors Figure 2-4	16
Figure 2-5 The adversary gains access to the service center network.....	18
Figure 2-6 Architecture of Cloud Stash and baseline in uploading operation.....	20
Figure 2-7 Proposed model that uses random pattern fragmentation and stores the random chunks in split files, which are then stored on a NoSQL database [60].....	21
Figure 2-8 The system model of fuzzy identity-based data integrity auditing protocol	23
Figure 3-1 Block diagram of the autonomous vehicle and Data Collection	25
<u>Figure 3-2 Autonomous vehicle</u>	<u>26</u>
Figure 3-3 Data encryption and fragmentation process	28
Figure 3-4 Time and CPU usage for fragmenting video and text files.....	29
Figure 3-6 Time and CPU usage for encrypting video files(AES).....	30
Figure 3-6 Time and CPU usage for encrypting text files(AES)	30
Figure 3-6 Time and CPU usage for encrypting video files(RSA)	31
Figure 3-6 Time and CPU usage for encrypting text files(RSA).....	32
Figure 4-1 Process diagram	34
Figure 4-2 CPU utilization for encrypting different sizes of video and text files (AES).....	36
Figure 4-3 Time for encrypting different sizes of video and text file (AES).....	36
Figure 4-4 Time for encrypting different sizes of video and text files (RSA).....	37
Figure 4-5 CPU utilization for encrypting different sizes of video and text files (RSA).	38
Figure 4.6 Total time to encrypt and transmit Video and text file to cloud.....	41
Figure 4.7 Total time to encrypt and transmit Video and text file to cloud.....	41

List of abbreviation

NHTSA	National highway traffic safety authority
VANET	Vehicular ad hoc network
RSU	Roadside units
OBU	Onboard units
ACC	Adaptive cruise control
DARPA	Defence advanced research projects agency
LIDAR	Light detection and ranging
GPS	Global positioning system
MOD	Mobility on demand
COG	Centre of graphic
AVs	Autonomous vehicles
DNN	Deep neural networks
AI	Artificial intelligence
LAN	Local area network
CPU	Central processing unit
GUI	Graphic user interface
RAM	Random access memory
GPIO	General Purpose Input/output
DSRC	Dedicated short-range communication
ECU	Engine control unit
EDR	Endpoint detecting and response
AFS	Advanced encryption standard

RAM	Random access memory
KEK	Key encryption key
DAC	Digital to Analog converter

1 Chapter Introduction

1.1 Introduction

This chapter gives a brief introduction of the autonomous vehicles along with its benefits, furthermore, a detailed discussion on the working of autonomous vehicles were also covered in this chapter. The remaining of this chapter is organized as follows: Section 1.2 explains the motivation of the study, the aim of the project is presented in section 1.3, section 1.4 explains the research questions of the project and structure of the thesis is presented in section 1.5

1.2 The motivation of the study

The concept of autonomous driving emerged in the 1980s, with the invention of Navlab autonomous vehicles made by Carnegie Mellon University which was able to operate in structured environments [1], and the University of the Bundeswehr Munich also showed promising results by inventing the high-speed motorway autonomous vehicle [2]. Even nowadays, most of the features of modern autonomous vehicles are taken from this technology e.g lane detection system which is used by autonomous cars to give lane departure, driving assistance systems, adaptive cruise system used for detection and tracking of vehicles driving ahead to keep a reasonable, safe distance, pre-crash system used to trigger braking power to reduce the damage in case if a driver fails to react on time. Furthermore,

Exhibit 3 - Fatality and Injury Rates per Population and Vehicle Miles Traveled, 1999-2009

Year	Fatalities	Killed			
		Resident Population (Thousands)	Fatality Rate per 100,000 Population	Vehicle Miles Traveled (Billions)	Fatality Rate per 100 Million VMT
1999	41,717	272,691	15.30	2,690	1.55
2000	41,945	282,172	14.87	2,747	1.53
2001	42,196	285,082	14.80	2,796	1.51
2002	43,005	287,804	14.94	2,856	1.51
2003	42,884	290,326	14.77	2,890	1.48
2004	42,836	293,046	14.62	2,965	1.44
2005	43,510	295,753	14.71	2,989	1.46
2006	42,708	298,593	14.30	3,014	1.42
2007	41,259	301,580	13.68	3,031	1.36
2008	37,423	304,375	12.30	2,977	1.26
2009	33,808	307,007	11.01	2,954	1.14

Figure 1-1 [1]

they are embedded with an intelligent system that not only can learn from their environment

but also make them capable of doing physical tasks without human interaction. Nowadays these types of vehicles are widely used in an industrial environment, transport, and military. In the United States alone, approximately 33,000 people are killed in car accidents every year, according to a National Highway Traffic Safety Authority (NHTSA) study [1].

In 2011, the U.S. Chamber of Commerce released a Transportation Performance Index report stating that the performance of America's regional transportation system improved by only six percent, while the nation's passenger traffic increased by 39 percent from 2003 to 2008 [2]. The Ad hoc Vehicle Network (VANET) has a great deal to say for the world today. Such a form of the network allows inter-vehicle connectivity and contact between vehicles and different roadside units (RSU). A standard car simply has to be fitted with an onboard system that includes the connectivity capabilities necessary to make it a networked vehicle. VANETs allow information exchange and sharing between vehicles or between vehicles and RSUs, thereby improving the overall driving experience. VANETs have high mobility and have a dynamic topology. VANET comprises of roadside and wireless communication service equipment. A VANET is a complex array of networked vehicles that connect on a dedicated short-range communication over 5.9 GHz According to the Dedicated Short Range Communications (DSRC) protocol, each vehicle in a VANET broadcasts a traffic safety message every 100-300 ms, which keeps the vehicle's driving related information, such as location, speed, turning intention, and driving status. Cloud could be anywhere and an attacker could intercept data from anywhere. In VANET should be in close distance. Also encryption in cloud could be very strong as you have the processing power Encryption is fundamental and crucial in protecting vehicular data transmission. Through encryption, the confidentiality of the data transmission can be assured. Depending on the encryption scheme used, it may also be possible to rely on the keys to verify the identity of the sender. It is also necessary to have message authentication code (MAC) algorithms in place to protect and verify the integrity of the data being received. The method with each other or with neighboring RSUs [3]. Such vehicles are fitted with on-board wireless units (OBUs) as shown in Figure 1.1 to carry out this communication [4].

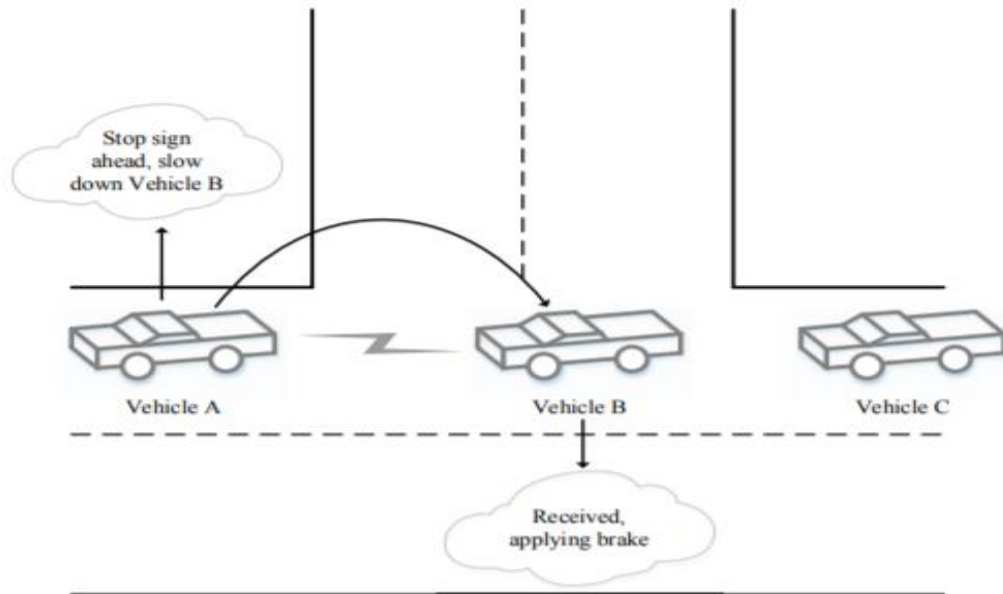


Figure 1-2 Single-hop Example.

VANET has immense significance these days and these VANETS could be created by regular vehicles as well as autonomous vehicles. Moreover, VANETs have high mobility and have a dynamic topology. In order to research on autonomous vehicles, Induct Technology created Navia, which is a robotically driven electric shuttle, with a maximum speed of 20 Km/h. The shuttle has an optical camera and four Lidar sensors on it and has been tested in many universities in England, Singapore and Switzerland [5]. In each car, it is possible to install many kinds of sensors e.g. LIDAR, GPS, Camera sensor, which can perform specialized tasks. In the last decade, driverless vehicles have been piloted on the roads. Besides, advanced driver assistance systems and autonomous vehicles continue to increase rapidly, e.g. the 2004 and 2005 DARPA challenges for vehicles to autonomously navigate via desert terrain. It is significant to note that autonomous cars have sensors with connectivity between them. Most cars manufactured now use some autonomous features such as lane-keeping, Adaptive Cruise Control (ACC). Further, the DARPA challenge in 2007 developed and tested cars that independently explored via a mock urban condition amid traffic, have created significant excitement and research enthusiasm for the field of autonomous driving [6],[7]. The design of autonomous driving car technology in many ways has been carried out. The self-driving work was carried out in several areas: in actual implementation [8] as shown in figure 1-2 or a scaled system [9]. The urban population is expected to double over the next 30 years [13] according to the UN estimate so the number of cars will also increase.

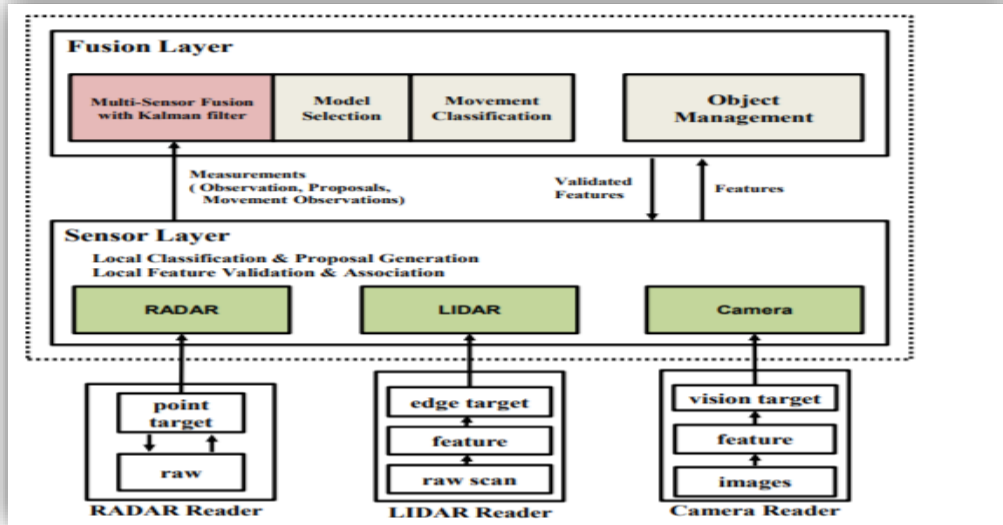


Figure 1-3 architecture of autonomous cars

Also, different methods of pattern recognition image processing are available, such as filtering images, harsh methods and tracking references [11],[12]. The urban population is expected to double over the next 30 years [13] according to the UN estimate with restricted additional roads and parking spaces accessible in existing (mega) towns, private cars seem to be impractical for the prospect of personal urban mobility [14]. Perhaps, the one-way car sharing with the electric cars (called the Mobility on demand, or MOD) that directly target parking space, congestion, and low vehicle use issues [14] is one of the most promising solutions to addressing this issue. Recent operations in several European and American cities have included limited-size MOD projects utilizing human-driven vehicles [15]. Nevertheless, these schemes contribute to imbalances in vehicles that are easily exhausted at some stations while others are unnecessary because some stations are more common than others. It is surprising to note that even if the transport network is symmetrical (i.e. the base network topology is a regular network grid and arrival rates and routing options for clients at all stations are uniformed), the stochastic nature of customers arriving at stations quickly disrupts the system and thus results in instability [16]. In increasingly popular biking systems, the problem of re-equilibrium is solved by employing trucks that can transport many bikes simultaneously, with algorithms designed very recently to optimize truck routes [17],[18]. Because this solution for vehicles is not practical [19], consider the possibility of recruiting a squad of drivers re-equilibrating cars throughout the transport network. This strategy, therefore, unbalances the re-equilibrium drivers themselves and involves 'restructuring of the re-equilibrium vehicles,' which greatly increases congestion and

costs [19]. Another means of incentivizing riding sharing [20] is, though, the goal of the MOD's program to guarantee personal mobility. A convergence platform has recently been proposed in [21],[22] which provides consumers with on-demand connectivity in driverless electric cars. Self-governing driving offers MOD systems great prospects, given that robotic vehicles can rebalance (eliminating the equalizing problem), allow system-wide coordination, free passengers from driving and possibly increase safety. In effect autonomous cars specially designed for personal urban mobility (e.g., the Induct Navia vehicle [23], the General Motors Vehicle EN-V [24], Google Motor vehicle [25]) are already being tested; nevertheless, little is understood about how robotic transport networks are planned and operated [21]. Several approaches, including the use of marked and unmarked lanes [26],[27] were established concerning driving guidance. The use of the guideline mark is common; for example, in [28], [29].

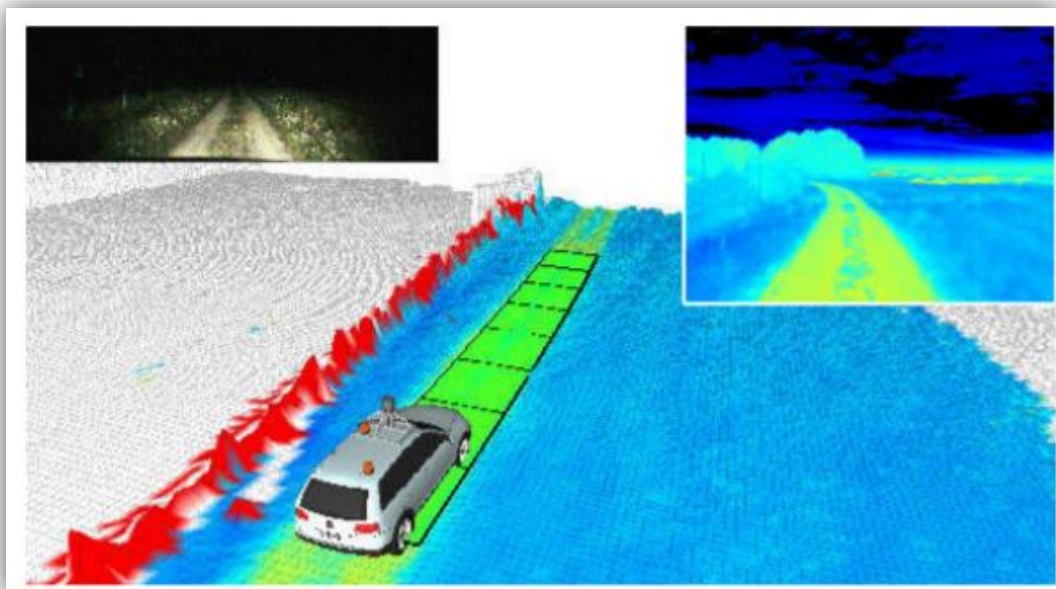


Figure 1-4 Tracking a rural road at night using a multi-layer

Terrain map (thermal layer is shown). Red cells indicate obstacle cells. Upper left: Color

Night Vision camera, Upper right: a thermal camera [28]

The use of the guideline mark is common; for example, in [28],[29]. In [29], a camera processing image was used to detect the road mark. The Intel T5750 2.0 GHz Clock Speed optimized for single operation requires less than 14 Ms. However, in adverse turning circumstances they faced obstacles in determining the boundary line. Although a diameter or a set of lines could be overcome by using a rough lane border, the rotation of the road was

difficult to detect because they used the transformation of Hough. For the powerful aspect, expense and efficiency according to [10] the use of a mini picture machine are favoured. To this purpose, in this research self-driving investigations that can recognize the road or street mark route was carried out. A model for a 1:10 scale auto system with camera image processing in Raspberry Pi 2 was proposed. This work aims at using the process of detecting street marks for the concept drive system. The coordinate of the detector region of the filtered picture outcome acquired through a single camera by COG (Centre of Graphics), as used in [29]. This article does not dwell on the lighting effect [12] but defines the HSV philtre parameters only for detecting street markings. Autonomous driving holds incredible guarantee for MOD frameworks because robotic cars can rebalance themselves (in this manner eliminating the rebalancing issue at its centre), enable framework wide coordination, free travellers from the assignment of driving, and conceivably increment safety. The increased connectivity and interaction of an autonomous vehicle give rise to vulnerabilities in data integrity. A single-vehicle compromised can lead to security hazards affecting multiple vehicles. Hence, it is important to design robust systems that ensure data integrity and will be very difficult for attacker to get data. There are many ways to protect data, such as hiding identities of communication parties; however, they still let them authenticate each other, encryption and authentication Data encryption offers some security. Therefore, to make data secure from hackers it is better to encrypt data before sending data to the cloud to protect the data from the attackers; data is encrypted using a key verified by the identity of the sender [30] as shown in figure 1-5.

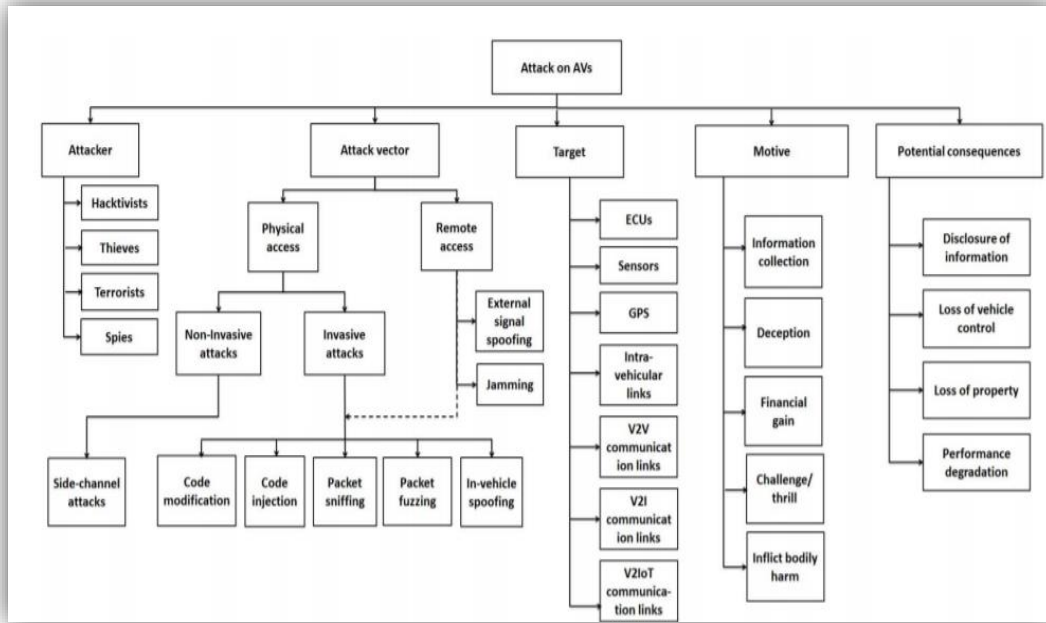


Figure 1-5 Autonomous Vehicle Attack Taxonomy

A single-vehicle compromised can lead to security hazards affecting multiple vehicles so the data integrity and protection must be ensured. There are many ways to protect data, such as hiding identities of communication parties which can be used to secure data and eventually provide assistance in protecting vehicle security [31]. Moreover, as managing and protecting key is a problem because of numerous security hazards which could lead to the hacking of the system, therefore, authors in [32] propose a cloud stash, which splits the files to multi shares, and offer to send it to multiple clouds where the files are reconstructed. Authors in [33] propose a threshold scheme to divide the data into several pieces in such a way that the individual pieces do not reveal any information about the data. Another way to keep the data safe from hackers is symmetric and asymmetric encryption, in symmetric encryption sender and the receiver use the same key to encrypt and decrypt data whereas in asymmetric, each user has its public and private keys whereas the public key is used for data encryption and it's corresponding is used to encrypt data one key to encrypt and decrypt the data [34]. The transport infrastructure and architecture are an important consideration in all smart cities. There has been a taxi ride in Singapore recently [35] for autonomous vehicles (AVs). More vehicles in many other countries are expected to start roaming on the roads soon. Today, opponents are becoming ever more competitive. Autonomous vehicles have more network connectivity and sensors than current cars to increase safety, but this amount of sensors and connectivity brings new vulnerabilities, growing the AV's attack surface.

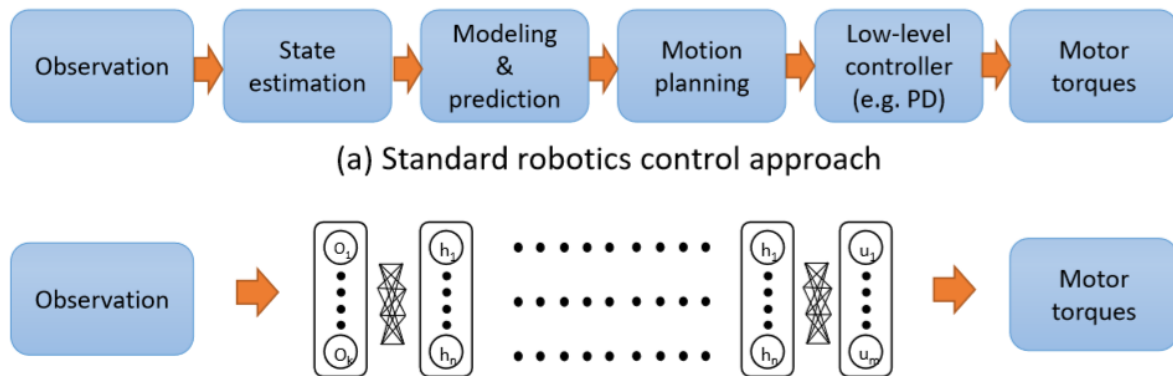
The mix of ability and efficient low-cost offensive equipment will easily break through vehicle safety systems. The latest ways of targeting an AV, by linking to a certain port or by cable harnessing the vehicle network, must be brought to the notice of stakeholders. Therefore, it is vital to find out which defensive strategies are most successful in combating attacks. The sector of robotic vehicles includes user mechatronics, artificial intelligence and multi-agent systems (multi-agent systems) to assist vehicle operation. The scope of control devices technologies has been expanded since the development of the integrated circuit.

The previously mentioned features enable vehicles to be categorized as smart or intelligent. Besides this, semi-automatic vehicles also exist which uses automation for difficult processes, especially for navigation. Manufacturers and researchers also introduced a range of automatic features, especially in automobiles and other equipment [36]. In general, when it is capable of recognizing the environment and moving within it, without human intervention, a car is characterized as autonomous. By using technologies such as radar, LIDAR (Light Detection And Ranging), GPS, Odometry and Computer Vision, autonomous vehicles detect their environment.

Advanced control systems are reading and processing information from sensor readings to define correct travel routes, hazards, and markings. Autonomous cars have control systems that can interpret sensory data to distinguish between various cars on the road, which is very helpful when designing a path to the desired destination. The car will also have the option of self-governance under the word 'autonomously.' While these machines are referred to as robotic or autonomous, based solely on automation systems.

Most recent autonomy-related research ventures are in reality programmed (they are designed to be automatic) rely heavily on artificial elements in their system, such as magnetic tapes. Under significant environmental uncertainty, autonomous control requires good efficiency over long periods and the capacity to account for possible system damage without external intervention. Several documented studies have suggested expanding the possibilities of an autonomous car across communication networks with both neighbouring and long (congestion management) vehicles as it can be seen from several projects. In the decision-making process, adding two more external influences, some now consider the car's behaviour or potential as autonomous [37]. In recent years, autonomous cars have been a subject of growing interest as many businesses are aggressively evolving similar hardware and software technology for fully autonomous driving capability without human intervention. Throughout recent years, deep neural networks (DNNs) have been used extensively throughout various tasks of vision and regulation. For autonomous vehicles as well, they are important workloads. Tesla Model S, for

example, was known to use a specialized chip (MobileEye Eye), which used a DNN-based real-time vision-based obstacle detection system. More recently, researchers are studying end-to-end real-time robotics control based on DNN [38], [39]. Further workloads focused on artificial intelligence (AI) based on DNN are likely to be used in potential autonomous vehicles as shown in Figure 1-6. There are several additional challenges to performing such AI workloads on an embedded computing network.



Standard robotics control vs. DNN based end-to-end control. Adopted from

Figure 1-6

First, many AI workloads are computationally demanding and have strict real-time requirements, particularly those in automobiles. Computing latency in a vision-based object detection mission, for example, can be directly linked to vehicle safety. This requires both a high computing capacity and the means to ensure the timing. The computational hardware architecture, on the other side, must also satisfy the expense, scale, weight and power constraints that demand a highly efficient computing platform. As noted in [40], these two contradictory criteria confuse the method of platform discovery.

Therefore, the main contribution of this research is to propose a scheme to protect the anonymity of the vehicular data, thus, increasing the uncertainty for an attacker to access the data on the cloud. Vehicular data is collected from an autonomous vehicle with ultrasound sensors and a camera on a Raspberry Pi 3 microcontroller [41].

Autonomous driving holds incredible guarantee for MOD frameworks because robotic cars can rebalance themselves (in this manner eliminating the rebalancing issue at its center), enable framework wide coordination, free travelers from the assignment of driving, and conceivably increment safety. The increased connectivity and interaction of an autonomous vehicle give rise to vulnerabilities in data integrity. Hence, it is important to design robust systems that

ensure data integrity and are very difficult to be hacked. Data encryption offers some security. Therefore, to make data secure from hackers it is better to encrypt data before sending data to the cloud to protect the data from the attackers; data is encrypted using a key verified by the identity of the sender, so a need exists for designing secure and efficient method for data security of data transmitted by autonomous vehicles to different cloud providers.

1.3 Aims of the project

The aims of this research project are as follows:

1. Set up an autonomous vehicle and collect data from sensors
2. Define best practices for data anonymity and performance through literature review
3. Propose and test a new method to protect data from tampering and at the same time be lightweight to be implemented on autonomous vehicles.

1.4 Research questions

1. What data an autonomous vehicle has?
2. What is the size of these data and frequency?
3. What is the optimal size of the file that should be used to apply encryption?
4. What kind of encryption can be applied in order not to overkill CPU?
5. How to achieve the anonymity of the data and make it hacker-proof?

1.5 Contributions and Structure of the thesis

This thesis presented a concept of an efficient method to cloud provider for data encrypting and fragmenting so that they can protect the anonymity and increase the uncertainty for an attacker having access to the data on cloud, furthermore, the experimental results show that splitting and encryption of 5 KB is the ideal size and it is more efficient in terms of CPU and memory utilization as compared to different sizes of data fragmentation.

The remainder of the thesis is structured as follows:

Chapter 2 provides an outline of the current literature; it shows Raspberry pi 3, Data security, Data fragmentation, and Data encryption.

Chapter 3 devoted to the approaches and the research methodology used by the author for carrying out the experiments in the research.

Chapters 4 and 5 present the results of the tests that have been carried out for approaches, discussion and analysing the results. Finally, the conclusions and future work are presented in Chapter 5 as well as the limitation of the work and how to improve it for future work.

2 Chapter Background

2.1 Background

The swathe of next-generation V2V (vehicle-to-vehicle) communication systems had shown amazing improvement regarding the safety of autonomous vehicles and according to the US Government, they would be able to reduce about 80% of road accidents, and as a result, they would be able to save large numbers of human lives which was estimated to be 1.24 million worldwide every year.

According to Gartner by 2020, there will be 150 million connected cars on the road and most of them will share data via the internet, as these are programmed and controlled by software program, therefore, are immune to different types of attacks and so on, lots of hacks were performed exploiting the vulnerabilities in this software. In the last few years, there is an exponential increase in the vehicle automation industry, nowadays almost all mechanical components in a modern car are controlled by ECU known as electronic control units, these ECU are small computing unit which is used to control different aspects of the functioning of the mechanical part, furthermore, they are also used for managing brake, for transmission and the locking system, etc., moreover, these small ECU units are connected via CAN bus network to communicate with each other. For the last two decades, external access to the networks was done by using On-Board Diagnostics socket, which is embedded under the driving seat of the driver, but nowadays the internal networks are accessible using mobile telephone connections which also allows services like call, rescue services in case of a serious accident. Charlie Miller and Chris Valasek, American hackers exploit the vulnerabilities so that they can gain remote access to internal networks of a car Chrysler-Jeep, as a result, they were able to control different functions of the car e.g. the engine, brakes, ventilation, later on, Kevin Mahaffey and Marc Rogers another group of hacker claimed that they hacked a Tesla Model S.8.

One of the most eye-catching this year was when Fiat Chrysler was forced to patch 1.4 million jeeps which were exploited by a zero-day flaw in the SUV's entertainment system to seize control of a target Jeep Cherokee, furthermore, the Tesla Models were hacked by the White hats hackers, moreover, in July 2015, American hackers Mahassey and Rogers claimed they have hacked a Jeep Cherokee by hacking the vital control function of these cars using mobile networks. The reason behind the growing number of hacks and attacks on this V2V communication is because of the reason that they are heavily dependent on wireless networks

which can be compromised by exploiting the vulnerabilities. Therewith, cybersecurity has become an issue for automobiles and much attention is been required to secure vital digital infrastructures underlying this automobile.

There have been several ways in which data protection from hackers has been presented in the literature. Encryption and fragmentation techniques offer secure ways to protect data from hackers. Researchers in [6] proposed a cloud stash to split data to multi shares, where threshold shares are required to reconstruct the file. They used multithreading to manage secret sharing into multiple clouds. Therefore, when the client tries to download data and it is corrupted, cloud stash can reconstruct a new one from another cloud. To protect and make data more secure from attackers, work presented in [10] encrypted the data before sending it to the cloud to make it more secure. Only the clients had access to the decryption keys making it difficult for the attackers to get the data. The work presented in [11] [15] proposed a new technique to fragment the original file to chunks, they used random pattern fragmentation for splitting data, each split file had the same length as the associated pattern. Moreover, the aim of using this method is to protect the data from attackers, as the attackers do not know the information about the length of each split data, therefore, they cannot get the data as shown in Figure 2-1.

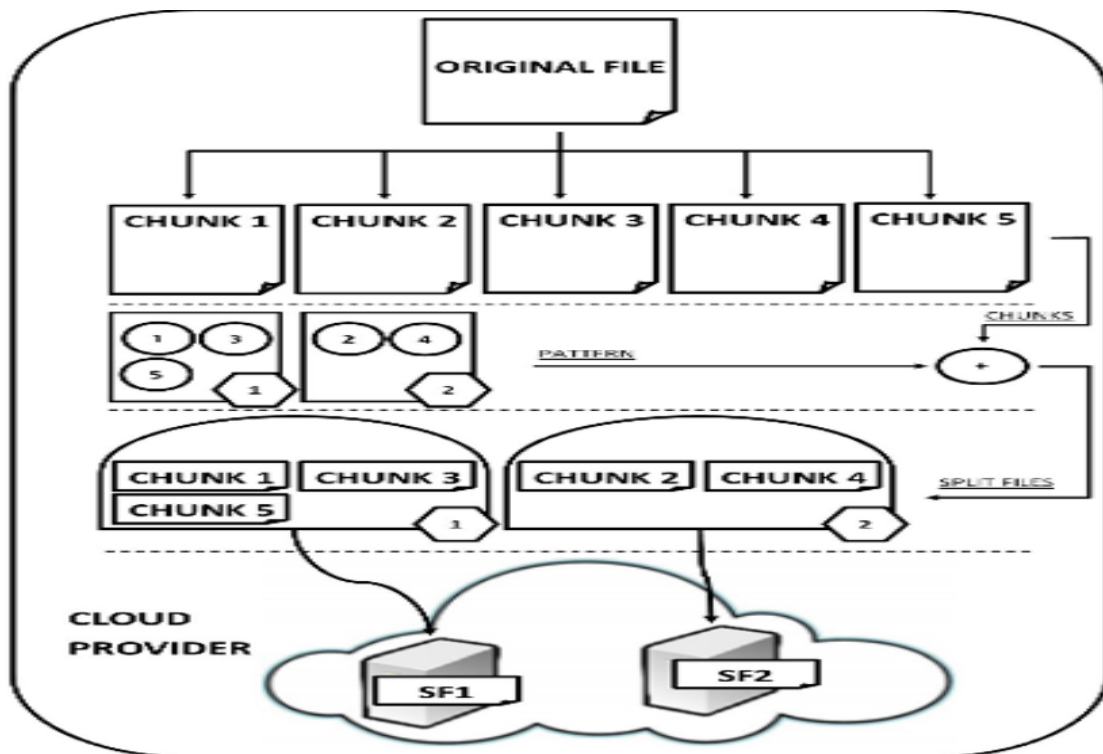


Figure 2-1 Random pattern fragmentation.

Authors in [15] present a new approach for data fragmentation combined with AES 256 CBC to encrypt the data and split to multiple chunks, and the chunks were sorted to split files by using a random pattern fragmentation. In [4] authors propose a taxonomy of defense against attacks on autonomous vehicles to enable the targeted defense to be developed. According to [12] to make the data more secure they proposed to split the data, then after splitting the data, the header was stored in another file of smaller size, they then added some bytes to the header making it harder to be attacked as it appears to be like other header files. This will make it harder for the attacker to find the header because when attacker attacks the data inside the cloud, they cannot find the header because all the split files have the same size, as it shows that the header was not sent to the cloud, so the attackers cannot defragment the split data without headers. Moreover, to make data safer from the eyes of the attackers, authors in [13] proposed a way to encrypt data by using the master key and symmetric encryption. For each authorized client key-encryption key (KEK) obtained via passphrase and also by using KEK the master key has been encrypted, after the data encrypted each client the master key and client identifier encrypted depending on user key-encryption key. For decrypting the data, the user enters a passphrase to the client who obtains the KEK, so the client decrypts the master key and data. From the literature, it can be concluded that many researchers have worked on autonomous cars data splitting data or encrypting data and splitting headers, but no one has tried to split and encrypt the headers of different files.

2.2 Raspberry Pi 3

The Raspberry Pi is a single-board device with a size of credit card that was launched commercially in 2012 as shown in Figure 2-2. The Raspberry Pi needs only a display and a keyboard to be inserted. With a Quad-Core 1.2 GHz BCM2837 64-bit CPU processor, 1 GB



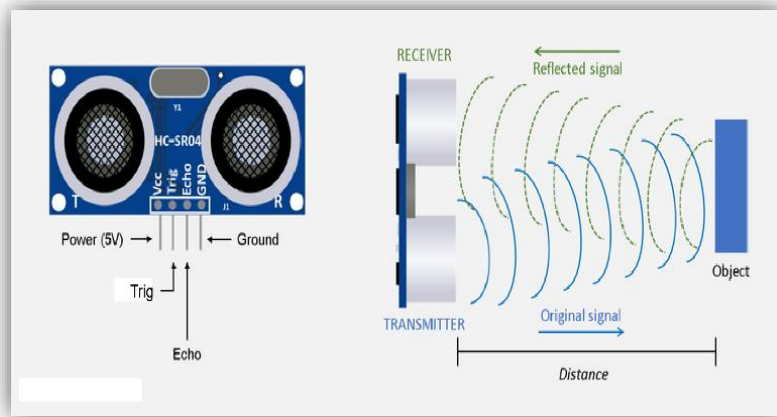
Figure 2-2 Raspberry Pi 3 Model B

RAM and built-in wireless LAN and Bluetooth, the Raspberry Pi 3 Model B is the most advanced model as of May 2018 [42], A Raspberry Pi's cost is around £ 32.00, so compared to its competitors, it's a relatively cheap computer.

Unlike other microcontrollers, Pi has the advantage of being a device that can be used as a microcontroller and thus a graphical user interface (GUI) allows for more realistic navigation around the Pi system than simple micro controls. Another computer can connect to Pi in real-time, to export commands or to receive Pi data from it, such as motor speed or the distance to an obstacle, using the integrated module. The Pi can be powered by a 5V battery pack, which can be used to make the system fully mobile. The Raspberry Pi comes with built-in compilers and can work in many languages like Python, Java, C, C++, so users are less restricted and can use their preferred language. The Raspberry Pi 3 Model B has a 600-700MHz idle clock speed but can achieve a maximum speed of 1.2GHz until overclocking. The Raspberry Pi's drawbacks, though, involve weighing more than the Arduino (an alternate, lighter microcontroller) and getting fewer RAM and GPIO PINs than the Udoo (Quad) [43]. Pi's research studies have shown that 32-bit operating systems like Microsoft Windows cannot be used [44] forcing users to use more complex Linux OS that is not widely used by the public through strong. Gurjashan Singh Pannu et al, [45] proposed that the overview is "Development and deployment of autonomy cars using Raspberry Pi," algorithms like the identification of lanes (two methods strictly feature-based strategy, by integrating low-level features in the road photos, such as painted lines or lane edges, whereas a few parameters reflect the roads). The processing of lanes is performed while performing the processing of approximation for the layout parameters if the lane forms are both straight or parabolic curve, and obstacle identification is combined to provide the necessary inspection for the automobile. The workings of Google proposed by [46], shows that a precise 3D map on the area is created by Velodyne 64-beam laser. The car then combines laser measurements with world maps, produces different types of data models, which enable it to operate without barriers and respect traffic legislation.

The Composite RCA port for attaching the yellow video cable to the TV enables you to use the TV as a display. The High Definition Multimedia Interface (HDMI) port enables the Raspberry Pi to be linked to high definition televisions and monitors. It is also used to transmit video and audio from the internet to the TV. DSI (Monitor Serial Interface)–Raspberry Pi can be combined with the monitor. CSI (Computer Serial Interface) support–Raspberry Pi can be expanded using a computer. The GPIO (General Purpose Input and Output) pins on the

ultrasonic sensors will be considered concerning the scope of this project.



Ultrasonic sensors Figure 2-4

2.3 Data Security

“Data security is a set of standards and technologies that protect data from intentional or accidental destruction, modification or disclosure. Data security can be applied using a range of techniques and technologies, including administrative controls, physical security, logical controls, organizational standards, and other safeguarding techniques that limit access to unauthorized or malicious users or processes”

All big and small tech companies, like Amazon, Google, and Facebook; nowadays rely on customer data to perform analytically and to capture new business e.g. telecom companies used customer data to find the churn customers, big tech companies rely on data to understand users trends and patterns, so in summary, data is at play in companies both large and small. The main goal of companies protects data, the data which was collected, stored, created, received and later on transmitted by an organization. As different companies use different type of technologies and devices to store, manage and process data and this data is used by the companies to gain more business, therefore, data protection law which is known as GDPR states that it is primary duty of an organization to protect all types of user data whether in transition or at rest and in all forms. So it doesn't matter which device, technology or process is used to for collecting data, moreover, these data breaches can result in litigation cases and huge fines, so the importance of data security from threats is more important today than it has ever been.

Data security in cloud computing involves more than encryption of data, furthermore, requirements for data security depend upon different services of cloud which are as follows: Service models include SaaS(Software as a service), PaaS(Patform as a service), and IaaS (Infrastructure as a service). Normally there are two states of data which are mentioned below.

2.4 Data at Rest

This means the data stored in the cloud and Data in Transit which means data that is moving in and out of the cloud. Confidentiality and Integrity of data are based upon the nature of data protection mechanisms, procedures, and processes. The most significant matter is the exposure of data in above mentioned two states.

A. Data at Rest Data at rest refers to data in the cloud or any data that can be accessed using the Internet. This includes backup data as well as live data. As mentioned earlier, sometimes it is very difficult for organizations to protect data at rest if they are not maintaining a private cloud since they do not have physical control over the data. However, this issue can be resolved by maintaining a private cloud with carefully controlled access.

B. Data in Transit Data in transit normally refers to data that is moving in and out of the cloud. This data can be in the form of a file or database stored on the cloud and can be requested for use at some other location. Whenever data is uploaded to the cloud, the data at the time of being uploaded is called data in transit. Data in transit can be very sensitive data like user names and passwords and can be encrypted at times. However, data in unencrypted form is also data in transit [17]. Data in transit is sometimes more exposed to risks than the data at rest because it has to travel from one location to another. There are several ways in which intermediary software can eavesdrop the data and sometimes can change the data on its way to the destination. To protect data in transit, one of the best strategies is encryption.

In conclusion, Data at rest is data that is not actively moving from device to device or network to network such as data stored on a hard drive, laptop, flash drive, or archived/stored in some other way. Data protection at rest aims to secure inactive data stored on any device or network. While data at rest is sometimes considered to be more vulnerable than data in transit, attackers often find data at rest a more valuable target than data in motion. The risk profile for data in transit is less easy but data at rest depends on the security measures that are in place to secure data in either state. In contrast, data at rest are in a database and just need to hack the database. Data in transit, or data in motion, is data actively moving from one location to another such as across the internet or through a private network. Data protection in transit is the protection of

this data while they travel from network to network or being transferred from a local storage device to another cloud storage device – wherever data is moving, effective data protection measures for the transit data are critical as data often considered less secure while in motion. In order to evaluate the utility of this weakness and to make it practical, we have built a system that incorporates all these moves. It links any PassThru advertised devices (for instance, through their WiFi connectivity or when connected directly via Ethernet), exploits them through shell injection, and allows the adversary to access the network of the service centre, and (1) compromise all PassThru devices over a network, each compromising any car used by them to support (2 and 3). The system PassThru even virally (4) can be distributed to other systems in PassThru (for example, when a computer is borrowed to others), thereby replicating it (5).

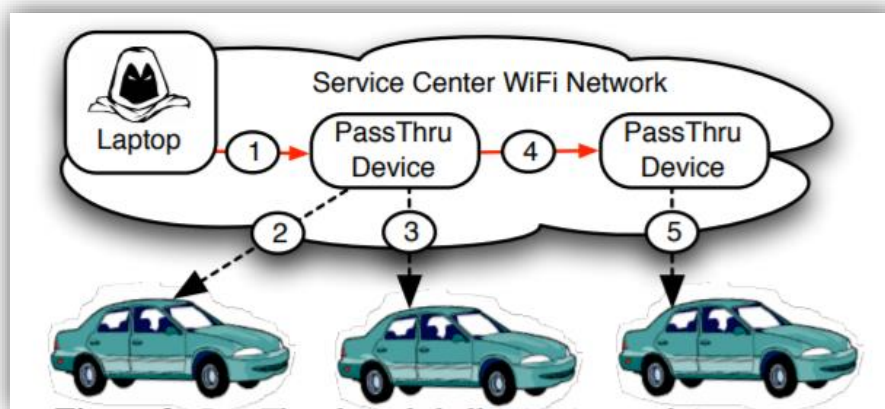


Figure 2-5 The adversary gains access to the service center network

Authors in [52] show that their analyses focus on a representative sedan, which is moderately priced as shown in Figure 2-5. They also developed the model for vehicle risks and applied detailed attacks in key points of this system on several occasions. For example, if the car's radio is compromised and custom firmware uploaded via a phoned CD, they can compromise the Pass-Thru devices of the technician and thus compromise all cars connected afterward to the Pass-Thru device. They can also contact their mobile phone number to get complete arbitrary control of the telematics unit of the car. The only thing that can compromise an ECU in a car, however, is half the story: what an attacker can do with this ability remains a concern. In reality,

the car displayed with an external E / A device to activate arbitrary functions remotely and to share filtration data such as the location of the vehicle and the cabin audio. Ultimately, the circumstances under which an aggressor may use the capability that they build up in this research are clear, financially motivated. LIN and CAN, protocols that don't support default security mechanisms of any kind, are used by key vehicle networks; thus they are vulnerable to multiple attacks [53]. Several attacks have been caused by faulty encoding systems, allowing automatic, unauthorized and arbitrary vehicle access [53]. There is much vulnerability that benefits from entry points that modern vehicles have, such as Bluetooth, or DSRC. Besides, major faults have been established in long-range wireless Internet connectivity, and many attacks have been created that target transmitted and addressable channels [55], [56]. Given the above vulnerabilities, the US and others are revealed. Recently issued a Request for Information, the Transportation Department outlined the threats to home security [57]. Projects such as EVITA2 and [58] are trying to foster new security-aware frameworks for vehicles. The privacy of drivers can be easily protected (i.e. preventing other users from accessing the data stored in EDR). One simple solution is to locally encrypt the data with a driver-friendly key. This approach solves every privacy issue; unfortunately, the problem arises because in the event of a serious accident the driver may die and cannot provide the authorities with the decryption key. After all, EDR has a key role to play in providing information about serious accidents, and a simple encryption regime in these cases makes EDR useless. EDR's protection system aims to achieve the correct balance between driver and passenger safety and the successful use by officials of EDR in the event of an accident. The plan also gives approved parties (i.e. health agencies, vehicle manufacturers, etc.) the use of EDR-specified data only in emergencies (i.e. after an accident) without the driver's involvement. Besides, only the latest recorded values are allowed in our system. Only the driver can access the complete log file. Therefore, only if drivers consent, authorities can access old data [58].

2.5 Data Fragmentation

There are different ways to protect data. Researchers in [59] proposed a cloud stash to split data to multi shares, where threshold shares are required to reconstruct the file. They used multithreading to manage secret sharing into multiple clouds. Therefore, when the client tries to download data and it is corrupted, cloud stash can reconstruct a new one from another cloud as shown in Figure 2-6.

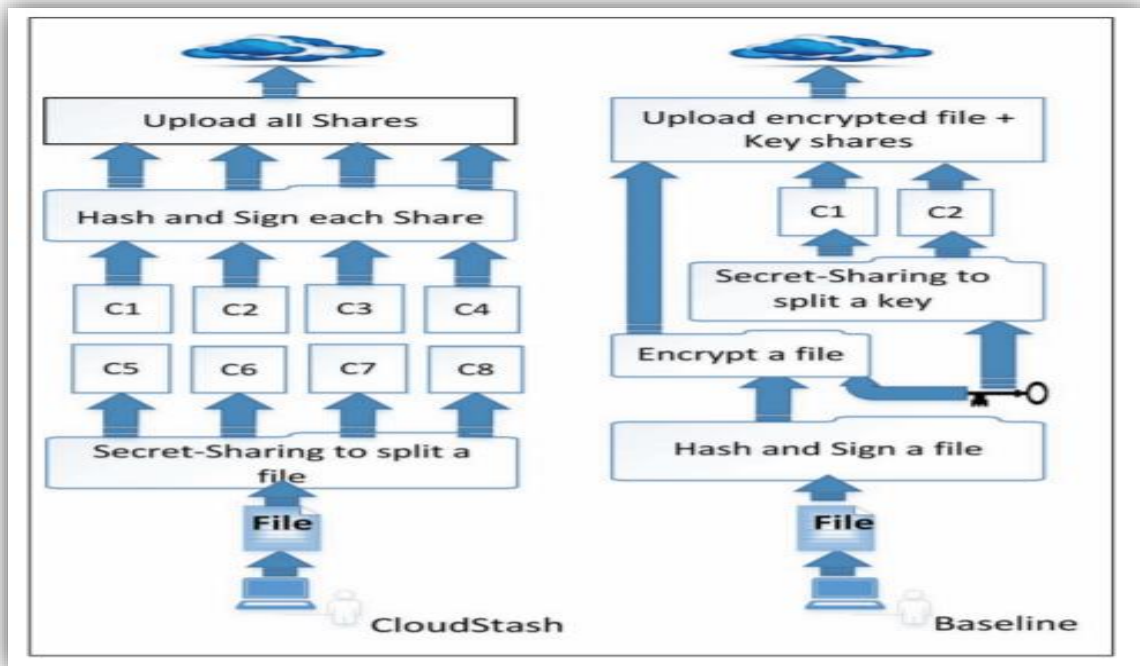


Figure 2-6 Architecture of Cloud Stash and baseline in uploading operation

The authors in [60], [61] proposed a new technique to fragment the original file to chunks, and they used random pattern fragmentation for splitting data, each split file had the same length as the associated pattern. Moreover, the aim of using this method is to protect the data from attackers, as the attackers do not know the information about the length of each split data; therefore, they cannot get the data. Authors in [61] present a new approach for data fragmentation combined with AES 256 CBC to encrypt the data and split to multiple chunks, and the chunks were sorted to split files by using a random pattern fragmentation. This article [62] indicates a self-driving cloud-based vehicle in which the information is removed from the vehicle entirely all this stored in the cloud then. The software speaks to the cloud manager and receives the information needed to go to a specific place. Only the information is uploaded into the vehicle for that place. The cloud remains with all the remaining information. In situations where the vehicle has to move to places where the network connectivity is not contiguous, internal storage is also needed. In the newly proposed systems, the information is not in the database at all. Since the vehicle downloads information that is only needed on the go and used up information is wiped off after use, there is now no fear of outsiders mishandling the information.

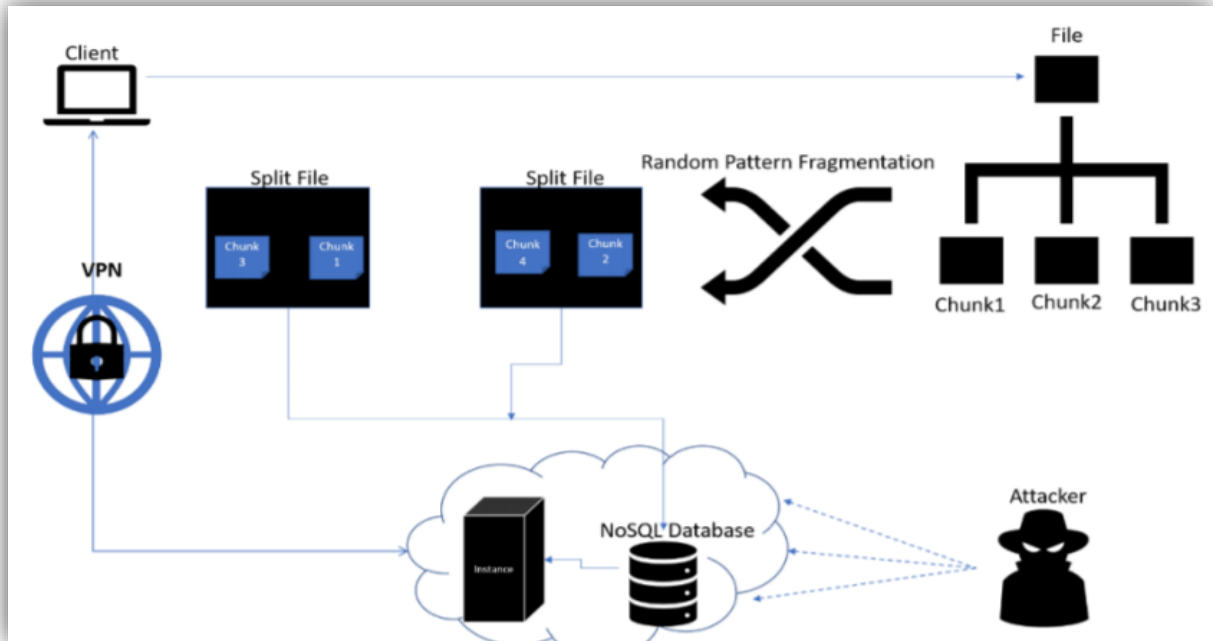


Figure 2-7 Proposed model that uses random pattern fragmentation and stores the random chunks in split files, which are then stored on a NoSQL database [60].

Also, even the data held inside the external storage is encrypted so that it is not easy for the attacker to manipulate the information quickly. All the information resides in the cloud, including the private data of the customers, in the newly suggested concept. The customer can stop the download of his private data when going for a significant conference or any pressing situation so that the vehicle immediately brings him to the place that is wasting time analysing the user's behavioural patterns. This makes it possible to download fewer data into the system, saving more space. This part of the information is with the vehicle now, but untouched as well, which is even better. For security authors in [63] proposed that the data storage security is the focus of attention. As a defence, distributed storage infrastructure will be used as a cloud data security service. To achieve maximum accuracy, the data is divided into several batches and then encrypted by generating a polynomial function for each batch described with the RSA algorithm and stored in separate databases. To guarantee cloud performance, time sensitivity will play an important role. If the collision is observed, the cloud provider should only take time to reply to the consumer promptly. By simultaneously encrypting and downloading encrypted data parallel to cloud storage, the performance of the cloud will also be improved. It increases the speed of sending the post. The reliability of hard discs is ensured to make the cloud available, using reliable storage agreement as the hard drive is the main storage medium

in the cloud environment. Cloud data privacy is safeguarded using the overlooked RAM technology (ORAM). Oblivious RAM (ORAM) is a validated safety mechanism to ensure consumers have accessible, distributable data entry instances. Various ORAM developments have been proposed late to improve the ORAM's communication skills, but capacity efficiency has not been taken into account. Authors in [30] propose a taxonomy of defence against attacks on autonomous vehicles to enable the targeted defence to be developed. According to [64] to make the data more secure they proposed to split the data, then after splitting the data, the header was stored in another file of smaller size, they then added some bytes to the header making it harder to be attacked as it appears to be like other header files. This will make it harder for the attacker to find the header because when attacker attacks the data inside the cloud, they cannot find the header because all the split files have the same size, as it shows that the header was not sent to the cloud, so the attackers cannot defragment the split data without headers.

2.6 Data Encryption

To protect and make data more secure from attackers, work presented in [65] encrypted the data before sending it to the cloud to make it more secure. Only the clients had access to the decryption keys making it difficult for the attackers to get the data. Moreover, to make data safer from the attackers, Jonas in [66] utilised symmetric encryption. For each authorized client key-encryption key (KEK) obtained via passphrase and also by using KEK the master key has been encrypted, after the data encrypted each client the master key and client identifier encrypted depending on user key-encryption key. For decrypting the data, the user enters a passphrase to the client which obtains the KEK, so the client decrypts the master key and data. From the literature, it is concluded that many researchers have worked on autonomous cars data splitting data or encrypting data and splitting headers, but no one has tried to split and encrypt the headers of different files.

2.7 Public Key Encryption

Public key encryption also is known as asymmetric encryption and is a type of cryptographic that consists of two keys, one is the public and the second one is the private key. The public key (which is accessible to everybody) is used to encrypt the data whereas the private key is used to decrypt the data. In [60], Sana et al. proposed an algorithm which by using lightweight symmetric encryption, moreover, the author used asymmetric encryption security mechanism to distribute keys. Although the proposed algorithm had shown promising results but were a few disadvantages to this method. One of them was the management of key, another drawback

was that this solution is not flexible and scalable because encryption and decryption are needed when a user leaves the group to prevent him from accessing the data.

2.8 Identity-Based Encryption (IBE)

In [61], researchers have introduced the concept of identity-based encryption. The owner of the data can encrypt his sensitive data by postulating the identity of the person which is truly worthy in his view, furthermore, he will authorize entity to decrypt it based on that entity identity, which should be same as specified by the owner, and as a result, there was no need for key exchange.

2.9 Attribute-Based Encryption (ABE)

In this type of encryption, a set of attributes are used to identify the user, later on, these set of attributes are used to create the secret key, moreover, it also defines the access structure which can be used for access control, furthermore, these access controls are used for encrypting data to ensure the confidentiality. This type of encryption is created by combining encryption with access control.

2.10 Fuzzy identity-based encryption

In [62], another type of attribute-based encryption was proposed by the authors and named it fuzzy identity-based encryption. In this scheme, a group of attributes identifies someone's identity. The data owner encrypts his data and only the one who has attributes that overlap with the attributes specified in the cipher text can decrypt it. There are general schemes than ABE, which is based on trees.

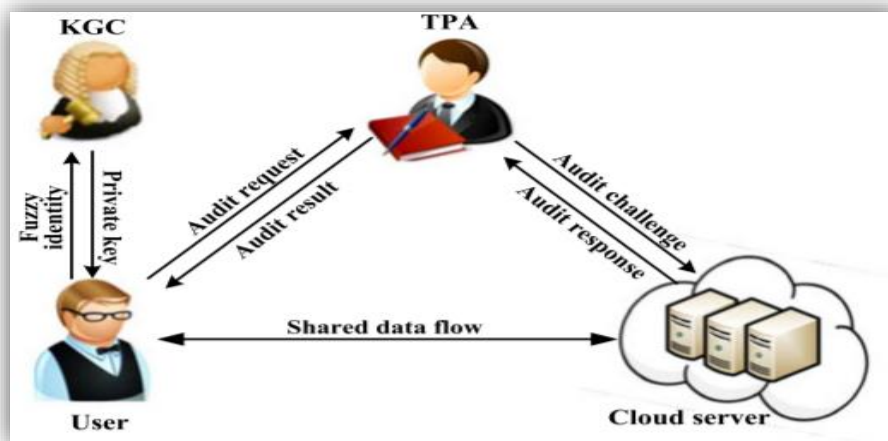


Figure 2-8 The system model of fuzzy identity-based data integrity auditing protocol

3 Chapter Proposed Method

3.1 Proposed method

As privacy is needed to hide the data from the attacker's eyes, it is proposed to split data to smaller chunks and encrypt the headers, then send the chunks to multiple clouds which lead data more secure. There are many techniques to make data more secure, i.e. split data before sending to the cloud or encrypt whole data, another way to secure data is encrypting just headers and send it to the cloud. Data collected from autonomous buggy, the data has been split to different sizes to make a comparison between different sizes to see the time and CPU usage to split data. Then encrypted headers which need less time and CPU usage of both text and video files and whole file to see the difference between all sizes which encrypted and the whole file for both text and video, after splitting data and encrypting, the data re-joined then compared the file sizes after encryption to see the difference before and after encryption. In this work two algorithms have been used to encrypt data RSA and AES, RSA (Rivest–Shamir–Adleman) is an asymmetric cryptographic algorithm; also it is called public-key cryptography. RSA is one of the first public-key cryptosystems which uses a public key for encryption and for decryption the key should be kept secret. The other algorithm that used for encryption is AES, AES is a symmetrical block cipher used by the U.S. government for securing classified data and is deployed globally in software and hardware to encrypt sensitive data. The video files are big files, so we split them to 512KB chunks to be manageable and then each chunk further to 5KB and 507KB in order to encrypt the header and part of the data and conduct are experiments with multiple cloud file storage.

Figure 3.1 shows how the data is collected from sensors of the autonomous buggy. First, the microcontroller is connected to DAC then to the Raspberry Pi, which is also connected to the ultrasonic and camera sensors and microcontroller connected to autonomous vehicle, as privacy is needed to hide the data from attacker's eyes, we proposed to split data to smaller chunks and encrypt the headers, then send the chunks to multiple clouds which leads data more secure as shown in (Fig.3.1).

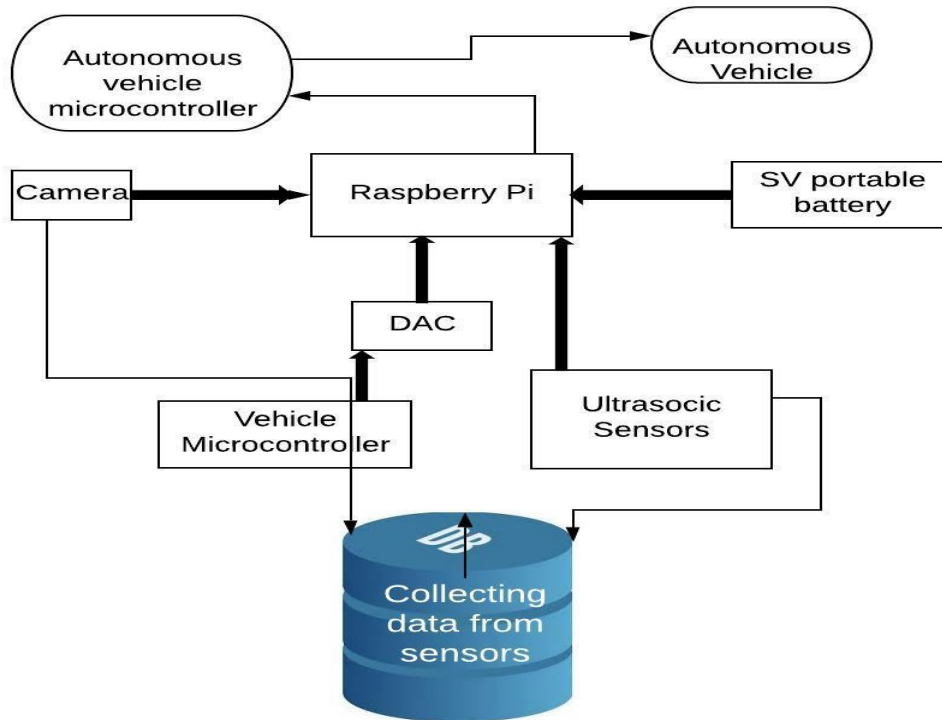


Figure 3-1 Block diagram of the autonomous vehicle and Data Collection

3.2 Experimental Methodology

The vehicle used for this project is a Capricorn Electric Wheelchair from Better life Healthcare [67], as shown in Figure 3-2. It is a small, four-wheeled vehicle with caster type front wheels; two fixed driven rear wheels and powered by two 12V batteries. It is driven by two separate electric motors, which are connected directly to each of the rear wheels. It has a maximum speed of 4mph, a maximum incline of 6°, a turning circle of radius 475mm, the maximum range of 9.5km wheelchair, solid tires, are and a larger radius than many other models of its type, helping to improve performance on rough or uneven surfaces.

3.3 Connecting the Raspberry Pi 3

The autonomous vehicle was built from a mobility scooter as shown in Figure 3-2. The scooter had an inbuilt microcontroller which was used as a communicative tool between the Raspberry Pi version 3 and the vehicle's motors. To make space for a platform on which the system can be installed, the chair was removed, as was the housing surrounding the frame of the vehicle. This thesis will investigate the effectiveness of a Raspberry Pi as a microcontroller to control an autonomous buggy to achieve successful object detection capabilities by recognizing both test obstacles and real pedestrians. Computer vision techniques and their algorithms will be compared for their suitability to detect pedestrians. We used Raspberry Pi because it is cheaper than real controller and also our research project was not on the actual hardware but on the communication and the data collected so we wanted a real simulation that could be as close as possible to car manufacturers who in the end they use microcontrollers as well.



Figure 3-2: Autonomous vehicle

The central column between the chair and the frame was also removed, allowing the new chassis to be placed over the frame. The chassis has enough space for the control panel – rewired to connect the Raspberry Pi directly to the joystick input, the Pi itself, and two breadboards with which the circuitry could be modified during the built and testing process. The chassis is designed so additional components and sensors can be added. Two digital to analog converters were installed, controlling both forwards/backward motion and the yaw of the vehicle, respectively. The front wheels were fixed in place by the removal of the bearings contained in the shafts. This allowed the connecting bolts to be tightened fully and restricting the motion of the vehicle to forwards and backward.

For the vehicle to be autonomous it would have to be controlled by the General-Purpose Input Output (GPIO) pins on the Pi which would send signals emulating the joystick. The GPIO pins work with digital signals, therefore, a Digital to Analogue Converter (DAC) would be required to alter the signal type. An Adafruit MCP4725 DAC [20] was used and functioned well with the Raspberry Pi. To ensure the vehicle was mobile, a portable battery producing 5V was powering the Pi and a laser-cut housing was designed to hold the system in place.

3.4 Connecting Ultrasonic Sensors to the Raspberry Pi 3

Ultrasonic sensors provide basic object-detection autonomy to the vehicle. The HC-SR04 sensor was used which could work with the Pi's GPIO pins through jumper wires. The principle of the HC-SR04 is that there are four pins: power, trigger, echo, and ground. The power and ground were connected directly to the Pi's voltage and ground pins, the trigger acts as a 'starting gun' for the sensor signifying when to produce a soundwave, and the echo receives the sound wave. While these are binary input/output functions they can be used on Python to determine the distance of the closest object. To connect the ultrasonic sensors to the Raspberry Pi, the circuit was adjusted so that all of the triggers were controlled by the same i/o pin on the Pi, keeping the amount of i/o pins used to a minimum of $x+1$, where x is the number of sensors used in the design. Each of the sensors outputs the result to an array that is continually updated, and it is this array that can be communicated to an infrastructure hub. As sensors connected to Raspberry Pi, and the microcontroller makes a connection between vehicle and Raspberry Pi, we got some data from the sensors and we collected the data. In this thesis will investigate the effectiveness of a Raspberry Pi as a microcontroller to control an autonomous buggy to achieve successful object detection capabilities by recognizing both test obstacles and real pedestrians. Computer vision techniques and their algorithms will be compared for their suitability to detect pedestrians. We used Raspberry Pi because it is cheaper than real controller and also our research project was not on the actual hardware but on the communication and the data collected so we wanted a real simulation that could be as close as possible to car manufacturers who in the end they use microcontrollers as well. (Figure 3-1) shows the whole process of connecting sensors to the raspberry and collecting data.

3.5 Software used for implementation

An important aspect of the speed and efficiency of the network is the programming language that an autonomous car uses. The efficacy of a programming language relies upon many factors, from readability and difficulty to usability and functions. Python, a general-purpose open-source object language developed by Guido van Rossum in 1991, focuses on reading

ability. Python developers can produce the same output with fewer code lines than a C++ or java developer because of the tremendous variety of libraries available making developers' life easier.

In machine education, Python is also the most commonly used language for 57% [79][80] and supports Open CV, a common software library for device viewing. Python language is probably the most common and useful for independent systems, as its various library users will easily use it to become one of the most dominant languages [81]. Python is often compared with others, such as the MATLAB and the R. Academic comparisons prefer Python over MATLAB because of its open-source nature and its quicker readability debugging [82].

The Python 3.7.1 scripts will be used in this report as flowcharts for readability, but all code for this project can be found in Appendix 2.

3.6 Data encryption and fragmentation

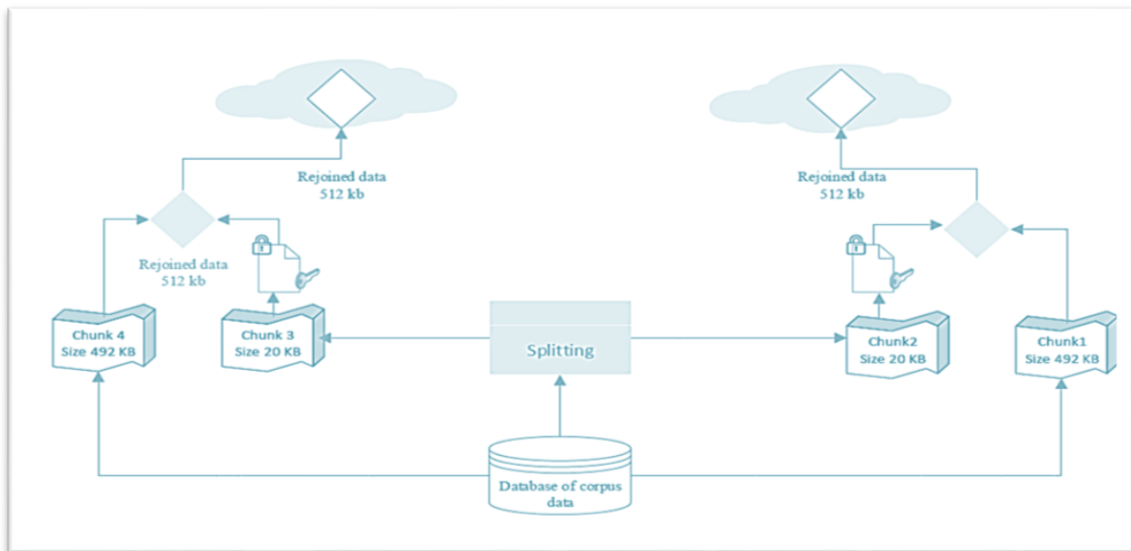


Figure 3-3 data encryption and fragmentation process

Text data from the ultrasonic sensors and video data from the camera were collected from the autonomous vehicle. The AES encryption algorithm was used as it is the most widely used encryption method; it is symmetric encryption which uses one key for encryption and decryption. Data was collected from the autonomous vehicle ultrasound sensors and the camera. The whole text and video file was then encrypted, and then fragmented both the text and video files into four fragments of sizes 5KB, 10KB, 15KB and 20KB. The size of the whole

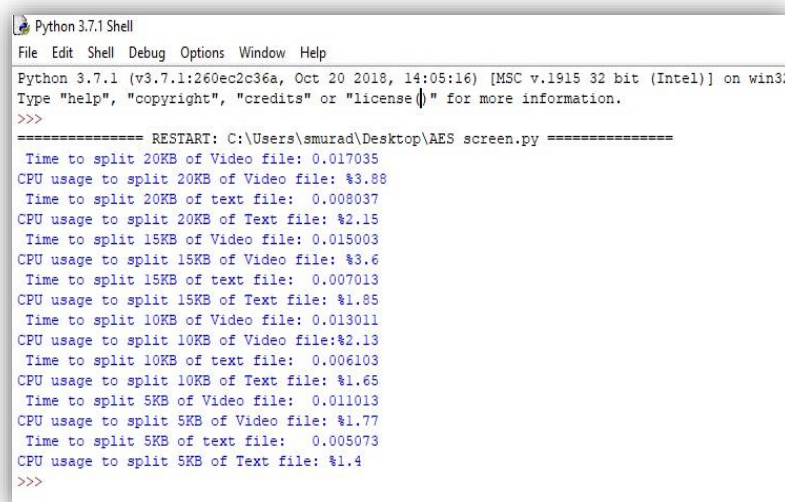
video file was 512KB and the text file was 247KB. This allowed us to make a comparison in terms of time, size and CPU usage for the full-size file and when fragmented.

Currently, there are various types of encryption methods available. AES method and e-stream are two of those methods. E-stream was introduced by EUROCRYPT for the sake of identification of stream ciphers. It was used when the other six ciphers were failed to produce the required results when they were submitted to the NESSIE project. This project has used the AES method for research because it is the most advanced procedure of encryption that exists in the contemporary tech world. The full-size video and data files were encrypted with the AES method. Finally, the fragmented video and text files were encrypted with the AES method as described earlier in Figure 3-3. After fragmenting the data files in 4 different sizes, we encrypted them and then re-joined it with the remaining.

Files were fragmented as 507, 502, 497 and 492KB, and send it to multiple clouds, so when the attacker tries to attack the data in the cloud they will not be able to decrypt it. This is because the hacker cannot understand anything from the data. After all, it will be difficult to find all the chunks and determine the difference between the header and the body. For reconstructing data, we used the same method i.e. the client downloads the data chunks and decrypts the headers, then re-joining the chunks and the client has access to the original data. After the client has downloaded text or video data, as each client has his key so it will be easier for the client to get the first chunk after that they can find all chunks as each chunk has the information about previous and next chunk.

3.6.1 AES 256 CBC Encryption

From the proposed method, two different methods have been used for data encryption but



```
Python 3.7.1 Shell
File Edit Shell Debug Options Window Help
Python 3.7.1 (v3.7.1:260ec2c36a, Oct 20 2018, 14:05:16) [MSC v.1915 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\smurad\Desktop\AES screen.py =====
Time to split 20KB of Video file: 0.017035
CPU usage to split 20KB of Video file: %3.88
Time to split 20KB of text file: 0.008037
CPU usage to split 20KB of Text file: %2.15
Time to split 15KB of Video file: 0.015003
CPU usage to split 15KB of Video file: %3.6
Time to split 15KB of text file: 0.007013
CPU usage to split 15KB of Text file: %1.85
Time to split 10KB of Video file: 0.013011
CPU usage to split 10KB of Video file: %2.13
Time to split 10KB of text file: 0.006103
CPU usage to split 10KB of Text file: %1.65
Time to split 5KB of Video file: 0.011013
CPU usage to split 5KB of Video file: %1.77
Time to split 5KB of text file: 0.005073
CPU usage to split 5KB of Text file: %1.4
>>>
```

Figure 3-4 Time and CPU usage for fragmenting video and text files

before encryption the data fragmented to different sizes 20 KB, 15KB, 10KB, and 5KB, to see the time and CPU usage for splitting headers and make a comparison between them. Figure 3-4 shows the time and CPU usage of splitting data. Figure 3-5 show Python shell of splitting video and text files to different sizes, from the output we can see that there is a difference to fragment 20 KB of video which needs 0.017 seconds for fragmentation and CPU usage is 3.88%, for the text files it needs 0.008 seconds and 2.15% CPU usage, from the output of running program best size which needs less time and CPU usage is 5KB for both text and video files which is 0.01, 1.77%, 0.005, 1.4% respectively.

```

1. Press '1' to encrypt file.
2. Press '2' to decrypt file.
3. Press '3' to Encrypt all files in the directory.
4. Press '4' to decrypt all files in the directory.
5. Press '5' to exit.
1
Enter name of file to encrypt: video.mp4
it tooks:0.08 seconds to encrypt 20KB Video
CPU usage:%30.00 to encrypt 20KB Video
it tooks:0.007 seconds to encrypt 15KB Video
CPU usage:%29.00 to encrypt 15KB Video
it tooks:0.005 seconds to encrypt 10KB Video
CPU usage:%23.40 to encrypt 10KB Video
it tooks:0.002 seconds to encrypt 5KB Video
CPU usage:%21.30.00 to encrypt 5KB Video
it tooks:0.24 seconds to encrypt 512KB Video
CPU usage:%33.10.00 to encrypt 512KB Video

```

Figure 3-5 Time and CPU usage for encrypting Video files (AES CBC256).

After fragmenting data the data encrypted to see the comparison between both methods that have been used in this work according to time CPU usage, time and storage. The Figures 3-6 and 3-7 below shows the output of encrypting fragmented headers and whole file of video and

```

1. Press '1' to encrypt file.
2. Press '2' to decrypt file.
3. Press '3' to Encrypt all files in the directory.
4. Press '4' to decrypt all files in the directory.
5. Press '5' to exit.
1
Enter name of file to encrypt: data.xlsx
it tooks:0.007 seconds to encrypt 20KB Text
CPU usage:%15.5 to encrypt 20KB Text
it tooks:0.005 seconds to encrypt 15KB Text
CPU usage:%17.8 to encrypt 15KB Text
it tooks:0.003 seconds to encrypt 10KB Text
CPU usage:%12.00 to encrypt 10KB Text
it tooks:0.001 seconds to encrypt 5KB Text
CPU usage:%10.00 to encrypt 5KB Text
it tooks:0.1 seconds to encrypt 248KB Text
CPU usage:%24.00 to encrypt 248KB Text

```

Figure 3-6 Time and CPU usage for encrypting Text files (AES CBC256).

After fragmenting the data files into four different sizes, we encrypted them and then re-joined them with the remaining. We first split the data (both text and video) into smaller chunk as the size of video file is to big we first split the video file to chunks of size 512KB. Moreover, split each 512KB to 5KB, and encrypted it then re-joined with the remaining of the files 507KB and sent it to multiple clouds that makes data more secure and the hacker cannot understand anything from the data. If we send some data to the cloud as a clear text, it lets attackers to get data easily. When they get the data, they can read and understand everything from the data but in our work we split video to multiple chunks and we split headers from each chunk and encrypt them as it makes data safer from attackers. text file using AES CBC256. Outputs show that encrypting small chunks of video and text files needs less time and CPU usage while using AES encryption

3.6.2 RSA Encryption

Figures 3-7 and 3-8 show the output of encrypting both video and text chunks after fragmenting headers to 20KB, 15KB, and 10KB, 5KB and whole video and text file. The output shows the comparison of time and CPU usage after encrypting different sizes

```
Confirm password: sahand
RSA Encryption
it tooks:0.1 seconds to encrypt 20KB Video
CPU usage:%32.00 to encrypt 20KB Video
it tooks:0.08 seconds to encrypt 15KB Video
CPU usage:%30.10 to encrypt 15KB Video
it tooks:0.06 seconds to encrypt 10KB Video
CPU usage:%25.30 to encrypt 10KB Video
it tooks:0.04 seconds to encrypt 5KB Video
CPU usage:%21.00 to encrypt 5KB Video
it tooks:0.24 seconds to encrypt 512KB Video
CPU usage:%43.10.00 to encrypt 512KB Video
```

Figure 3-8 Time and CPU usage to encrypt Text files (RSA).

```
RSA Encryption
it tooks:0.07 seconds to encrypt 20KB Text
CPU usage:%17.5 to encrypt 20KB Text
it tooks:0.05 seconds to encrypt 15KB Text
CPU usage:%15.8 to encrypt 15KB Text
it tooks:0.03 seconds to encrypt 10KB Text
CPU usage:%12.70 to encrypt 10KB Text
it tooks:0.02 seconds to encrypt 5KB Text
CPU usage:%9.20 to encrypt 5KB Text
it tooks:0.2 seconds to encrypt 248KB Text
CPU usage:%29.00 to encrypt 248KB Text
```

Figure 3-7 Time and CPU usage to encrypt Video files (RSA).

The 512KB chunks are split to 5KB and 507KB. This 5KB file could be either the header of the video file or payload and in this 5KB file we insert the location of the next file for re-joining. This chunk of 5KB is encrypted using symmetric encryption and is transferred along with the 507KB to clouds. Note that the 5KB and the 507KB from the same chunk could ended in different clouds and locations (example one in Europe, the other in ASIA). In case of sniffing the attacker could have either the encrypted part which would be no use, or in case of the bigger unencrypted file would be a series of raw data which the attacker needs to figure how to view them and reconstruct the chunk which is time consuming as he need to reconstruct the headers, the video frame, video size and many other features which according to forensics is very difficult procedure.

4 Chapter Results and discussion

4.1 Results and Discussion

In this Chapter, we have presented the results of our experiments along with a comparison table of resources consumed by using different encryption techniques. The results of this research are as follow:

- Collect both text and video data and propose a novel way to achieve data integrity.
- To split the data (both text and video) into smaller chunk sizes of 20KB, 15KB, 10KB, and 5KB, and encrypted it.
- To compare that to the whole file encrypted in terms of performance metrics of time, CPU usage and size of files, moreover, we have used Python 3 for programming and encryption e.g. Pycrypto Aes CBC256.

4.1.1 Data collection phase

Data was collected in text form from the ultrasonic sensors and video data from the camera on the autonomous vehicle. We used the AES encryption algorithm as it is the most widely used encryption method; it is symmetric encryption which uses one key for encryption and decryption. We collected the data from the autonomous vehicle ultrasound sensors and the camera. We then encrypted the whole text and video file, then we fragmented both the text and video files into four fragments of sizes 5KB, 10KB, 15KB and 20KB. The size of the whole video file was 512KB and the text file was 247KB.

But while conducting this research there were some challenges. Data collection of thousands of autonomous cars was a big problem that caused some major issues. Secondly, data analysis from such massive available data was another problem this research had faced. Thirdly, the privacy and confidentiality of the individual's communication devices connected to the cloud was another major issue.

4.1.2 Data analysis phase

After data collection, data is analysed to proceed with the research. So, in this study once the data was collected from the autonomous vehicle, we have analysed the results in terms of the comparison in time, size and CPU utilization of the text and video files data were collected from different cloud providers which were further pre-processed to bring it in a standard format by using python. Furthermore, data was fragmented to different chunks and the encryption

algorithm was applied. It is significant to note that finally, the data was analysed by comparing the proposed scheme of data fragmentation to the other existing schemes to keep the efficiency, efficacy, and integrity of the proposed technique.

4.1.3 Data encryption phase

The data files were encrypted as the whole file and as the fragmented chunks of size 5, 10, 15 and 20KB. The size of the whole video file is 512KB and the text file is 247KB. We found that encrypting the whole file needs more time and storage rather than encrypting the first part of the file. There is a change in the size of files before encrypting and after encrypting the first 20, 15, 10 and 5 KB. The size of both files (text and Video) increased after encrypting to 20.2KB. For both (encrypting part of file then re-joining and encrypting whole file), as the size of file before encrypting was 20KB or other sizes and after encrypting increased, encrypting whole of the text file results to bigger size of file rather than encrypting just some KB of file as the file size was 247 (253943bytes) but after encrypting the size increased to 248 (254247 bytes). This research has first encrypted the data file and then split it because this is the most efficient and effective technique. In addition to this, this method requires fewer resources e.g. memory and CPU utilization as compared to other methods. As in our experiments, data security was the main focus so to implement this technique we have done thorough studies of existing techniques as proposed by the researchers e.g., we have gone through a cloud stash technique to split data to multi shares, where threshold shares are required to reconstruct the file. In this technique multithreading was used to manage secret sharing into multiple clouds, as a result, users were able to reconstruct corrupted data using cloud stash, furthermore, another technique which we have analysed is to fragment the original file to chunks, In this technique, we have observed that random pattern fragmentation was used to split data, moreover, each split file had the same length as the associated pattern, as this technique was designed in a way that attackers do not know the information about the length of each split data, as a result, security of data is ensured. Another method which gives us intuition to the proposed method with robust security is based on data fragmentation combined with AES 256 CBC to encrypt the data and split to multiple chunks, and the chunks were sorted to split files by using a random pattern fragmentation, so in this research at initial stage data files were encrypted as the whole file and as the fragmented chunks of size 5, 10, 15 and 20KB.

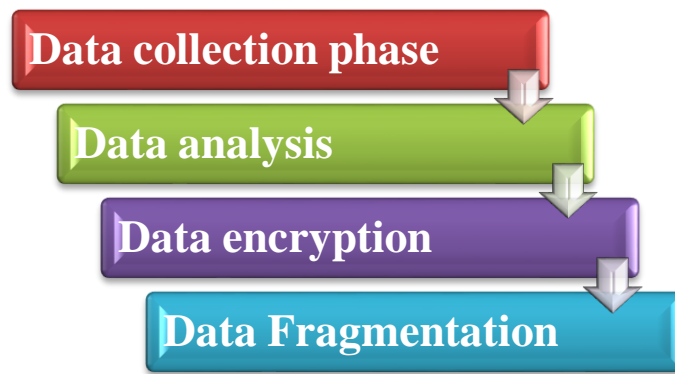


Figure 4-1 Process diagram

The full-size video and data files were encrypted with the AES method. After fragmenting the data files in 4 different sizes, we encrypted them and then re-joined it with the remaining files i.e. 507, 502, 497 and 492KB, and send it to multiple clouds, so when the attacker tries to attack the data in the cloud they will not be able to decrypt it.

Table 4-1 shows that splitting 512 KB to 20 KB and 492KB of both files take more time than splitting the same size to 15 KB and 497KB, 10 KB and 502KB, and 5 KB and 507 KB, also splitting 15 KB of video file takes more time rather than splitting the same size of text file. The chart shows that splitting 10 KB takes more time compared to the time to split 5 KB. Besides, splitting 20 KB takes more time than 15KB, 15KB takes more time for splitting than 10 KB and 5 Kb of a video file is also for the text file splitting 20 KB takes more time than splitting other sizes of the same file. Moreover, from Table 4-1 we can see that splitting 20 KB, 15 KB, 10 KB and 5 KB of video file takes more time rather than splitting the same sizes of a text file. Table 4-1 also shows the CPU utilization for splitting text and video files in different sizes, from the chart we can see that splitting 20 KB need more CPU usage than other sizes of video and text files, also for splitting 15 KB CPU utilization is more than 10 Kb and 5 KB for both files. On the other hand, splitting 20KB of video file needs more CPU usage compared to 12 KB of video rather than splitting 20KB of a text file.

Text	Time	CPU usage
Video		
20 KB	0.008	%2.15
	0.017	%3.88
15 KB	0.007	%1.85
	0.015	%3.6
10 KB	0.006	%1.65
	0.013	%2.13
5 KB	0.005	%1.4
	0.011	%1.77

Table 4-1 Time and CPU utilization for splitting different sizes of video and text file

In recent years, the use of AES data protection encryption has increased. A key to encrypt and decrypt data provides a high level of protection on the data. AES encryption requires more computer resources relative to other approaches as the cycle of encodes and decodes is time-consuming. We can see from Fig. 4-2 the time to encrypt 20 KB of video file needs more time rather than encrypting the same size of a text file, at the same time encrypting 20 KB takes more time than other sizes. Encrypting 15 KB of video and text file needs more time than encrypting 10 KB of both files, but encrypting 20 KB of video file takes more time rather than encrypting the same size of a text file. Encrypting 5 KB of text takes less time than encrypting the same files of size 20KB, 15 KB, and 10 KB and less than the same size of the video file.

Moreover, the chart represents that encrypting 512 KB of video and text file takes more time than encrypting just 20, 15, 10 and 5 KB.

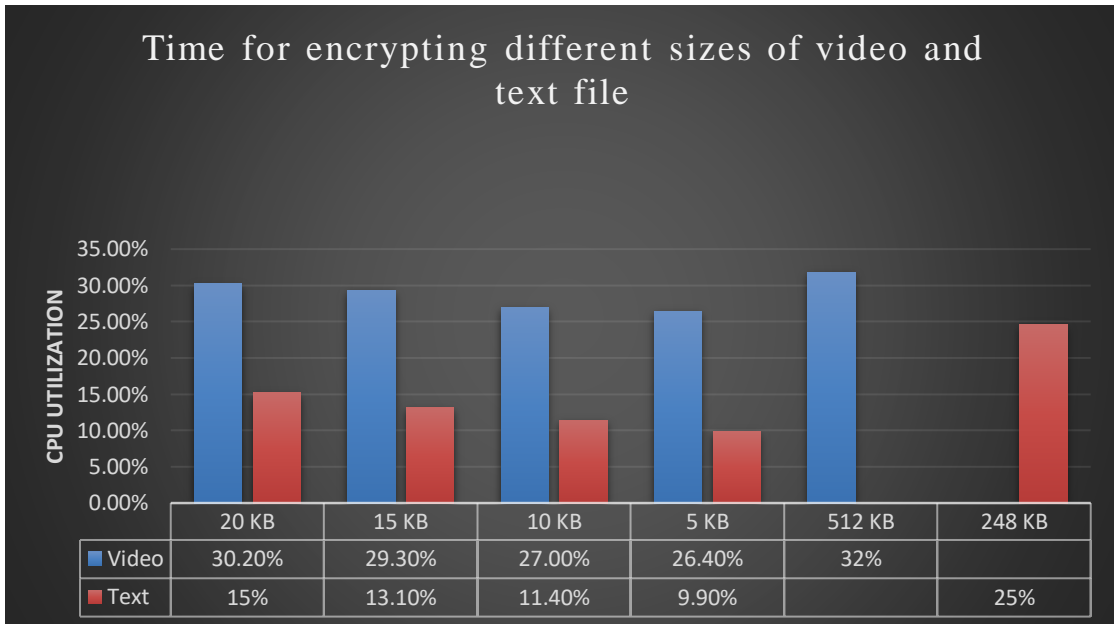


Figure 4-2 Time for encrypting different sizes of video and text file (AES)

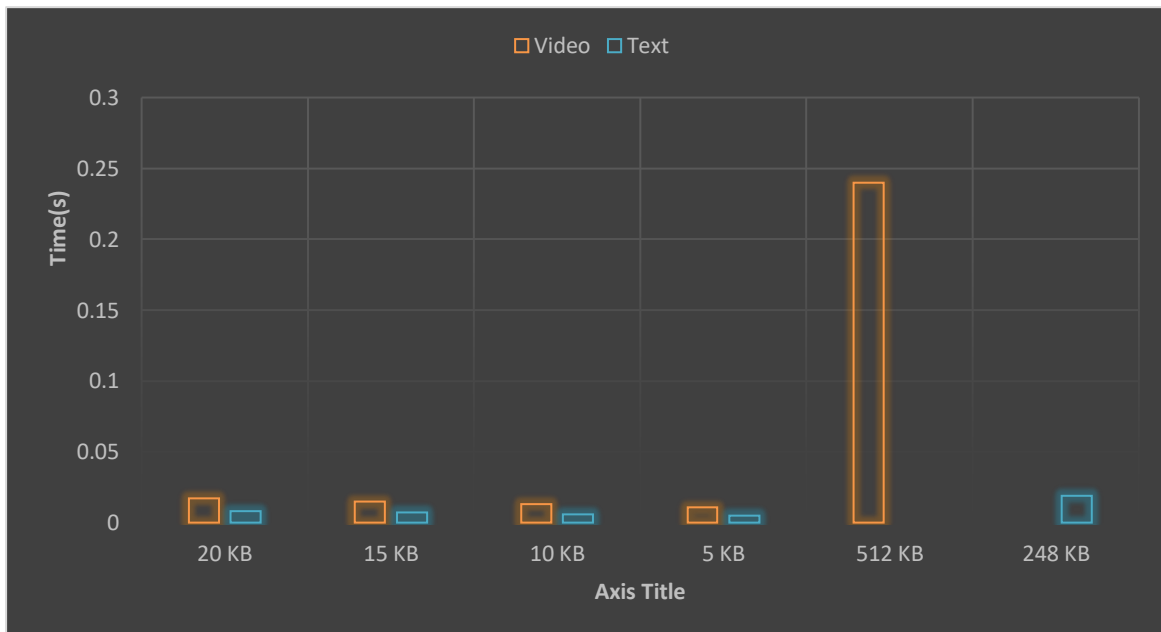


Figure 4-3 CPU utilization for encrypting different sizes of video and text files (AES).

The CPU usage of 20 KB shown in Figure 4-3 shows that usage of CPU for encrypting 20KB of the video is higher than encrypting other sizes. Moreover, when we encrypted 15KB of video file the CPU usage is higher than encrypting 15KB of text file also when we encrypted 10KB for both video and text we can see that the CPU usage of encrypting 10 KB of a video file is bigger than encrypting the same size of a text file. Also, the CPU usage for encrypting 5KB of the text file is less than the video file. Overall from Figure 4-3 we can see that encrypting 5KB results in less CPU usage compared to other sizes. From the experimental results, we can see that the best way to save time and CPU usage is to split and encrypt 5KB of 512 KB rather than other sizes or encrypting the whole file without splitting.

4.2 Comparing Symmetric with asymmetric encryption

RSA is an algorithm for the encryption and decryption of messages by modern computers. It's a mathematical asymmetrical algorithm. Asymmetric implies that two separate keys occur. This is also referred to as public-key encryption because anyone can get one of the keys. We can see from Figure 4-4, furthermore, we have applied the same mechanism to data fragmentation as we have performed for AES to compare the efficiency and efficacy of both algorithms. It can be seen from Figure 4-4 that RSA is utilizing more resources as compared to AES when text and video are fragmented into the same size as in AES.

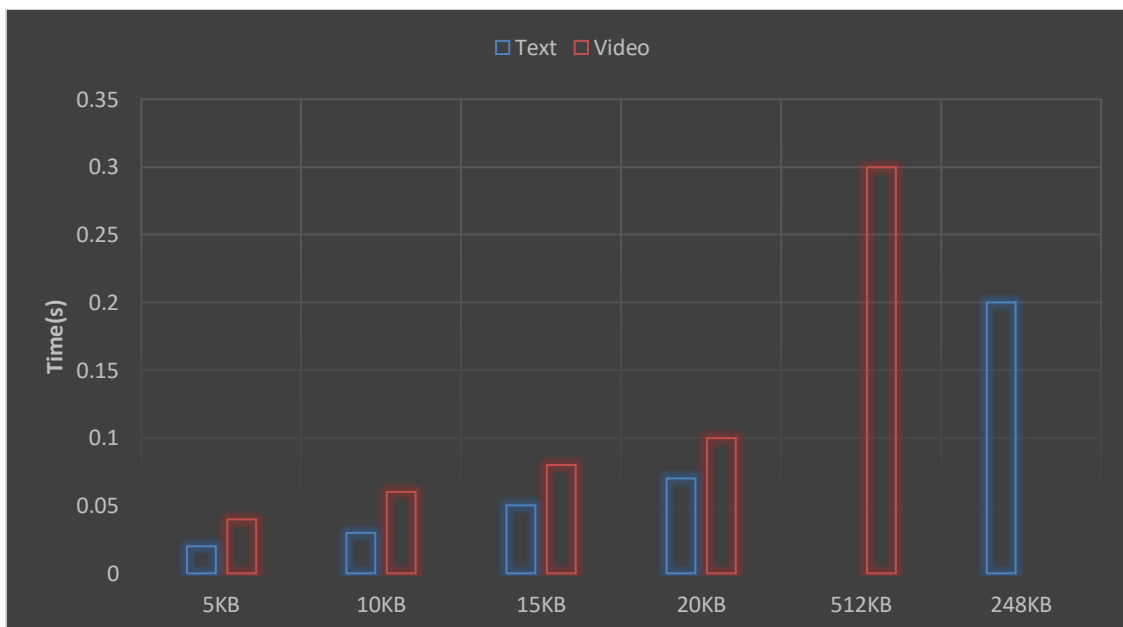


Figure 4-2 Time for encrypting different sizes of video and text files (RSA).

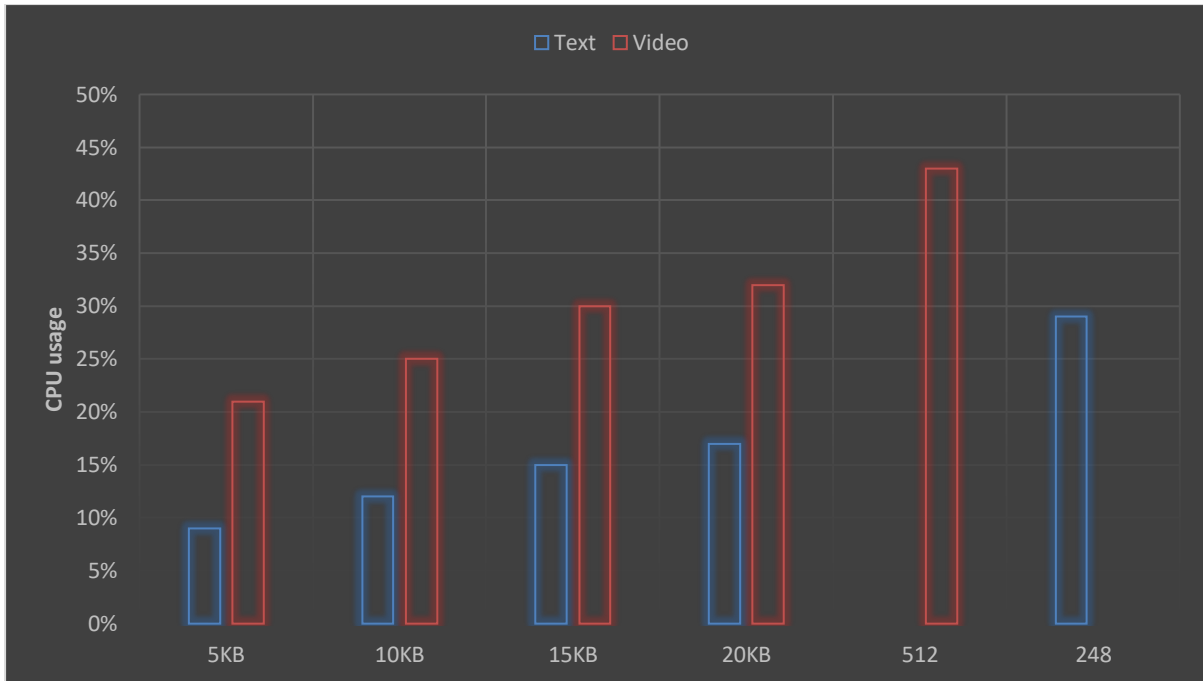


Figure 4-3 CPU utilization for encrypting different sizes of video and text files (RSA).

CPU usage to encrypt video file is higher than encrypting the same size of text files, Figure 4-5 shows that encrypting 20 KB of text file needs less CPU usage rather than encrypting the same size of video files. The CPU usage of 20 KB encryption is higher than encrypting other sizes. To encrypt 15KB of video file needs more CPU usage rather than encrypting 15KB of text file also encrypting 10KB for both video and text, we can see that the usage of CPU to encrypt 10 KB of the text file is smaller than encrypting the same size of the video file. Also, the CPU usage for encrypting 5KB of the text file is less than the video file. Overall from Figure 4-5 we can see that encrypting 5KB results in less CPU usage compared to other sizes.

In this research we have tried to explore different type of cryptographic algorithms (encryption/decryption) to find the best suitable solution which suits our research problem, furthermore, initially, we have started with the first type of cryptographic algorithm that is symmetric algorithm also known as secret key algorithms and the reason is that because they use single key for both encryption and decryption, furthermore, they are considered to be fast as compared to asymmetric key algorithms and best suitable in situations where we have bulk of data was encryption. There are lots of symmetric key algorithms available e.g. DES,3DES, AES, RC4, etc., furthermore, all these algorithms have their own strength and weakness, for example, DES was developed in 1976 and was widely used encryption algorithm in the past until a weakness was found that it uses 56-bit encryption key, which implies that hacker can use any fast process machine available nowadays to run through all bit of combination to break

this algorithm, therefore nowadays it is no longer widely used. RC4 which is considered as the fourth version of the river cipher and is widely being used in the wireless network e.g. WEP and WPA, furthermore, it is also been in Hypertext transfer protocol over SSL also known as HTTPS, moreover, other applications are Secure Sockets Layer (SSL) and Transport Layer Security (TLS.) In this algorithm, an encryption key of variable length can vary from 40 to 256 bits, however, the most commonly used is with a 128-bit key. The most powerful feature of this algorithm was that is very simple and can be easily be implemented, however, if it is implemented improperly than it can easily be hacked and because of the aforementioned reason it is being phased has also been used with secure shell, Kerberos, and the Remote Desktop Protocol and result could be devastating. In this, we have discarded other symmetric algorithms and focused on AES for our experiments because of drawbacks and weakness in other algorithms and another reason for using AES it is nowadays standard algorithms and has been used in several devices ranging from mobile phones to other electronic gadgets, furthermore, till date it is considered as secure and light-weighted, therefore, we have utilized these features of AES in our research to get promising results, moreover, we have extended the scope of two asymmetric cryptographic to see which cryptosystem best suits with our research problem. As symmetric key algorithms are quite efficient, but there are lots of problems in existing system and one of them is key distribution in end devices, as for this purpose we need a secure connection between the key distribution between devices, so to overcome this problem and other aforementioned problem we have explored asymmetric cryptography which is also known as public-key cryptographic system. In contrast to the symmetric cryptography, the asymmetric is developed in a way that it overcomes the problem in the symmetric key , as it was derived from the mathematically hard problem and it is based on “trapdoor” functions, that can be easily compute but complex to reverse. It consists of two keys one is called public and another one is private, the public key is used for encryption and the private key is used for decryption, so in summary, a pair of keys is generated, one is used to encrypt the message, other is used to decrypt cipher message text. One of the most widely used asymmetric key algorithms is RSA. RSA is considered as a secure algorithm as the hard problem in it is to find the prime factors of a composite number and because of this feature of RSA we have used it in our research to see CPU and time consumption for our research problem, furthermore, we have found that AES is performing well for problem as it can be seen by comparing the figures 4-2, 4-3, 4-4 and 4-5.

The data files were encrypted as the whole file and as the fragmented chunks of size 5, 10, 15 and 20KB. The charts below shows the time and CPU usage after encrypting and re-joining data then send it to multiple clouds, the charts show that the reasonable size for video and text file is 5KB and re-joining with the remaining of file which is 507 KB as it needs less time and CPU usage than 20KB, 15KB, 10KB and 512KB.

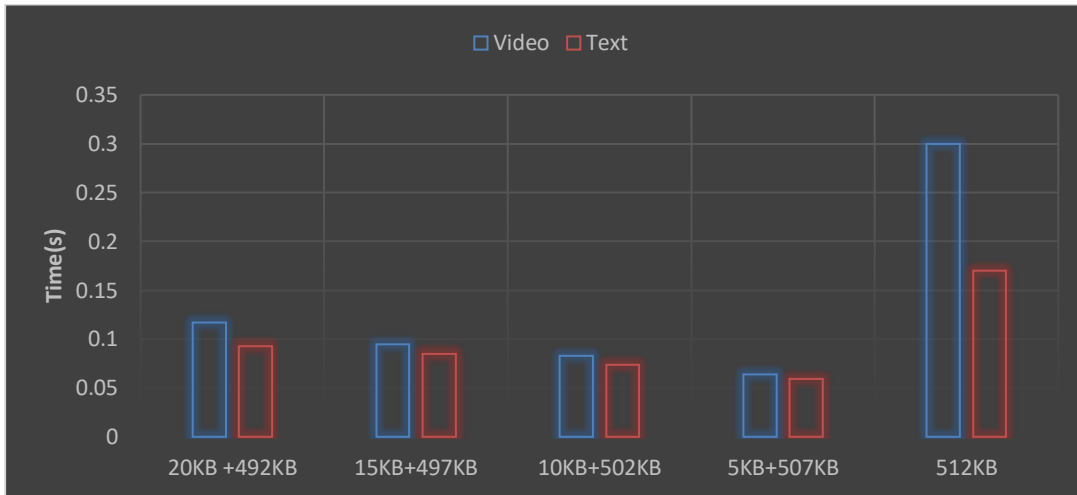


Figure 4-6 Total Time for encrypting and transmitting different sizes of video and text files to cloud

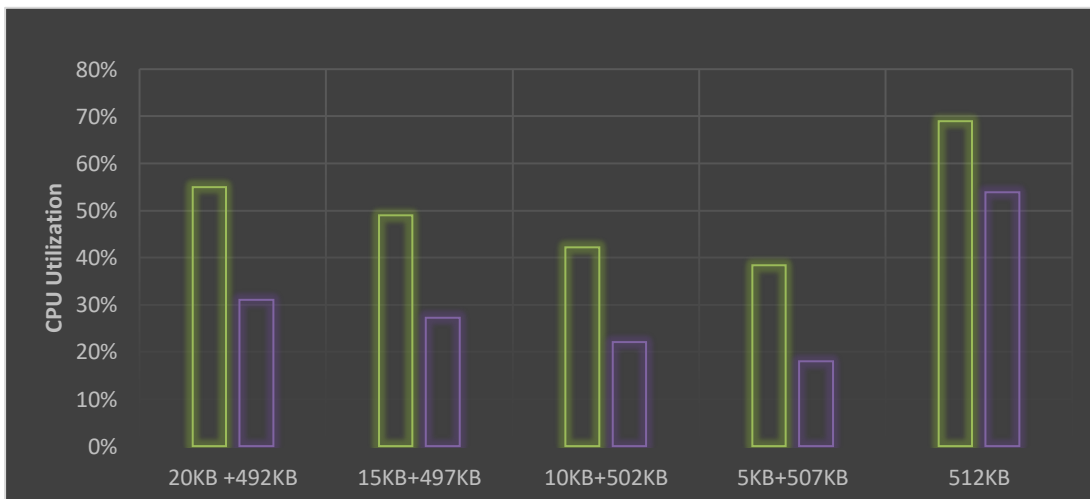


Figure 4-7 Total CPU usage for encrypting and transmitting different sizes of video and text files to cloud

5 Chapter 5

5.1 Conclusion and Future work

This study investigates the privacy and security of cloud-connected autonomous vehicles using cryptography. The literature survey was carried out successfully and targets were also established. Cryptography gives a data security approach. Experimental results have shown that splitting and encrypting data make data more secure, first data fragmented to different sizes then encrypted, 5 KB is the ideal size as it is not using many CPU resources and memory compared to other sizes. We compared our results in terms of time, CPU utilization and size for both AES and RSA algorithms. The problem with RSA is the time and CPU usage which needs more than AES encryption. Also, RSA is asymmetric encryption that uses two keys but AES uses one private key for encryption and decryption which makes data more secure. Also, using RSA encrypting increases file size rather than encrypting the same file with AES CBC256 which leads to use more memory storage. This allowed us to determine which one is the most useful method to make the data secure in the cloud and hence making it difficult for a hacker to reconstruct data. The privacy of data is at a higher level preventing a hacker to access the data as it is shared in multiple clouds and the header of each chunk is encrypted even if attackers get some chunks of data they cannot understand the data as it has been encrypted. Their main problem of this work is how to reconstruct data, after sending data to multiple clouds that the clients cannot reconstruct the data easily so that it makes data lose or helps attackers to get data for this issue. In the future, work on this problem will be extended to solve it and make easier for each client to get data.

In our future work, we will use an e-stream project which is utilized for the identification of efficient and compact new stream ciphers for widespread adoption. Besides that, we will also send data to multiple clouds and add location to chunks also adding steganography for each chunk to hide the information of each chunk so it will be difficult for hackers to find and get information about chunks, then we will reconstruct data from multiple clouds with each chunk having its location also each chunk knows about next and previous chunk location which will help the client to reconstruct the data easily and decrypt data then send it back to the autonomous vehicle.

References

- [1] U.S. Department of Transportation, "2009/2010 Traffic Safety Facts Annual Report," U.S. Government, NHTSA Report, 2009.
- [2] LLC Susanne Trimbath and STP Advisory Services, "Economic Infrastructure: building for prosperity," 2011. [Online]. Available: https://www.uschamber.com/sites/default/files/legacy/issues/infrastructure/files/LRA_Index_Economic_Analysis_2011_10_17.pdf. [Accessed 20 July 2015].
- [3] C. Zhang, R. Lu, X. Lin, P.-H. Ho and X. Shen, "An efficient identity-based batch verification scheme for vehicular sensor networks," in Proc. 27th IEEE INFOCOM, Phoenix, AZ, USA, 2008.
- [4] A. M. Vegni, M. Biagi and R. Cusani, "Vehicular Technologies - Deployment and Applications in Smart Vehicles," in Technologies and Main Applications in Vehicular Ad-hoc Networks, 2013.
- [5] Zhang, R. and M. Pavone, Control of robotic mobility-on-demand systems: a queueing-theoretical perspective. The International Journal of Robotics Research, 2016. 35(1-3): p. 186-203.
- [6] Choi, W.-S., et al. Fast iterative closest point framework for 3D LIDAR data in the intelligent vehicle. in 2012 IEEE Intelligent Vehicles Symposium. 2012. IEEE.
- [7] Levinson, J. and S. Thrun. Robust vehicle localization in urban environments using probabilistic maps. in the 2010 IEEE International Conference on Robotics and Automation. 2010. IEEE.
- [8] H. Cho, Y. Seo, B. V. K. V. Kumar, and R. R. Rajkumar. A Multi-Sensor Fusion System for Moving Object Detection and Tracking in Urban Driving Environments, IEEE International Conference on Robotics & Automation, Hongkong, 2014: 1836–1843.
- [9] Mohomed, Iqbal. Self-driving Lego Mindstorms Robot, Proc. Python in Science Conf. (SCIPY), 2012.
- [10] Ujjainiya, Lohit, M. K. Chakravarthi. Raspberry-Pi Based Cost-Effective Vehicle Collision Avoidance System Using Image Processing, ARPN J. Eng. Appl. Sci, 2015; 10(7)

- [11] P. Zhao, J. Chen, Y. Song, X. Tao, T. Xu, and T. Mei. Design of a Control System for an Autonomous Vehicle Based on Adaptive-PID, *Int. J. Adv. Robotic Syst.*, INTECH, Rijeka, 2012;9(44)
- [12] J. M. Alvarez and A. M. Lopez. Road detection based on illuminant invariance, *IEEE Trans. Intelligent Transportation Systems*, 2010.
- [13] UN. World Urbanization Prospects: The 2011 Revision Population Database. Technical report, United Nations, 2011.
- [14] W. J. Mitchell, C. E. Borroni-Bird, and L. D. Burns. *Reinventing the Automobile: Personal Urban Mobility for the 21st Century*. The MIT Press, Cambridge, MA, 2010.
- [15] CAR2GO. CAR2GO Austin. Car Sharing 2.0: Great Idea for a Great City. Technical report, 2011.
- [16] C. Fricker and N. Gast. Incentives and regulations in bike-sharing systems with stations of finite capacity. 2012. Available at <http://arxiv.org/abs/1201.1178>.
- [17] M. Dell’Amico, E. Hadjicostantinou, M. Iori, and S. Novellani. The bike-sharing rebalancing problem: Mathematical formulations and benchmark instances. *Omega*, 45(0):7 – 19, 2014.
- [18] L. Di Gaspero, A. Rendl, and T. Uri. Constraint-based approaches for balancing bike-sharing systems. In *Principles and Practice of Constraint Programming*, volume 8124 of *Lecture Notes in Computer Science*, pages 758–773. Springer Berlin Heidelberg, 2013.
- [19] S. L. Smith, M. Pavone, E. Schwager, E. Frazzoli, and D. Rus. Rebalancing the rebalances: Optimally routing vehicles and drivers in mobility-on-demand systems. In *American Control Conference (ACC)*, 2013, pages 2362– 2367. IEEE, 2013.
- [20] M. Barth, J. Han, and M. Todd. Performance evaluation of a multi-station shared vehicle system. In *Proceedings 2001 IEEE Intelligent Transportation Systems* pages 1218–1223. IEEE, 2001.
- [21] M. Pavone, S. L. Smith, E. Frazzoli, and D. Rus. Robotic load balancing for mobility-on-demand systems. *The International Journal of Robotics Research*, 31(7):839– 854, 2012.
- [22] L. Burns, W. Jordan, and B. Scarborough. *Transforming personal mobility*. The Earth Institute, 2013.

- [23] Induct. Navia - The 100% Electric Automated Transport, 2013.
- [24] GM. EN-Vs Impress Media at Consumer Electronics Show, 2011.
- [25] A. Fisher. Inside Google's Quest To Popularize SelfDriving Cars. Popular Science (Online Article), 2013.
- [26] T. Kuhn dan J. Fritsch. Visio-spatial road boundary detection for unmarked urban and rural roads, in IEEE Intelligent Vehicles Symposium Proceedings, 2014: 1251–1256.
- [27] Stein, Gideon P., Ofer Mano, and A. Shashua, Vision-based ACC with a Single Camera: Bounds on Range and Range Rate Accuracy, IEEE IV2003 Intelligent Vehicles Symposium, USA, 2003.
- [28] S. F. X. Bayerl, T. Luettel, H. Wuensche. Following Dirt Roads at Night-Time: Sensors and Features for Lane Recognition and Tracking, Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems, 2015; 117–122
- [29] Buczkowski, M. and Stasinski, R., Automatic Lane Detection, PWT 2012, Poznan, 2012
- [30] Thing, V.L. and J. Wu. Autonomous vehicle security: A taxonomy of attacks and defenses. in 2016 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData). 2016. IEEE.
- [31] Chaum, D.L., Untraceable electronic mail, return addresses, and digital pseudonyms. Communications of the ACM, 1981. 24(2): p. 84-90.
- [32] Alsolami, F. and T.E. Boulton. CloudStash: using a secret-sharing scheme to secure data, not keys, in multi-clouds. in 2014 11th International Conference on Information Technology: New Generations. 2014. IEEE
- [33] Shamir, A., How to share a secret. Communications of the ACM, 1979. 22(11): p. 612-613.
- [34] Esslinger, B., CryptTool. Available via [www. crypto or. de](http://www.cryptool.de), 2008.
- [35] The Associated Press, "World's 1st self-driving taxi debut

in singapore," [HTTP://www.bloomberg.com/news/articles/2016-08-25/world-s-first-self-driving-taxis-debut-in-singapore](http://www.bloomberg.com/news/articles/2016-08-25/world-s-first-self-driving-taxis-debut-in-singapore), August 2016.

[36] Ge, Shuzhi Sam και Frank, Lewis L. Autonomous Mobile Robots: Sensing, Control, Decision Making, and Applications. s.l. : CRC Press, 2006.

[37] Dokic, Jadranka, Müller, Beate και Gereon, Meyer. European Roadmap Smart Systems for Automated Driving. smart systems integration.

[38] M. Bojarski et al. End-to-End Learning for Self-Driving Cars. arXiv:1604, 2016.

[39] S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-end training of deep visuomotor policies. Journal of Machine Learning Research, 2016.

[40] N. Otterness, M. Yang, S. Rust, E. Park, J. H. Anderson, F. D. Smith, A. Berg, and S. Wang. An Evaluation of the NVIDIA TX1 for Supporting Real-Time Computer-Vision Workloads. In Real-Time and Embedded Technology and Applications Symposium (RTAS), 2017.

[41] Day, C., McEachen, L., Khan, A., Sharma, S. and Masala, G., L. Pedestrian Recognition and Obstacle Avoidance for Autonomous Vehicles using Raspberry Pi. accepted for presentation at the Intelligent Systems Conference (IntelliSys) 2019, 5-6 September 2019 in London, United Kingdom.

[42] raspberrypi.org. (2017, November 19). Raspberry Pi 3 Model B Specifications. Retrieved from raspberry pi: <https://www.raspberrypi.org/products/raspberry-pi-3-model-b/>

[43] Maksimović, M., et al. (2014). "Raspberry Pi as the Internet of things hardware: performances and constraints." design issues 3(8).

[44] Chaudhari, H. (2015). "Raspberry pi technology: A review." International Journal of Innovative and Emerging Research in Engineering 2(3): 83-87.

[45]Gurjashan Singh Pannu, Mohammad Dawud Ansari, Pritha Gupta, "Design and Implementation of Autonomous Car using Raspberry Pi", International Journal of Computer Applications (0975 – 8887) Volume 113 – No. 9, March 2015.

[46] Erico Guizzo, "How Google's Self-Driving Car Works", October 18, 2011.

- [47] Rzevski, G. (2014). Mechatronics: Designing Intelligent Machines Volume 1: Perception, Cognition and Execution, Newnes.
- [48] Ambrose, C. D. (2003). Frequency Range of Human Hearing. Retrieved from The Physics Factbook: <https://hypertextbook.com/facts/2003/ChrisDAmbrose.shtml>
- [49] (sensorwiki.org, 2018) - <http://www.sensorwiki.org/doku.php/sensors/ultrasonic>
- [50] Tesla. (2017, November 18). Full Self-Driving Hardware on All Cars. Retrieved from Tesla: https://www.tesla.com/en_GB/autopilot
- [51] Varghese, J. Z. and R. G. Boone (2015). Overview of autonomous vehicle sensors and systems. Proceedings of the 2015 International Conference on Operations Excellence and Service Engineering, IOEM Society.
- [52] Checkoway, S., et al. (2011). Comprehensive experimental analyses of automotive attack surfaces. USENIX Security Symposium, San Francisco.
- [53] T. Hoppe, S. Kiltz, and J. Dittman, “Security Threats to Automotive Can Networks — Practical Examples and Selected Short-Term Countermeasures,” Proc. Computer Safety, Reliability, and Security (SAFE-COMP), 2008, pp. 235–48.
- [54] E. Biham et al., “How to Steal Cars — A Practical Attack on Keeloq,” CRYPTO 2007, 2010.
- [55] S. Checkoway et al., “Comprehensive Experimental Analyses of Automotive Attack Surfaces,” USENIX Security, 2011.
- [56] K. Koscher et al., “Experimental Security Analysis of A Modern Automobile,” IEEE Symp. Security and Privacy, Oakland, CA, 2010, pp. 447–62.
- [57] FBO, Cyber Security and Safety of Motor Vehicles Equipped with Electronic Control Systems, Solicitation Number: Dtrt57-11-ss-00007, tech. rep., Federal Business Opportunities, 2011.
- [58] C. Patsakis and K. Dellios, “Securing In-Vehicle Communication and Redefining the Role of Automotive Immobilizer,” SECRIPT, 2012, pp. 221–26.
- [59] Alsolami, F. and T.E. Boulton. CloudStash: using a secret-sharing scheme to secure data, not keys, in multi-clouds. in 2014 11th International Conference on Information Technology: New Generations. 2014. IEEE.

- [60] Santos, N. and G.L. Masala. Big Data Security on Cloud Servers Using Data Fragmentation Technique and NoSQL Database. in International Conference on Intelligent Interactive Multimedia Systems and Services. 2018. Springer.
- [61] N. Santos, S. Lentini, E. Grosso, B. Ghita, and G. Masala, "Performance Analysis of Data Fragmentation Techniques on a Cloud Server" in the press on the International Journal of Grid and Utility Computing, InterScience publishers.
- [62] Yeshodara, N. S., et al. (2014). Cloud-based self-driving cars. 2014 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM), IEEE.
- [63] Dave, R., et al. (2019). "Efficient Data Privacy and Security in Autonomous Cars." Journal of Computer Sciences and Applications 7(1): 31-36.
- [64] Bahrami, M., and M. Singhal. A light-weight permutation-based method for data privacy in mobile cloud computing. in 2015 3rd IEEE International Conference on
- [65] di Vimercati, S.D.C., et al., Encryption and fragmentation for data confidentiality in the cloud, in Foundations of security analysis and design VII. 2013, Springer. p. 212-243.
- [66] Jonas, P.E., A.L. Roginsky, and N. Zunic Encrypting data for access by multiple users. 2009, Google Patents.
- [67] Betterlife Healthcare, 2018. Betterlife Capricorn Electric Wheelchair.

