# NAEBA: An Encrypted Black-Box Architecture for Tamper-Proof Automotive Evidence

Vijay Bhagwat Ukande[1], Prof. Dr. Swapnesh Taterh[2]
[1]*MSc Cybersecurity, MIT Arts, Commerce & Science College, Alandi, Pune, Affiliated to Savitribai Phule Pune University*
[2]*Guide, MSc Cybersecurity, MIT Arts, Commerce & Science College, Alandi, Pune, Affiliated to Savitribai Phule Pune University*
doi.org/10.64643/ijirt.208743-459

***Abstract*—In India, hit and run cases are very hard to solve. Usually there's just no proof of what happened. Dashcams are getting common now, but they don't hold up in court. Anyone can pull the SD card and delete or change the clip, and that's it, evidence gone. In this paper I am proposing a system called NAEBA, National Automotive Encrypted Black-Box Architecture. The idea is simple. The car records video and locks it with encryption the second it's captured. Nobody can open that footage unless two separate keys come together. One key stay inside the car. The other sits with a judicial authority, not the police officer on the spot, not the car owner. So even I, as the owner of my own car, wouldn't be able to peek at my own footage whenever I feel like it. Police only get access if an FIR is actually filed, and even then, only for the exact hour or two they need, not the whole month of recordings. I used AES-256-GCM for the encryption and based the key-splitting on Shamir's 1979 idea. Storage wipes itself every 30 days automatically, which keeps it in line with India's DPDP Act. While I was running the storage numbers for this paper, I actually found an error in my own earlier calculation, and I've kept that in the Results section instead of quietly fixing it and pretending it was never wrong. The last part of the paper is about what building this for real would actually take.**

***Index Terms*—Automotive Cybersecurity, AES-256 Encryption, Threshold Cryptography, Digital Public Infrastructure, Hardware Security Module, Evidence Integrity**

## I. INTRODUCTION

### 1.1. Background

Hit and run accidents are a bigger problem than people realize. A car hit someone. It drives off. Unless a camera happened to be pointed at that exact spot, or someone remembers a plate number under pressure, the case goes cold. Fast. This happens constantly in India because CCTV coverage is patchy at best, decent near big junctions, basically nonexistent on the smaller roads where a lot of these accidents actually occur.

Dashcams got popular over the last few years mainly because they became cheap. But nobody designed them thinking about court evidence. The footage just sits on a regular SD card. Anyone in the car, owner, driver, doesn't matter, can pull it out, format it, or just record over it. There's no way to prove the clip wasn't touched. So even when a dashcam does catch something real, a defense lawyer only needs one sentence to throw doubt on it: how do we know this wasn't edited?

### 1.2. Problem Statement

So basically, the problem is, India doesn't have any standard way to record vehicle video evidence that is both hard to tamper with and also respects the privacy of the person driving. Either the system records everything and becomes a privacy issue, or it's unencrypted and anyone can just edit it, so it becomes useless as evidence. What is actually needed is something in between, a system that stays locked normally and only opens through a proper legal process for the exact time that is needed.

### 1.3. Objectives

- To design a system built into the vehicle hardware that records and encrypts video on its own, without depending on the car owner to keep it safe.
- To design a key system where the video can only be decrypted if a key stored in the hardware and a key held by a judicial authority are combined together, so no one person can access it alone.

- To see how this kind of system could connect with things India already has, like VAHAN and CCTNS, so evidence extraction becomes a proper digital process instead of depending on individual police officers.

1.4. Significance

People usually assume privacy and security are opposites. Give up one to get the other. NAEBA is my attempt to show that's not actually true if the encryption is done right. Nobody can browse someone's driving footage out of curiosity, that's just mathematically off the table. But if there's a real accident, a real crime, the legal path to that evidence still exists. Both things, at once. That's the whole point.

1.5. Existing Frameworks

Aviation already solved a similar problem a long time ago using Flight Data Recorders, but that works because there are very few aircraft and one central authority like DGCA handles investigations, which is very different from having millions of private cars on the road. In cars, the US has something called an Event Data Recorder under a standard called SAE J1698 since around the mid-2000s, but it only records a few seconds of data like speed and braking before a crash, not video, and there's no judicial key system involved in it at all [1]. The EU also has something called eCall but that is mainly for automatically calling emergency services after a crash, not for storing evidence. None of these systems actually combine continuous encrypted video with a two-party legal decryption process, and that is the exact gap NAEBA is trying to fill.

## II. LITERATURE REVIEW

Research on automotive black boxes has changed a lot over the last twenty years or so. In the early 2000s, right after SAE J1698 came out, most of the research was only about using the recorded data to reconstruct accidents, basically treating it like a diagnostic tool and nothing about privacy or security. It was only around 2010 to 2013 that people actually started worrying about how this data was being protected, because before that it mostly wasn't protected at all.

One paper that really stood out to me was by Patsakis and Solanas [1], who suggested a privacy aware EDR design using something called time-release encryption, where the data stays locked until a certain time or condition is met. This was one of the first papers that treated EDR privacy as an actual cryptography problem instead of just a policy or legal issue, and it's honestly the closest thing to what NAEBA is trying to do, just with a different method of locking the data. Around the same time, Chim and other researchers [2] pointed out a different problem, what happens if the EDR device itself gets destroyed in a bad crash along with all the evidence in it. Their solution was to send the event data to nearby roadside units over a network so there's a backup, while keeping it privacy-preserving using their own protocol.

Later on, around late 2010s and into the 2020s, two new directions came up in this research area. First, Vieira and their team [4] built something called BEDR which uses blockchain instead of one single device, spreading the crash data across many nearby vehicles using a consensus system, and they actually tested it on a platform called Hyperledger Iroha. This is very different from the dual-key model I am proposing in NAEBA, but it's a good comparison because their method is more resistant to hardware being destroyed, even though it's harder to audit legally compared to a simple two-key system. Second, Kurachi and their team [5] looked at EDR data as something useful for digital forensics, not just physical crashes, and showed that this data can also help figure out if there was a cyberattack on the vehicle's internal network, which connects well with the tamper detection feature I have included in NAEBA.

For the actual cryptography part, the key splitting idea in NAEBA comes from Shamir's secret sharing method from 1979 [3], which is still basically the standard way whenever you want multiple people to be needed together before a secret can be opened. For encrypting the video, itself, I am using AES-256-GCM which is a NIST approved method that also checks the data hasn't been changed [6][7]. On the legal side, India's DPDP Act from 2023 is the reason the 30-day auto-delete idea exists in NAEBA, since that law is about not keeping data longer than necessary [8]. Section 65B of the Indian Evidence Act, along with the newer Bharatiya Sakshya Adhiniyam, is what decides whether this kind of digital evidence would even be allowed in court [9][10]. All of these papers and laws together are basically what NAEBA is built on top of.

## III. METHODOLOGY

### 3.1. System Architecture

NAEBA is meant to work as a software layer on top of hardware the car already has, so it's not like you need a whole new set of sensors installed. It just listens quietly on the vehicle's CAN FD bus and uses the existing ADAS camera feed. There are five main parts to how it works: capturing and encoding the video, encrypting it as it's being recorded, storing it safely with wear leveling, the dual-key access system, and finally an extraction process that only turns on once an FIR has actually been filed.

### 3.2. Video Capture and Storage

The video from the ADAS cameras is encoded using H.265 at 720p and 15 FPS, and it just keeps recording over itself on a loop. For storage, I'm specifying automotive grade NAND flash, either JEDEC eMMC 5.1 or UFS 3.1, rated for extreme temperatures because India has such a huge range, from very hot places like Rajasthan to very cold places like Ladakh. Since this storage gets overwritten constantly every 30 days, wear leveling becomes very important, so a wear leveling algorithm spreads out the writing evenly across the flash memory so it lasts as long as the car is expected to be used.

### 3.3. Encryption and Key Management

As soon as the video is being encoded, it gets encrypted right away using AES-256-GCM [6][7], and the key for this is generated fresh every time the engine starts, using a random number generator inside a Hardware Security Module. That key is then wrapped using RSA-2048, and this is really the main privacy feature of the whole system, because the private key is never stored as one whole piece anywhere. It's split using Shamir's secret sharing method [3] into two parts, a Hardware Share that stays permanently inside the car and can never be taken out on its own, and a Judicial Share that is kept separately by a government gateway under the Ministry of Law and Justice. Neither piece works on its own, decryption only happens when both parts are combined, and that only happens through the extraction process explained next.

### 3.4. Evidence Extraction Workflow

The whole extraction process is designed to be fully trackable. First an officer files an FIR through CCTNS, then the vehicle's registration and VIN get checked against VAHAN, and then a digital token carrying the Judicial Key Share gets created for just the specific time window that's needed, not the entire 30 days. This token is loaded onto an official OBD-II tablet used by police, which connects to the car directly. The car's HSM checks if the token's signature actually matches the Ministry's public key, and only if it matches does it combine the Judicial Share with its own Hardware Share to rebuild the key. Even after that, it only opens up the specific time window that was requested, everything else stays locked.

### 3.5. Low-Bandwidth Integrity Auditing

One easy way someone could try to cheat this whole system is by just unplugging the camera. To stop that, NAEBA sends a very small 1 KB message once every time the engine starts, using the car's existing eSIM connection. This message only contains a hashed version of the VIN, not the actual number, along with the system's health status and a tamper flag, and nothing about location, which matches the data minimization idea behind India's DPDP Act [8]. If the tamper flag ever comes back as true, VAHAN can flag that vehicle for a physical inspection. Any firmware update to NAEBA itself needs to be signed by both the car manufacturer and the government authority together, following UN Regulation No. 156 [11], so that one single compromised party can't push a bad update to every vehicle at once.

## IV. IMPLEMENTATION

I want to be honest here, this paper is presenting NAEBA as an architecture and design on paper, it is not something I have actually built and tested on real hardware. Getting this properly integrated at the car manufacturer level would take years and involve a lot of people, way more than what one student can realistically do alone. What I think is actually realistic as a next step is doing this in three phases. First, building just a software version of it, maybe on something like a Raspberry Pi with a fake CAN bus feed, just to prove that the encryption and key splitting logic actually works correctly from start to end. Second, building an actual small hardware version using a real HSM chip and automotive flash storage to check if the real-world constraints match what I've assumed. Third, if a manufacturer or state transport

department is actually willing to work on this, doing a small pilot with a limited number of vehicles. For writing the actual software, I am planning to use Rust mainly because of its memory safety, which really matters when the code is running close to a car's safety systems.

## V. RESULTS

Since NAEBA is only a proposed design and not something actually deployed, I don't have real field data to show. What I can show instead are the numbers I got from actually working through my own design, and honestly one of these numbers showed me I had made a mistake earlier, which I think is worth showing instead of hiding.

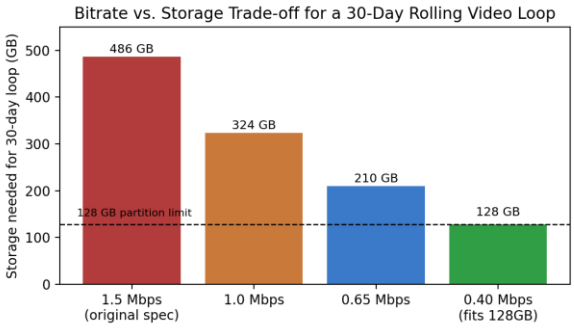


*Figure 1: Storage needed for a full 30-day rolling video loop at different bitrates, compared with the 128 GB storage I had originally planned.*

At the bitrate I had originally planned, around 1.0 to 1.5 Mbps at 720p and 15 FPS, a full 30-day loop actually needs somewhere between 324 GB and 486 GB, not 128 GB like I had written in my early notes. That's a pretty big gap, not just a small rounding error. To actually make 128 GB work, the bitrate would need to come down to around 0.40 Mbps, which is honestly a very aggressive compression for footage that still needs to be readable, or the storage needs to go up to somewhere around 350 to 500 GB instead. I think the more realistic fix is a mix of both, using motion-triggered recording so it doesn't waste storage recording nothing happening, along with slightly bigger storage than what I first planned.

On the cryptography side there isn't much to question honestly. AES-256-GCM and RSA-2048 are both extremely well tested and there are no known practical ways to break them right now, and Shamir's secret sharing method makes sure that even if someone steals just the police tablet, or somehow gets access to just the car's HSM alone, they still get nothing useful. That's really the whole point of this design, no single person, not even the car owner or a single police officer, can unlock the footage by themselves.

## VI. DISCUSSION AND CONCLUSION

NAEBA is my attempt at solving a real problem, India doesn't currently have any standard way of recording vehicle evidence that is both hard to tamper with and also respects privacy. I don't think the real contribution here is any one specific cryptography method, since AES, RSA, and Shamir's scheme are all already well-known techniques, I think the contribution is combining them into one workflow that actually connects to systems India already has like VAHAN, CCTNS and eSign, instead of asking for a whole new surveillance system to be built from scratch.

That said, I know this paper has real limitations and I'd rather say that clearly than have someone else point it out. None of this has actually been tested on real hardware yet.

The storage versus bitrate problem in the Results section shows that my own original numbers were wrong when I actually worked through them properly, which is a good reminder that things look different on paper compared to when you actually calculate them. Getting any car manufacturer to adopt something like this would take years of testing and regulation that is way beyond what this paper is trying to do. The honest next step is building the small software prototype I mentioned in the Implementation section, then getting the cryptography properly reviewed, and only after that would it make sense to talk about actually deploying this somewhere.

Even with all these limitations, I think this paper still shows something useful, that it is actually possible to design a vehicle evidence system that stays private by default but still works legally when it's actually needed, which is more than what current dashcams or CCTV systems can offer right now.

## ACKNOWLEDGMENT

The bitrate versus storage chart in Figure 1 was calculated by me and made into a graph using an AI visualization tool (Claude, Anthropic), but the actual numbers and what they mean in the Results section are

my own work. This paper was written under the guidance of Dr. Swapnesh Taterh.

## REFERENCES


[1] C. Patsakis and A. Solanas, "Privacy-aware event data recorders: Cryptography meets the automotive industry again," IEEE Communications Magazine, vol. 51, no. 12, pp. 122–128, 2013.

[2] T. W. Chim, S. M. Yiu, C. Y. Yeung, V. O. Li, and L. C. Hui, "Secure, privacy-preserving, distributed motor vehicle event data recorder," in Proc. 2013 Int. Conf. Connected Vehicles and Expo (ICCVE), Las Vegas, NV, USA, 2013, pp. 337–342.

[3] A. Shamir, "How to share a secret," Communications of the ACM, vol. 22, no. 11, pp. 612–613, 1979.

[4] A. R. Vieira, C. M. Farias, W. S. Melo, and E. L. Madruga, "BEDR: Blockchain Event Data Recorder," in Proc. 2022 IEEE 25th Int. Conf. Intelligent Transportation Systems (ITSC), 2022, pp. 2716–2721.

[5] R. Kurachi, T. Katayama, T. Sasaki, M. Saito, and Y. Ajioka, "Evaluation of automotive event data recorder towards digital forensics," in Proc. IEEE 95th Vehicular Technology Conference (VTC2022-Spring), Helsinki, Finland, 2022, pp. 1–7.

[6] National Institute of Standards and Technology, "Advanced Encryption Standard (AES)," FIPS PUB 197, 2001.

[7] M. J. Dworkin, "Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC," NIST Special Publication 800-38D, 2007.

[8] Government of India, "The Digital Personal Data Protection Act, 2023," Ministry of Electronics and Information Technology, 2023.

[9] The Indian Evidence Act, 1872, Sec. 65B (as amended).

[10] Government of India, "The Bharatiya Sakshya Adhiniyam, 2023."

[11] United Nations Economic Commission for Europe, "UN Regulation No. 156 – Software Update and Software Update Management System," 2021.