

USENIX

THE ADVANCED COMPUTING SYSTEMS ASSOCIATION

BLE Theft Auto: Evaluating the Security of Aftermarket BLE-based Automotive Remote Control Systems

Jerry Yu, Yibo Wei, Sumanth Rao, Mohak Vaswani, Jefferson Chien, Christian Dameff, Nishant Bhaskar, and Aaron Schulman, *UC San Diego*

https://www.usenix.org/conference/usenixsecurity26/presentation/yu-jerry

This paper is included in the Proceedings of the 35th USENIX Security Symposium.

August 12–14, 2026 • Baltimore, MD, USA

ISBN 978-1-939133-58-8

Open access to the Proceedings of the 35th USENIX Security Symposium is sponsored by



جامعة الملك عبدالله للعلوم والتقنية

King Abdullah University of Science and Technology

ARTIFACT EVALUATED
USENIX
AVAILABLE

BLE Theft Auto: Evaluating the Security of Aftermarket BLE-based Automotive Remote Control Systems

Jerry Yu* Yibo Wei* Sumanth Rao Mohak Vaswani Jefferson Chien
Christian Dameff† Nishant Bhaskar Aaron Schulman
UC San Diego

Abstract

Vehicles can increasingly be remotely controlled through smartphone apps that function like wireless key fobs, unlocking the vehicle's doors and even starting the engine. These systems use Bluetooth Low Energy (BLE) to provide offline remote control capability. However, they often do not use BLE's built-in authentication and encryption; instead, they implement their own application-layer security, introducing a new attack surface. In this paper, we explore a previously understudied segment of BLE remote control systems: aftermarket systems installed by car dealers or individuals who want to add BLE remote control capability to their vehicles. We discovered six aftermarket BLE remote control systems in use today and analyzed their application-layer security. We identified critical vulnerabilities in three such systems, two of which are installed by car dealers, exposing what we estimate to be 1.4 million vehicles to attack. We further show that targeted BLE remote attacks are practical: an attacker can use crowdsourced Bluetooth scanning databases to target specific vehicles with vulnerable systems installed. Our disclosures prompted an aftermarket vendor to significantly improve BLE security in their dealer-installed system.

1 Introduction

Vehicles increasingly include Bluetooth Low Energy (BLE) connectivity to enable smartphone-based remote control even when offline. Smartphone applications now function like key fobs and can directly control the most sensitive features of a vehicle over BLE, such as door locks, the immobilizer, and the engine starter. These systems are now widely deployed: Several automotive manufacturers (OEMs), such as Tesla and Rivian, have BLE-based remote-control systems in their vehicles. The security of OEM systems has been scrutinized, through public standardization [6] and reverse engineering [48].

Far less understood than OEM systems are aftermarket BLE remote control systems. These products add BLE connectivity to older or lower-end cars. Unlike OEM systems, they have received little attention from security researchers. This gap is concerning: In 2018, a researcher found that one aftermarket BLE remote system had a critical vulnerability that allowed unauthorized smartphones to control it [42], which we confirmed remains present in the product today.

Scrutinizing BLE vehicle remote control systems is necessary as they forgo BLE's standardized and tested security mechanisms [10,21,23,36,43,49] and instead implement their own custom application-layer security protocols. This choice is driven by the need for several required features that are not provided by smartphone BLE APIs, including: (1) revoking pairings, (2) sharing pairings, and (3) preventing co-located malicious apps from accessing paired vehicles [38].

Unlike the OEM ecosystem, the aftermarket ecosystem is opaque: products are not generally sold directly to consumers but are instead distributed by installers through a relatively closed ecosystem. Also, many aftermarket systems do not even advertise that they use BLE connectivity. Owners may not know their vehicle has an aftermarket remote control system; these systems are often hidden behind a car's dashboard and spliced into the car's most sensitive wiring (e.g., CAN bus and ignition wires), effectively bypassing the car's built-in security system. If attackers can target vehicles with vulnerable aftermarket systems, they can steal the victim's car by unlocking the doors while it is running and performing carjacking, or remotely starting the car while it is parked.

1.1 Our Work

This work investigates the feasibility of targeted attacks on aftermarket BLE vehicle remote control systems. Our key insight is that these systems broadcast BLE advertisements, enabling identification and targeting via BLE wardriving — both locally to find nearby targets and globally via public crowdsourced wardriving databases (e.g., WiGLE [4]), where BLE remote control systems have been hiding in plain sight.

*These authors contributed equally to this work.
†Primarily affiliated with UC San Diego Health.

We conducted local wardriving scans in Southern California and discovered an aftermarket BLE remote control system, KARR, pre-installed by car dealers in a majority of cars sold in the southwestern U.S. We also discovered five other similar aftermarket systems, including: CarLink, Rockledge, DS4, EvoStart, Linkr, deployed across the U.S. and Canada. We reverse engineered all of their application-layer BLE protocols and discovered that each system implements its own pairing, authentication, and encryption. Two systems had robust security features, roughly comparable to OEM protocols.

Three systems (KARR, CarLink, and Rockledge) had critical pairing and authentication vulnerabilities, allowing attackers to gain unauthorized control over the vehicles. One system pre-installed by car dealers used a shared pairing key, making all installations vulnerable to an attacker who could unlock and immobilize them from 3–10 m away. Another remote-start system allows an attacker to unlock and remotely start a vulnerable car using the vendor's official app from up to 150 m away. We developed several short-term and long-term mitigations, and disclosed the vulnerabilities and their mitigations to their respective vendors.

To evaluate the deployment scale of these vulnerable BLE systems, we developed a new methodology using sequential MAC address patterns. We estimate that at least a million vehicles have the vulnerable dealer pre-installed system, and roughly half of those owners do not know they have an active system in their vehicle because they did not purchase it. These systems are not only found in the southwestern U.S., but are also resold and transferred throughout the rest of the U.S. and even around the globe. We also estimate that tens of thousands of vulnerable remote-start systems are deployed, predominantly in Canada. Even though the deployment is sparse, we found it may be possible for an attacker to target them by searching in wardriving databases.

This paper makes the following contributions:

1. **Identifying aftermarket BLE vehicle control systems (§3)** by combining wardriving data, FCC filings, and reverse engineering.
2. **Analyzing application-layer BLE protocols (§4)** of six different aftermarket BLE remote control systems, leading to the discovery of vulnerabilities in three aftermarket BLE systems.
3. **Demonstrating the feasibility of attacks (§5)** that enable controlling door locks, immobilizer, engine starter, lights, and horn from a distance of tens of meters, even targeting sparsely deployed vulnerable vehicles with crowdsourced wardriving data.
4. **Estimating the scale of vehicular BLE deployments (§6)** based on sequential MAC-based estimation. We estimate that over a million vehicles have vulnerable aftermarket remote control systems, all of which need to be manually updated with patched firmware.

1.2 Ethical Considerations

This work exposes vulnerabilities in potentially over a million vehicles equipped with KARR, tens of thousands equipped with CarLink, and future Rockledge devices as their deployment grows. To mitigate the potential for exploitation at this scale, we do not provide the details on how to exploit the pairing, authentication, and encryption vulnerabilities we discovered in this work. We also practiced responsible disclosure to the vendors of KARR, CarLink, and Rockledge. Additionally, we filed a report with the the US National Highway Traffic Safety Administration (NHTSA) describing the vulnerability in KARR. We also gave all vendors involved sufficient time to implement and deploy mitigations by embargoing the paper until the conference in August, 2026.

Our considered view is that the benefits of our research outweigh the potential risks to publication. The aftermarket BLE remote control industry has had several adaptations over the years, but its security has yet to be scrutinized. On several online forums (e.g., Reddit), consumers have questioned the security of aftermarket systems after discovering their cars have them, especially if they paid to activate the system for their personal use [32]. Appendix A provides a detailed discussion of the ethical considerations of this work.

KARR Disclosure: We disclosed the vulnerabilities to KARR and their remote control system developer on January 21st, 2025. During the meeting with their representatives, we demonstrated the attack in person and had an hour-long discussion with them about possible mitigations.

CarLink and Rockledge Disclosure: The company that sells CarLink appears to be both the direct vendor of CarLink and the developer of Rockledge's firmware and mobile application. Therefore, we disclosed the vulnerabilities for both products to CarLink on July 31st, 2025. We had a video conference to discuss the vulnerabilities and mitigations they should consider. However, we did not receive an update on their mitigation plan, so we also disclosed the vulnerabilities to Rockledge directly on December 17th, 2025.

We describe mitigation strategies for current and future aftermarket BLE remote systems in Section 7. Updates on disclosures can be found at the website:

`https://bleauto.sysnet.ucsd.edu`

2 Background

We begin by describing how vehicle BLE remote-control systems securely enable offline control of sensitive components. These systems are given privileged access, similar to a standard vehicle's wireless remote, to control locks, engine ignition, engine immobilizer, lights, and the horn. The standard BLE security mechanisms do not provide the features required by these systems, so manufacturers choose to implement their own application-layer alternatives. Hence, we

describe the challenging security requirements that must be met to enable remote control of such sensitive components via application-layer protocols. We discuss two ecosystems: (1) how OEM remote control systems meet these challenging security requirements, and (2) what open questions aftermarket remote control systems raise.

2.1 BLE Remote Control Security

Designing BLE-based automotive remote control modules poses many security challenges. Manufacturers often need to balance security, robustness, and user convenience. The goal is to ensure that only commands issued from *trusted devices* are accepted and executed, while simultaneously making it easy to add and remove trusted devices. There must be a secure implementation for pairing, authentication, and encryption. A vulnerability in any part of the system allows an attacker to gain unauthorized control of the vehicle.

Pairing/Unpairing: Trusted user devices (e.g., smartphones) are registered to the BLE control system. These systems typically also allow a primary user to add other trusted users to control the vehicle. The ability to add new users is indeed a key benefit of phone-based remote controls. Attackers should not be able to initiate pairing to add malicious devices and gain unauthorized access.

Authentication: Every command sent to control the system must be authenticated. This requires verifying that the message originated from a trusted device and that the command is fresh. Attackers should not be able to bypass authentication to impersonate a trusted device.

Encryption: All messages exchanged between the remote control system and the trusted user's device must be encrypted, to prevent replay attacks and to prevent unauthorized capture of authentication credentials.

2.1.1 BLE pairing is incompatible with vehicle control

The BLE protocol provides standardized security features for secure pairing, authentication, and encryption. However, the standard pairing protocol does not provide sufficient flexibility to meet the requirements of automotive remote control systems. Because BLE connections are managed at the OS level, smartphone apps have a very limited set of APIs for properly adding, validating, and revoking BLE device pairings.

Cloud-based access control: Vendors aim to maintain control over vehicle access through their own cloud infrastructure. By centralizing authorization, vendors can dictate exactly which user is permitted to access a specific vehicle at any given time. However, neither Android nor iOS provides APIs for apps to unpair a BLE device. Consequently, to revoke a user's access to a vehicle, the user must either manually delete the paired vehicle in the OS Bluetooth settings menu or a primary user must revoke access while within BLE proximity of the vehicle. Neither option is feasible for cloud-based access control. Consequently, many vendors implement their own security protocols rather than adopting BLE's built-in security to manage vehicle access remotely from their cloud servers.

Validating trusted devices: BLE's built-in pairing validation methods (ensuring there is no attacker-in-the-middle during key exchange), such as 'Numeric Comparison", require visual confirmation on an in-car display. However, most of these vehicle remote control systems lack a physical display. Even when such I/O devices are available in the vehicle, requiring users to visually verify codes while pairing introduces friction, because the pairing can only be performed while inside the car.

Co-located app attacks: Prior work found that co-located apps can exploit the OS to access any existing paired BLE devices [38]. This is concerning for vehicle BLE remote control, as it enables attackers to control people's vehicles through a compromised smartphone application.

Due to these pairing limitations, vehicle remote control systems often do not use the BLE protocol's security features and instead "roll their own" application-layer security [38]. This introduces the potential for new vulnerabilities, as the BLE stack is already well-tested, and vulnerabilities have been found and fixed in pairing [23, 43, 49], authentication [21, 36], and encryption [10].

2.2 Threat Model

The primary threat is an attacker using a malicious smartphone to gain unauthorized access to vehicles through insecure BLE remote-control systems. Their goal is to exploit weak application-layer BLE protocols to gain control of the car to perform any of the following criminal acts: (1) breaking into the victim's car by unlocking the doors while it is running (e.g., carjacking), or while it is parked (2) preventing the victim from moving their car by activating the engine immobilizer; (3) causing the victim distress or distraction by remotely activating the car's horn and lights while the victim is driving.

We assume an attacker has access to a proof-of-concept exploit against a vulnerable BLE remote control system. These could be malicious apps developed by motivated car theft groups who understand the car security ecosystem (e.g., former installers or employees of the companies). The attacker is not required to know how the system works to attack a vehicle; they just need a malicious app running on their phone or laptop that performs the attack.

Finding vehicles with vulnerable BLE remote control capability, however, can be difficult. While OEM systems can be identified by a car's make and model, aftermarket systems can be installed in any vehicle. Attackers can find target vehicles by wardriving or looking for visual indicators.

Once they find a target, the attacker can approach the vehicle and try to control it over BLE. With a pairing vulnerability, they can take control of the car immediately or when pairing mode is automatically enabled (i.e., just after engine start) using a malicious BLE-based application on their smartphone. With authentication or encryption vulnerabilities, the attacker may need to wait until the victim uses the legitimate application to send BLE commands to the vehicle. Then, they will capture the commands with a sniffing device (e.g., a laptop with an nRF52840 BLE sniffer dongle), and then either replay them, or use the captured information to bypass authentication or to forge authenticated or encrypted control messages. This may take some time, but attackers are motivated to do so because the resulting access enables desirable criminal actions that are difficult to achieve by any other means, especially in the covert way BLE control enables.

2.3 OEM Systems

OEMs have standardized BLE-based smartphone vehicle control through the application-layer protocol called CCC "Digital Key" [6, 7], an industry standard adopted by automakers such as Audi, BMW, and Hyundai [3]. Additionally, OEMs like Tesla have developed their "Phone Key" protocol on a similar architecture. Both these protocols implement their application-layer BLE security as follows:

Pairing/Unpairing: Pairing a new phone with an OEM remote control system requires physical interaction inside the vehicle and generally requires the presence of the car's electronic keys. For instance, Tesla requires the user to tap their NFC keycard on the center console to complete the pairing transaction[1]. Pairing vulnerabilities have been the most significant problem with these systems. Tesla had a vulnerability in which BLE pairing mode could be entered without the owner's involvement, and the owner could unknowingly authenticate a malicious pairing request when they tapped their NFC card on the doorframe to enter the car [31]. Tesla partially mitigated this by adding an option that requires the user to enter a PIN on the car's screen to drive it.

Authentication: OEMs use asymmetric cryptography for challenge-response authentication. Each phone has a unique private key, generally stored securely in its trusted execution environment. During pairing, its public key is added to the car's remote control system's trusted list. The primary attack against BLE authentication has been relay attacks [48], in which an attacker relays authentication messages from a trusted phone far away from the car to the car itself.

[1]Tesla's manual indicates the presence of a phone-key pairing mechanism that only requires an authorized user to be in proximity of a vehicle without the need of a keycard. The NFC keycard pairing mechanism (as described above) is used as a fallback. However, this is yet to be reverse-engineered and verified.

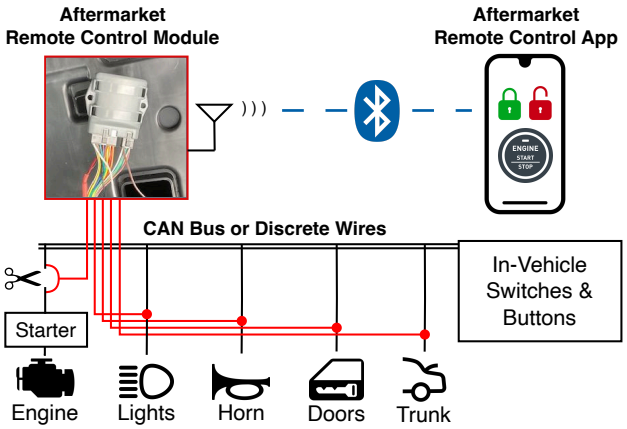


Figure 1: Aftermarket BLE vehicle remote control. The hardware implant is connected to the ignition, horn, headlights, door locks, and trunk locks by tapping or cutting into the factory wiring. The user's mobile application issues commands to the remote control module via BLE.

Encryption: Most OEM systems use ECDH to derive symmetric session keys for message encryption such as AES-GCM with 128-bit keys (or similar). The GCM mode makes it more difficult to perform replay attacks.

2.4 Aftermarket Systems

Aftermarket vendors sell add-on hardware modules that integrate BLE remote control into cars without built-in functionality. The hardware add-on is often a basic microcontroller with automotive I/O (e.g., PIC 18F26K80), a BLE module and optionally, a cellular module for online remote control.

Installing an aftermarket system requires splicing a remote control module directly into sensitive ignition wiring under the dashboard and the "Body/Convenience" CAN bus, which controls the door and trunk locks and activates the lights and horn. This is necessary because vehicles do not provide an easy-to-use interface for controlling sensitive features, as these features are not accessible via the OBDII port. Essentially, this makes an aftermarket system a legitimate hardware implant that bypasses the car's built-in security system (Figure 1). One would expect aftermarket systems to provide application-layer security similar to those of OEM BLE remote control systems to compensate for bypassed security measures. This is the first work to observe the security of the aftermarket ecosystem. Indeed, in this work, we identified numerous critical vulnerabilities stemming from insecure design of their proprietary application-layer BLE security protocols.

Aftermarket remote engine starters must also bypass the Passive Keyless Entry and Start (PKES) security feature in most modern vehicles. PKES requires a physical remote keyfob to be present in the car in order to start the ignition. Remote starters capture a challenge-response exchange with the

keyfob, and a virtual keyfob is created using an online service. Then, without the keyfob in the car, the virtual key produces a valid PKES exchange to allow the car to start.

2.4.1 Open Questions About Aftermarket Systems

Discovery: *How do we discover what aftermarket systems exist today?* Aftermarket BLE remote control systems are a closed ecosystem: They are often purchased and installed only by specialized installers[2]. Also, these modules are hidden: They are often installed under the dashboard and not visible even to the driver.

Scale: *How many aftermarket remote control modules have been installed in cars?* The threat posed by aftermarket systems could range from a few car modders upgrading inexpensive cars to aftermarket systems installed en masse across entire vehicle fleets.

Security Analysis: *What application-layer protocols are aftermarket systems building on BLE?* Prior to this work, only one aftermarket BLE remote control protocol (CarLinkBT) had been publicly reverse engineered, and its application-layer protocol was found to use a single shared key for pairing/authentication across all devices [42]. When we studied the product, we found that it is still vulnerable to the same attack after seven years. This is evidence of how hidden this ecosystem is and how vulnerable it can be, which motivates our study. We found that, indeed the entire ecosystem has similarly trivial vulnerabilities, and deployments of vulnerable devices are at a scale much larger than expected, even affecting vehicle owners who do not know that an aftermarket system exists in their own car.

Targeting: *Can an attacker feasibly target vehicles with aftermarket remote control systems?* There are potentially few installs compared to the number of cars on the road. Therefore, it would be impractical for an attacker to simply try to attack as many cars as possible, hoping that some are vulnerable.

Mitigation: *Once vulnerabilities are found, how are the fixes deployed, especially for offline devices with only BLE connectivity?* There may be inherent risks that cannot be mitigated at a large scale of vulnerable deployments.

3 Discovering Aftermarket Systems

We used an iterative search process to discover aftermarket BLE control products. We start by performing a broad search for all BLE systems operating in vehicles using wardriving data collected on highways. Then we use FCC documents to fingerprint devices based on the names observed in the scans. Then, we perform a focused search of smartphone app stores for other apps with similar properties (e.g., the same vendor) as the systems we found in scans. We also reverse engineer those apps to see if they contain any code that references other related apps. In total, we discovered 6 unique aftermarket BLE remote ecosystems. We also found that the aftermarket ecosystem is more interconnected than expected: different vendors share software developers and hardware platforms.

We now describe how we performed each of those steps, and what aftermarket systems we found. This process is repeatable and should be performed periodically to monitor for vulnerable systems and catch them before they are widely deployed. Even during the course of our study, a vendor deployed a new aftermarket BLE remote control system. We quickly identified and disclosed a potential vulnerability to the vendor (Section 4.3).

3.1 Wardriving Search

We conducted wardriving to collect BLE advertisements from devices encountered along major highways across San Diego, CA. Our hypothesis was that the most popular BLE remote control systems would appear in these scans if they existed. This resulted in 4,314 unique devices (i.e., name and MAC address). We found two aftermarket and three OEM systems.

BLE Wardriving App: We built a BLE wardriving app that captures complete raw advertisement packets from BLE devices, along with GPS locations where they were seen. Raw advertisements (i.e, name, MAC address, service UUIDs, manufacturer IDs) provided sufficient information to cluster devices and identify the most commonly seen products.

Highway Filtering: We filtered the BLE advertisements to focus on detections occurring near major roadways. Our insight was that advertisements on highways are likely to come from BLE devices in vehicles. We did this in post-processing by using OpenStreetMap (OSM) data to identify roads tagged as "motorway," and we filtered our data to only report devices within a 50-meter buffer zone around motorway centerlines. OSM's motorways are major highways like freeways and interstates, so this should inherently filter out devices from residences, businesses, and other types of transportation systems (e.g., bicycles and scooters). Indeed, OSM defines a "motorway" as a grade-separated road that does not allow access to pedestrians and bikes [9]: Only cars and trucks can use them. With filtering, we reduced the number of unique MAC addresses from 372,922 to 4,314.

BLE advertisement clustering: After geographically filtering the wardriving data, we clustered devices with structurally similar BLE advertisement packets. We used DBSCAN [16] with a series of distance functions, each focusing on a different property. We apply each distance function hierarchically on the clusters produced by the previous distance function. We applied the following three stages of clustering:

1. **Advertisement Fields:** First, we cluster raw advertisements based on the fields present (e.g., Flags,

[2]Only one of the remote control systems we studied was available on Amazon U.S.

Count	BLE Name Pattern	BLE Product
1506	S****************C	Tesla - OEM
608	**QT **********	**KARR BT - Aftermarket**
178	Rivian Phone Key	Rivian Phone Key - OEM
71	HarpBT*********	Titan truck ELD
68	LE-Bose SoundLink	Bose speaker
63	**DR **********	**KARR Cell - Aftermarket**
50	Govee_H6199_****	Govee light
44	MyQ-***	MyQ garage opener
42	[TV] Samsung 6 Series (55)	Samsung TV
30	GM_PEPS_VKM	GM Passive Entry - OEM

Table 1: Top 10 BLE products seen in wardriving scans from highways in a large US city. The highlighted products are BLE remote control systems. This study discovered KARR, a widely deployed aftermarket system.

Manufacturer-Specific Data, Service UUIDs, etc.) using a Jaccard-like distance. This coarsely partitions devices based on their advertisement structure.

2. **Advertisement Field Ordering:** We sub-cluster based on the sequential ordering of the advertisement fields in the advertisement packet. Since the advertisement field ordering is usually hard-coded in the BLE firmware, each cluster has similar firmware implementations.

3. **Advertised Name Prefixes:** We perform a final round of clustering based on the similarity of BLE device name prefixes. We compare every pair of device names using a distance metric based on the difference in their prefixes. This is useful because BLE devices from the same vendor and model often have names that follow the standard pattern `[Product Identifier]-[Serial Number]`.

Product fingerprinting: After clustering, we attribute advertisement clusters to specific products using multiple fingerprinting methods, including device name searches, manufacturer ID and UUID lookups, IEEE OUI MAC prefix analysis, and FCC document searches for advertised names. The U.S. Federal Communications Commission (FCC) publicly posts the instruction manual for every wireless product certified for sale in the United States. In those manuals, vendors often describe how to identify their specific BLE device by its advertised name prefix (e.g., `BH-` for devices with names like `BH-24512`). We use the `fccid.io` site to search all these manuals to see whether the BLE device name clusters we identified in the previous step appear in any of them.

3.2 Results

Our clustering analysis identified 267 clusters of BLE advertisements seen on highways. Table 1 shows the top 10 largest clusters. Notably, most of these clusters correspond to OEM vehicle remote-control systems, including Tesla's and Rivian's Phone Key systems. This demonstrates that the method can detect vehicle remote control systems when they are widely deployed. We fingerprinted all of the remaining BLE device clusters using FCC manual searches.

KARR Revealed: We discovered KARR—an aftermarket BLE remote control system with a deployment density similar to Tesla and Rivian's Phone Key systems. This system was well hidden: the FCC filing obscures the purpose of this device in multiple ways: The title refers to this system as a "Bluetooth Sensor"; the company that makes the hardware has a name similar to the BLE device name, but in the manual, the vendor that markets this system is a subsidiary of a large publicly traded insurance company, Acrisure. The `QT` name prefix matched the title and the section of the manual where the vendor describes how to search for the device.

Based on references to the name prefixes in KARR's manuals we found that KARR's remote control module comes in two models: one is BLE-only (`QT`) and the other is BLE+Cellular (`DR`) adding online control.

KARR is so widely deployed because many car dealerships install it in every car they put on their lots. The system provides dealers with lot management features, such as smartphone-based access to unlock cars for test drives; it also enhances car security on their lot by introducing an immobilizer that can only be controlled via BLE with the KARR app. Dealers also benefit financially from this system: They resell the modules as upgrades that provide anti-theft protection and BLE control on cars that do not otherwise have it. Additionally, they may receive a discount on their car insurance due to the addition of the immobilizer. We found that KARR is sold by dealerships for Kia, Honda, Toyota, Mazda and Ford.

3.3 App Search

For the remote systems we were less likely to encounter on the highway in San Diego, we searched for aftermarket vehicle control apps in the Google Play and Apple App Stores. This process uncovered five other aftermarket systems, including another dealer-installed remote control ecosystem that just began deployment at the time of publication of this paper.

Rockledge Revealed: We searched for similar companies to KARR that sell dealer-installed remote control systems and found the Rockledge app, which provides similar dealership functionality to KARR. This app, however, was built by a third-party contractor that develops several different BLE remote control apps for different vendors. This specific model we found is a BLE upgrade that can be added to Rockledge's existing non-BLE remote control module.

CarLink Revealed: The same developer who developed Rockledge's mobile app also developed CarLink's app. The BLE control library shares some source code with the Rockledge application, but provides a different set of commands because CarLink is designed for consumer upgrades rather

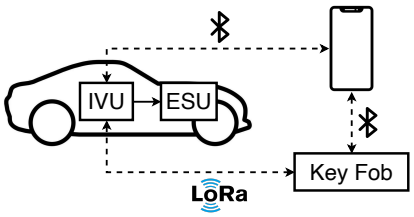


Figure 2: The CarLink architecture. The In-Vehicle Unit (IVU) controls a compatible Engine Starter Unit (ESU).

than dealer-installed lot management. Specifically, it is a long-range extender upgrade for many compatible remote engine starter products [44]. Unlike Rockledge, the CarLink system can be purchased publicly through online markets. As shown in Figure 2, the CarLink in-vehicle unit (connected to a compatible engine starter) accepts LoRa-based commands from a proprietary key fob to facilitate long-range control. A mobile application can also transmit BLE-based commands directly to the in-vehicle unit or to the key fob, which acts as a BLE-to-LoRa relay.

DS4, EvoStart, and Linkr Revealed: The reverse-engineered CarLink application code contained a variety of protocol and product names, which led us to discover three additional aftermarket BLE remote control systems: DS4, an all-in-one remote start system with built-in BLE control; EvoStart and Linkr, two products similar to CarLink that add BLE remote control to existing remote start systems.

Summary: We discovered six aftermarket BLE remote control systems, ranging from dealer-installed systems to remote-control upgrades for consumer-installed remote-starting systems. This study also shows that the aftermarket remote control ecosystem is surprisingly interconnected. This means that vulnerabilities discovered in one vendor's products may also affect other vendors using the same shared components. Next, we investigate the application-layer BLE security approaches implemented in each of these aftermarket remote-control ecosystems.

4 Security Analysis

In this section, we systematically analyze the security of the application-layer BLE remote control protocols used by all six aftermarket systems we discovered in Section 3. For each product, we evaluate its implementation of pairing, authentication, and encryption.

Table 2 summarizes how all six products implement BLE security and their functional capabilities. We also provide a comparison with Tesla's secure BLE remote control system. Three of these systems had fatal security flaws, including: hard-coded authentication and encryption keys, a pairing mechanism that is routinely activated without explicit user requests, and always allowing unauthorized connections without performing pairing. We were able to experimentally validate two of these vulnerabilities on running hardware (Section 5). in Section 5.

4.1 Methodology

For each of the six ecosystems, we use the following process to analyze the application-layer protocol and check for vulnerabilities. Our methodology is similar to that used to discover the first vulnerability in an aftermarket system [42].

4.1.1 APK reverse engineering

We started by reverse engineering the Android APK files for each remote control application. For the applications primarily written in Java/Kotlin, we decompile them using JADX. We discovered one application where the BLE protocol was implemented in C# and included in the APK as a .NET DLL, which we separately decompiled using ILSpy [8]. To the best of our knowledge, we are the first to discover the use of a non-Java library to implement a BLE protocol within an Android application. Future Android BLE research should take such implementations into consideration.

We analyzed the decompiled code to understand their application-layer protocol. Specifically, we scrutinized the custom BLE protocol to identify hard-coded credentials, insecure key-exchange methods, and improper handling of sensitive data. For dealer-installed products, we also check whether the application supports switching between dealer and user modes and, if so, whether it requires additional authentication.

4.1.2 Remote control module observation

To analyze parts of the protocol that were not revealed by the application code, we procured and ran each remote control module and instrumented its APK to observe its interactions with the application. We observed parts of the protocol implemented only in the module's firmware, including the secrets used for authentication and the module's pairing behavior. We also tested the conditions under which the module accepts pairing. Specifically, for dealer pre-installed products, we also checked if pairing is possible on deactivated modules.

4.2 KARR

KARR is a dealership pre-installed security system that allows users to lock/unlock doors and arm/disarm the immobilizer.

Pairing/Unpairing: KARR's default protocol does not pair devices with modules; there is no way to add a trusted device to a module. Instead of pairing over BLE, they control access to modules via a UI filter in their smartphone application. Once users and dealers log in to the application, the UI only

Product	Application-Layer BLE Security Features			Remote Control Feature
	Pairing	Authentication	Encryption	
KARR	Default: *None* Optional: BLE pairing/bonding	Challenge-response w/hard-coded keys	*None*	Door locks, Horn, Immobilizer
Rockledge	Per-user ID	Check if user ID matches	AES/CBC with partially hard-coded key	Door locks, Immobilizer
CarLink	Per-user ID added when engine started while doors unlocked	Check if user ID matches	AES with partially hard-coded key	Door locks, Engine starter
DS4	Per-user BLE pairing + Public key added by in-cabin button press	BLE bonding + Asym. challenge-response	BLE encryption & AES with DH key	Door locks, Engine starter
EvoStart	Per-vehicle key added in factory and accessed via cloud login	Sym. challenge-response	AES with session key encrypted with per-vehicle key	Door locks, Engine starter
Linkr	Per-vehicle key added in factory and accessed via cloud login	Sym. challenge-response	Keeloq	Door locks, Engine starter
OEM Tesla	Per-user key transferred after infotainment trigger w/NFC confirm	Asym. challenge-response	AES/GCM with ECDH key [48]	Door locks, Engine starter

Table 2: BLE application-layer security of aftermarket remote control systems. Highlighting indicates vulnerabilities.

displays modules they are registered for. KARR dealers register modules in the cloud when they install them, and they transfer control of a module if the user pays for it.

KARR's application also provides the option to enable BLE's built-in pairing and bonding with the module, but this feature appears only in the settings menu.

Authentication: KARR implements a custom application-layer challenge-response protocol over BLE. The module sends an authentication request packet containing a random byte array, and the application must respond with the correct hash value computed from the following: the challenge bytes, the module name and mode (both contained in the module's advertisements), and either of two *hard-coded fixed keys*. Fixed keys are shared across all KARR modules for both dealer-level access (i.e., operation and configuration) and user-level access (i.e., operation only). An attacker with these keys can implement a malicious application that can authenticate with every KARR module.

Encryption: KARR does not implement encryption.

4.3 Rockledge

Rockledge is another dealership-installed anti-theft and keyless entry system similar to KARR. The BLE version of the Rockledge alarm entered the market close to the time of publication, so we were unable to obtain the actual hardware. The following analysis is based solely on our reverse-engineering of their application.

Pairing/Unpairing: We were unable to fully investigate Rockledge's pairing mechanism without access to the module's firmware. However, a unique 4-byte user ID for each logged-in application user appears to be stored in the module during the installation process.

Authentication: Every Rockledge command is individually authenticated by including the user ID in command messages, combined with a 4-byte stock ID. The user ID is a secret, it is likely loaded into the module during installation (and later synced to the application via an authenticated cloud API). The 4-byte user ID is too big to to brute force over BLE. However, the stock ID can simply be calculated by XORing the module name (encoded as hex string) with the manufacturer data in its BLE advertisements.

Encryption: Rockledge encrypts messages with AES-128, using an encryption key derived entirely from the plaintext portion of each message. The key is derived by concatenating a 4-byte random nonce included at the beginning of each message with a 6-byte session key, both obtainable from the module's advertisements, and a *6-byte hard-coded key* in the application. This provides obfuscation but not meaningful encryption. The attacker can sniff the nonce and session key, and combine them with the hard-coded key to decrypt traffic to obtain the user ID, thereby obtaining all information needed to authenticate with the module.

4.4 CarLink

CarLink adds BLE and Long Range (LoRa) connectivity to a wide variety of aftermarket remote engine starters, almost all of which control both the ignition and door locks [44]. Both CarLink and Rockledge appear to be created by the same developer and are built on the same module hardware, so they share some similarities in their application-layer protocols.

Pairing/Unpairing: CarLink's module enters pairing mode whenever the vehicle starts its engine while the doors are unlocked. An attacker can simply wait for an owner to unlock their doors and start their car to pair their malicious device with the CarLink module. Once in pairing mode, any nearby

smartphone with the official app can automatically pair with a CarLink module by transmitting a BLE message containing a unique 3-byte user ID. Up to 10 IDs can be stored in the module at a time, with a FIFO policy. The application provides a way for a user to remove a CarLink module from their account; however, our testing revealed the operation only removed the car from the user's app interface, and not the ID stored on the module. The user can even add the same vehicle again while within the Bluetooth range, without reactivating the pairing mode. We are not aware of any mechanism for the module owner to delete a pairing from the module, aside from pairing enough new user IDs to kick out all existing pairings.

Authentication: Similar to Rockledge, the module only authenticates incoming commands by verifying that the 3-byte user ID in the message has been previously paired with it.

Encryption: The CarLink module encrypts messages with AES-128 encryption. However, like Rockledge, the encryption key is derived by concatenating the following cleartext information: a 4-byte random nonce, the module's MAC address, and a predefined key suffix hardcoded in the application. The application reads the random nonce from a BLE characteristic in cleartext before connecting. Therefore, an attacker can reconstruct the AES key by first sniffing the random nonce sent when a user connects their device to the module. This will also capture the device's MAC address, as it is contained in every BLE packet. With the AES key, the attacker can decrypt all command messages, each of which contains a paired user ID needed for authentication.

4.5 DS4

DS4 is a popular retail aftermarket product that adds BLE remote control capability to a remote engine starter system

Pairing/Unpairing: DS4 is the only aftermarket remote control system we studied that simultaneously uses a custom application-layer protocol and BLE's built-in pairing and bonding mechanisms. Also, they provide a secure way to initiate pairing; the user must start the engine and hold a button on the module inside the vehicle for 10 seconds. The module then accepts pairing requests from the application. DS4 is the most difficult pairing process for an attacker to initiate out of all the products we have studied. Once BLE pairing is complete, the application and the DS4 module proceed to perform application-layer pairing. During this phase, the application and the module exchange public keys securely over the bonded BLE connection. DS4 uses a custom application-layer Diffie-Hellman key-exchange protocol to generate a shared secret key for subsequent authentication and encryption. Multiple phones can be added to a single DS4 module's trusted list. The primary owner of a DS4 system can remove any trusted phone from the module's trusted list via the mobile app over BLE. This removes both the Long-Term Key (LTK) used in BLE bonding and the application-layer session key.

Authentication: There are two layers of challenge-response authentication in DS4: (1) BLE's built-in challenge response with the Long-Term Key (LTK) obtained from the BLE bonding process, and (2) the custom application-layer Diffie-Hellman keys exchanged during application-layer pairing. DS4's custom Diffie-Hellman key exchange is cryptographically weak as it uses a 128-bit prime modulus, rather than the NIST-recommended length of 2048 bits [5].

Encryption: DS4's message payload is encrypted twice: first at the application layer using AES-128 with the shared secret key from its custom Diffie-Hellman key exchange protocol, and then encrypted again (i.e., double encryption) at the underlying BLE link, which has its own key and encryption. These two layers of encryption make DS4 one of the most difficult aftermarket systems for an attacker to decrypt messages sniffed in transit.

4.6 EvoStart

EvoStart is a BLE remote control from Fortin that pairs with an EVO-ONE aftermarket remote starter. The EvoStart application-layer protocol is part of a third-party BLE library called `hyll`[3], which is used to operate a range of automotive modules (e.g., OBD2 scanners [17]). One component of this library is used to remotely control EvoStart modules.

Pairing/Unpairing: EvoStart modules have a unique secret key preprogrammed. The application obtains this key (i.e., pairs) with the modules by obtaining it from the cloud. When the user first purchases a model, they first log in and register the module under their account. Then they can access a cloud API that returns the module's secret key. Pairing in EvoStart does not involve direct BLE communication with the module, nor any physical interaction (e.g., pressing a button).

The primary owner can use the cloud to share access to the module with other registered users, but doing so only gives them access to module's unique key. If a primary user revokes access for a shared user, the server will refuse to provide the module key, and the app will delete any cached copies of it. However, because the module's key is fixed for each module, a shared user can retain access to the vehicle indefinitely by extracting it from the mobile application[4].

Authentication: EvoStart authenticates clients using a custom challenge-response protocol based on the per-module unique key. When the application connects to the module it sends a randomly generated session key encrypted using the per-module key. The user's commands will only succeed if they are encrypted using a correctly-derived session key.

[3] `hyll`'s code references `http://www.topmem.net/Index.html`.

[4] We were able to discover that there is a per-module key by injecting logs into the APK file and printing out the secret key.

Encryption: All communication is encrypted using AES-128 with this session key. Attackers will therefore not gain any information needed for unauthorized access attacks by intercepting messages, because decryption requires the per-module key to derive the session key.

4.7 Linkr

Linkr provides similar BLE control over remote starter systems as CarLink and EvoStart.

Pairing/Unpairing: Similar to EvoStart's pairing mechanism, each Linkr module has a unique encryption key provided to the application through an authenticated cloud API. However, instead of using an AES-128 like the other systems, Linkr uses a 64-bit Keeloq [25] key. Keeloq is a popular cypher used in automotive keyless entry systems.

Authentication: Linkr authenticates with a bi-directional challenge-response using the module's Keeloq key.

Encryption: Linkr messages are encrypted with the unique Keeloq module key. Prior cryptographic attacks on KeeLoq [13, 15, 20] have been demonstrated. However, they are infeasible in our threat model, because they require observing thousands of encrypted messages. Our threat model assumes the attacker has access to the system for only a short period of time, and since messages are only transmitted when the user interacts with the application, an attacker is unlikely to be able to capture thousands of messages.

5 End-to-End Attacks

In the previous section, we demonstrated that there are critical vulnerabilities in the KARR and CarLink aftermarket remote control systems. However, an attacker still needs to overcome several practical challenges before mounting an attack. First, the attacker must find vehicles with vulnerable modules installed. Then, the module's firmware must allow an attacker to quickly gain control of the module or, if necessary, reconfigure it to do so. We demonstrate, through end-to-end tests of the actual remote control modules, that it is practical for an attacker to gain control of vulnerable vehicles from while standing near the vehicle and attacking from a nearby vehicle.

5.1 Vehicle Targeting

We show how an attacker can target vulnerable vehicles in two practical ways: (1) finding vehicles from public crowdsourced databases or (2) by visually identifying stickers or hardware indicating the presence of a specific remote control system.

5.1.1 Wardriving databases

An attacker can locate targets using the WiGLE public crowdsourced BLE wardriving database by querying for vulnerable devices by their BLE name prefix. They can then determine likely target locations by observing devices at the same location multiple times. This is feasible because WiGLE records locations over time, revealing frequently visited locations (e.g., parking lots, residential areas), where the vehicle is likely to be parked.

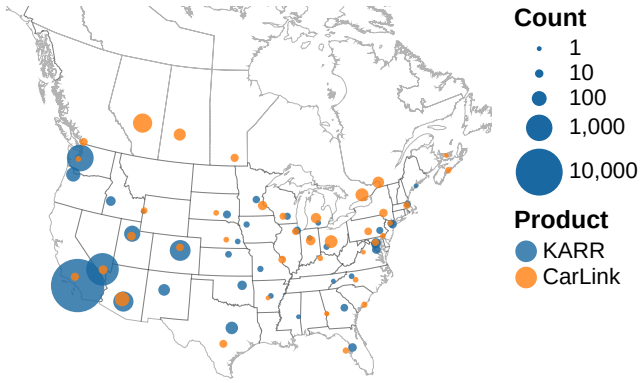


Figure 3: Geographic distribution of targetable car systems seen in public crowdsourced wireless scanning datasets. Each circle represents the total number of WiGLE entries with the highest QoS within each state/province, with the center of the circle being the center of the geographic distribution.

Finding where cars are parked: WiGLE provides an easy way to tell if a device has been observed at the same location several times: WiGLE's Quality of Service (QoS) metric. The QoS values increment by one for each additional day, or user, that observes the device at a location. The maximum QoS value of 7 indicates the highest likelihood that a device was observed multiple times at a specific location.

Evaluating the feasibility of targeting: To evaluate the feasibility of this targeted attack, we sampled 50 of the most recently observed CarLink[5] modules from WiGLE and downloaded all associated geospatial data points. We define a module as targetable if it has been observed at least three times within a 100m radius. This filters out observations of moving vehicles on roadways and retains only those that regularly appear at specific locations. We found that 31 out of the 50 devices (62%) could be targeted.

Geographic distribution of targetable vehicles: We evaluate the geographic distribution of vehicles that can be targeted from wardriving databases. Figure 3 shows the distribution of all KARR and CarLink vehicles obtained from WiGLE with $QoS = 7$: CarLink vehicles have deployments in areas with extreme climates, consistent with CarLink's remote start capability to pre-warm/cool the car. Tens to hundreds were observed in population centers across Canada and the U.S. KARR vehicles have an extremely dense deployment in the south-

[5]Targeting CarLink systems is particularly difficult because there are very few deployed.



Figure 4: An attacker can visually identify a target vehicle equipped with KARR by looking for a window sticker.

western United States. We expect this is the region where many dealerships pre-install KARR devices.

5.1.2 Visual indicators

Attackers can also target vulnerable vehicles by visually looking for indicators that a vehicle has an aftermarket remote-control system. KARR security systems come with a sticker on the driver's side window, affixed by dealers during installation (Figure 4). The CarLink system can also be visually targeted because the installation manual recommends affixing the module to the windscreen to provide line-of-sight to its internal antenna for long-range communication.

5.2 Testing the KARR Attack

The application-layer BLE vulnerability discussed in Section 4.2 gives an attacker dealer-level access to remotely control the KARR module (i.e., Door Locks, Horn, Lights). However, the module's firmware and hardware may impose limitations on the attacker or even create new vulnerabilities. We tested the full end-to-end attack on a real KARR module and found that the attacker's capability is affected by the following aspects of the KARR module: (1) operating modes, (2) BLE radio range, and (3) OTA firmware update capability.

5.2.1 Operating modes

The KARR module has four operating modes: *Installer*, *Dealer*, *User*, and *No Sale*. We describe how each of those modes work and how they create and prevent attacks.

Mode lifecycle (Figure 5): KARR modules come from the factory in *Installer* mode. Installers can individually test each component that the alarm can control, such as lights and the horn. The installer then switches the module to *Dealer* mode, allowing the dealer to control the locks and immobilizer while the vehicle is on the lot for sale (i.e., for test rides). When a buyer pays for access to KARR, the dealer switches the

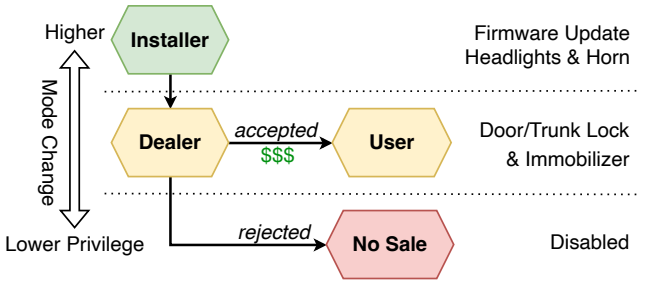


Figure 5: KARR remote control module's modes. The privileged commands for each level of mode are on the right.

module to *User* mode, which gives them the same features as the *Dealer*, except they cannot change the mode. The buyer is given an account on the KARR application. If the buyer does not opt to pay for access to the KARR module, the dealer switches it to *No Sale* mode, which deactivates all its features, while still allowing the dealer to reactivate over BLE. However, the module only allows reactivation when the ignition is on.

Attacks created by modes: In *No Sale* mode, an attacker can switch the victim's module to *User* mode and enable the immobilizer, preventing the victim from starting their car again. Since the victim did not purchase access to the KARR module, they will not have access to the application to disable the immobilizer, rendering the car disabled until it is brought to a service center to change modes or remove the KARR module.

Attacks prevented by modes: In *No Sale* mode, when the car is parked (i.e., ignition off), the module disables BLE. This thwarts an attacker from gaining access to parked *No Sale* vehicles. Instead, they have to wait for the victim to start the car before changing to a higher-privileged mode and then perform any of the previously described attacks.

Mode distribution: Next we analyze the distribution of operating modes across KARR modules using our wardriving dataset (Section 3.1). To avoid underestimating the percentage of *No Sale* vehicles, we restrict our analysis to running vehicles (i.e., ignition on); recall that KARR modules in *No Sale* mode turn off BLE when the ignition is off. We identify running vehicles using the battery voltage field in KARR's BLE advertisements we collected. A vehicle's alternator will raise the battery voltage above 13 Volts when the vehicle is running [11]. We found that 47% of all running KARR vehicles operate in *No Sale* mode, while 42% operate in *User* mode. These results indicate that nearly half of KARR-equipped vehicle owners may not be unaware that they have a BLE remote control system active in their car.

5.2.2 BLE radio range

We tested the BLE range of an attack on two different vehicles equipped with KARR alarms—a Mazda 3 initially in *No Sale* mode and a Toyota Camry initially in *User* mode.

Setup: We implemented an attacker application that unlocks the car using the KARR application-layer BLE protocol (including the hard-coded secret keys). The application first authenticates with the privileged dealer key, then performs a mode change to *Installer* mode, and finally issues the unlock command. Since the Mazda 3 was in *No Sale* mode, we had to perform the attack while the engine was running, but since the Camry was in *User* mode we attacked it while it was off.

For each vehicle, we attempted to perform the attack ten times at various distances and angles, then recorded the success rate and time taken. We also tested the attack both standing outside and from inside a nearby car, as if the attacker was driving on the road near it. We tested the BLE range of the Mazda 3 in a busy outdoor parking lot, and the Toyota Camry in a smaller outdoor lot with adjacent walls.

Observations: Table 3 shows the observed range for attacks that can be performed reliably in under 5 seconds. Attacks can be performed reliably from at least 3 meters away. Operating inside a nearby car reduces the range by half compared to standing outside of the car. The angle of the attacker significantly affects the range. Specifically, for the Camry, operating from the side of the car doubled the attack range.

5.2.3 OTA firmware update

The KARR module does not do firmware signature verification. An attacker can rewrite the firmware to completely take complete control over it. We tested the time needed to upload new firmware over BLE. It takes ∼60 seconds, making it difficult for an attacker to perform on a moving vehicle.

5.3 Testing the CarLink Attack

We tested an end-to-end attack on CarLink and found that it can be performed using the official smartphone application, and that the BLE radio range is much further than KARR. We also found built-in defenses in remote starters that may make it difficult to steal a CarLink even after remotely starting it.

5.3.1 Attacking with the official application

An attacker needs to construct a QR code to pair with a CarLink module using the official application. However, the QR code is just the module's MAC address contained in the BLE advertisement that is always sent. After scanning this QR code, a dialog box appears prompting the user to "turn on ignition" to enter the module into pairing mode. Once paired, the remote engine starter accepts unlock and ignition commands issued by the maliciously paired user.

System	Outside Standing	Inside Vehicle
KARR	5–20m	3–10m
CarLink	<150m	<60m

Table 3: BLE attack range depending on attack angle.

5.3.2 BLE radio range

To measure the attack range, we mounted a CarLink module in the Mazda 3, and tested the attack from standing outside and from inside a nearby vehicle. In both cases, the BLE range was approximately 6× longer than KARR (Table 3). This could be due to a combination of higher BLE transmit power in CarLink and a less concealed placement of the module in the vehicle compared to KARR.

5.3.3 Feasibility of car theft

The CarLink attack enables unlocking and starting a vehicle, but other protections may make it difficult to steal the car. Many remote start systems include a security mechanism that kills the ignition if the OEM keys are not in the car when the brake pedal is pressed. This prevents the attacker from holding the brake pedal to shift into drive while the car is remote started. Also, some vehicles have a steering wheel lock that also engages when the OEM keys are not in the car.

6 Scale of Vulnerable Aftermarket Systems

To understand the scale of this problem, we estimated the number of vulnerable vehicles with KARR and CarLink systems. To the best of our knowledge, there is no existing way of estimating the real-world deployment scale of a BLE device. We found that we can estimate deployment scale by analyzing MAC address patterns from a sample of BLE wardriving datasets. To evaluate accuracy of our method, we compare the estimated deployment scale of Rivian systems against their publicly announced sales figures.

6.1 Methodology

To estimate the total number of devices for a product, we make two assumptions: (1) manufacturers allocate MAC addresses sequentially within a small number of address blocks, and (2) wardriving datasets contain a random sample over the entire population of deployed devices.

KARR, CarLink, and Rivian all appear to allocate MAC addresses sequentially within a small number of address blocks. We collected all 725,808 unique KARR devices, 8,825 CarLink devices, and 92,443 Rivian sensors from WiGLE using name pattern searches. Figure 6 visualizes the MAC address allocation patterns we observed for these three products. The addresses are clustered in groups across a small set of 24-bit

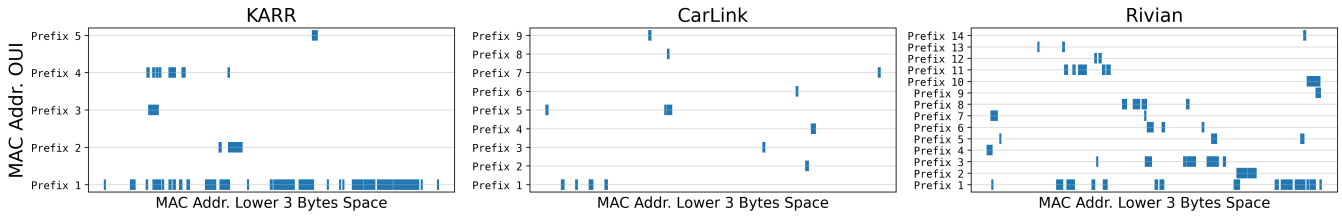


Figure 6: MAC address allocation patterns of the vulnerable remote control modules, and Rivian for comparison.

prefixes (OUIs). Within each cluster, the addresses are mostly sequential with small gaps between allocations. Each system also has a few MAC addresses whose OUI prefixes do not appear in the sequential allocation blocks, possibly because manufacturers use at least one BLE radio vendor that allocates random MAC addresses to its products. We only found 221 such addresses for KARR, 164 for CarLink, and 402 for Rivian in our WiGLE samples. Since these random addresses are a small fraction of the total sample size, we expect they do not have a significant effect on our estimation and so are filtered out in Figure 6.

Testing for sequential addresses: We verified that addresses are allocated sequentially by analyzing the statistical distribution of the intervals between observed addresses. If there was random sampling of a sequentially allocated population, the intervals between observed addresses should follow a *geometric distribution*, with the frequency of observed intervals decaying exponentially as the interval size increases. We tested for this property by performing a linear regression on the log-transformed frequency of each interval size. We use the coefficient of determination (R^2) as a goodness-of-fit measure to assess whether the observed MAC address intervals exhibit exponential decay. The R^2 value of this regression indicates how closely the data fits a geometric distribution. We define a passing threshold of $R^2 > 0.9$ if the data fits a geometric distribution. The results of this analysis are shown in Table 5. All three systems with sequential allocation patterns sequential allocation patterns pass this threshold, while DS4 and Tesla do not. DS4 has $R^2 \approx 0$ (309,675 observed addresses spread across 298,492 OUI blocks), indicating they are likely randomly assigned. Tesla's R^2 is only 0.26; although, its 9,951 observed addresses fall within 99 OUI blocks, most of which belong to Texas Instruments. Therefore, it is likely that Tesla does not randomize addresses, but our samples represent too small of a fraction of its millions of deployed vehicles to observe the geometric distribution.

6.1.1 Estimating Scale

Ideally, we can simply take the highest and lowest observed MAC addresses in each block to calculate the range of allocated addresses. However, as shown in Figure 6, these manufacturers allocate MAC addresses in fragmented, scattered batches separated by large, unassigned gaps within their OUI blocks. Using a range calculation approach will count these empty gaps as real devices, leading to overestimation.

Strategy	Samples	Estimation	R^2	Error
Full Dataset	92,443	150,867	0.981	2.9%
50% Random	46,221	148,989	0.997	1.6%
10% Random	9,244	157,681	0.902	7.5%
Los Angeles	3,562	230,573	0.222	57.2%
San Diego	1,590	114,816	0.119	21.7%

Table 4: Estimation robustness to excluding portions of the Rivian dataset.

To sidestep this problem, we instead estimate the sampling probability of our wardriving dataset. The sampling probability is the probability that a BLE device in the entire population is observed by a wardriver. Once we know the sampling probability, we can scale up the number of observed samples to estimate the total population size using the following formula:

$$\text{Population} = \text{Number of samples} \times \frac{1}{\text{Sampling probability}}$$

Estimating the sampling probability: We measured the probability by observing the frequency of consecutive MAC addresses in our wardriving sample (e.g., `00:11:22:33:44:02` and `00:11:22:33:44:03`). Intuitively, more consecutive addresses observed means a higher sampling probability of the space of allocated MAC addresses. For instance, if we had a 100% sampling probability, we would observe nearly 100% of the addresses to be consecutive (aside from the few gaps between allocations); conversely, if our sample probability is low we would rarely see consecutive addresses. Thus, the frequency of observing consecutive addresses in our sample gives us a measure of our sampling probability. Since we assume our sample represents a random sample of the population, we can use this sampling probability to scale up our sample size to estimate the total population. A high-level formula for our estimation is as follows:

$$\text{Sampling probability} \approx \frac{\text{Number of MAC intervals of size 1}}{\text{Number of MAC intervals}}$$

We provide a derivation of this formula in Appendix C.

We evaluated the robustness of our method by observing its sensitivity to excluding portions of the Rivian dataset and

Product	Sample Size	R^2	Estimation
KARR	725,808	0.92	1,473,136
CarLink	8,825	0.97	31,927
Rivian	92,443	0.98	150,867

Table 5: Estimates of number of vulnerable deployed devices, obtained using our wardriving-based MAC address scale estimation approach. Millions of devices will need a manual firmware update due the vulnerabilities identified in this work.

comparing our estimates to Rivian's sales numbers. Table 4 shows the results of various ways of excluding samples. Random downsampling to 50% and 10% of the original data yielded estimates with only 1.6% and 7.5% error respectively, demonstrating that our method remains accurate with smaller sample sizes. However, geographic sampling to specific cities, such as Los Angeles, increased error rates (e.g., 57.2%). This suggests that data may need to be collected across the entire deployment area, possibly due to clustering of specific MAC address ranges sold into certain areas. The R^2 values for the random downsampling experiments are high (0.997 and 0.902), supporting our assumption that MAC addresses are sequentially allocated. In contrast, the regional samples show poor R^2 values (0.222 and 0.119), indicating violations of our sequential allocation assumption. Therefore, we may be able to use the R^2 value to test if a dataset does not contain sequentially allocated MAC addresses.

6.2 Results

With our method, we estimate the populations of KARR and CarLink. We estimate that over 1.4 million vulnerable KARR devices, and around 31,927 vulnerable CarLink devices have been deployed (Table 5). Most of these devices now require users to manually update their firmware to patch the vulnerabilities identified in this work. The problem is that all CarLink modules, and most KARR modules, only have BLE connectivity; they do not have a cellular interface for distributing Over-the-Air updates. Vendors are moving more toward integrating cellular connectivity. For instance, KARR's newer `DR` module (Table 1) uses both cellular and BLE connectivity and therefore it may be able to be updated remotely.

6.3 Global Spread

Since these vulnerable systems are widely deployed in North America, we asked the question: are vulnerable vehicles also being resold or relocated overseas? To evaluate the global distribution of these systems we searched on WiGLE for KARR and CarLink modules observed outside of North America using their BLE advertisement name prefixes and lengths.

CarLink systems have been observed in 27 different countries, with the largest density of 37 seen in the UK and 35 in Japan, and the remaining countries have only 1–6 modules observed. The global spread of KARR is summarized in Table 6. The most dense deployment is in South America. Surprisingly, hundreds of KARR vehicles were also observed in Asia and Oceania. Note that these numbers represent the number of devices observed by WiGLE wardrivers, rather than estimates of the true global deployment scale. Based on our previous estimations though in Table 5, the population estimates can be two to four times the sample size.

Region	Observed Devices
Central and South America	2,263
Europe	600
Asia	949
Oceania	625

Table 6: Global geographic distribution of observed KARR devices captured in the WiGLE database.

These findings show that vulnerabilities in aftermarket remote control systems can spread across the global automotive supply chain.

7 Mitigations and Countermeasures

We now describe the mitigations a vendor can implement when application-layer authentication and encryption vulnerabilities exist in widely deployed BLE remote control systems. Specifically, we describe the mitigations KARR's vendor implemented, and the mitigations we proposed to CarLink and Rockledge. We also propose countermeasures for future iterations of aftermarket BLE remote control systems.

7.1 KARR mitigations

When we disclosed to KARR's vendor, we proposed several mitigations. We now discuss the mitigations they implemented and their priorities. This provides a unique perspective on the challenges vendors face when patching a large deployment of a BLE product with hardcoded authentication credentials.

7.1.1 Short-term Mitigations

KARR immediately implemented several short-term mitigations to reduce the likelihood of attacks. These changes could be implemented quickly because they involved modifying and removing applications distributed through centralized app stores.

Removing Shared Keys from Apps: They updated all applications to fetch BLE authentication keys from an authenticated web API rather than hardcoding them.

Purging App Stores: They also made it more difficult for attackers to find the hardcoded keys by requesting the removal of historical versions of their applications from online repositories [1, 2]. In addition, the vendor removed the "Installer" application from app stores because it disclosed hidden BLE commands that were not available to users.

7.1.2 Long-term Mitigations

The vendor implemented several long-term mitigations that significantly improved authentication and pairing security in their existing module deployment. These changes took about a year to implement and deploy because they required significant firmware updates and the development of a patching approach for owners that do not know they are installed in their vehicles.

Adding Per-Device Keys: KARR restructured its BLE authentication process to use a random, unique per-module key. Even if an attacker were to obtain the key, the impact is limited to a single vehicle.

Disabling BLE Advertising in No Sale Mode: KARR's developers decided to disable BLE entirely for modules in *No Sale* mode. This prevents attackers from performing BLE attacks on vehicles even though the modules remain installed.

Securely Pushing Updated Firmware: Both of these mitigations require updating firmware on all deployed modules. Updating the firmware requires using the previously hardcoded secret key, which can only be accessed through an authenticated web API. For modules in *User* and *Dealer* mode, this key can be fetched because users have KARR web credentials. Since these users are authenticated, they can get the previously hardcoded secret key through the web API. However, *No Sale* module owners do not have credentials because they did not purchase access to the module. Instead, *No Sale* owners use their Vehicle Identification Number (VIN) as a credential to access the hardcoded key. This is possible because, fortunately, KARR maintains a database of VINs of vehicles that have modules pre-installed in them. This solution strikes a balance between leaving devices unpatched and the risk of exposing the hardcoded key.

7.2 CarLink mitigations

The primary mitigation we proposed to CarLink's vendor was to require an interaction with the module while inside the vehicle to pair a new smartphone to the remote control module. For instance, requiring the user to press a button on the module inside the car to start pairing, instead of entering pairing every time the car starts. They acknowledged that this mitigation would prevent unauthorized pairing.

7.3 Rockledge mitigations

Rockledge's vulnerabilities in authentication and encryption can be mitigated by implementing per-device authentication keys and per-session encryption keys.

7.4 Countermeasures for Future Products

Future application-layer BLE protocols should incorporate the following defenses against the attacks described in this work. These are based on our observations of security mechanisms that were effective in the products we analyzed (Section 4), combined with our experiences developing mitigations for existing vulnerabilities.

Physical interaction to enter pairing mode: Modules should require a physical interaction to initiate pairing to grant access to a new smartphone. A physical button is sufficient for systems that only control the cars' locks. Remote starter systems should also include a numeric keypad or an NFC keycard to enter pairing mode, preventing an attacker from breaking into the car and pairing with the remote starter to bypass the built-in immobilizer and start the car.

Per-session encryption keys: BLE remote control systems need to securely exchange session keys. Developers need to implement their own Diffie-Hellman Key Exchange.

Mutual Authentication: The application and module should also mutually authenticate to each other during session establishment using per-user and per-module asymmetric keys. This will prevent attacker-in-the-middle and relay attacks.

OTA updates over cellular: Future remote control systems implemented with application-layer BLE protocols should also include cellular capabilities. If a vulnerability is found, this would allow security updates to be pushed to devices in a timely manner (similar to OEM systems), mitigating the risk of a forever-day attack, particularly for dealer-installed systems. Also, cellular connectivity would allow unpairing BLE smartphone apps remotely to prevent unauthorized control if a phone is lost or stolen, or if paired keys are compromised.

Token-based access control with a trusted third party: In a single-owner scenario, a per-device authentication key is sufficient to prevent system-wide compromise of the authentication key. However, in some products we studied, such as CarLink, ownership can be temporarily transferred to multiple users (e.g., lending a car to a friend). For temporary access, the module should require the application to present a time-limited access token issued and signed by a trusted third party (e.g., the vendor's cloud server). The access token may also contain fine-grained access control policies, such as expiration time and allowed features. This makes it possible to revoke access, either immediately over BLE or remotely over Cellular by refusing to issue new tokens. Vendors may also adopt existing industry-standard implementations of such

a system, such as ACE-OAuth [35], specifically designed for IoT devices.

8 Related Work

This paper builds on a significant body of prior work on wireless device discovery, analyzing application-layer BLE security, and finding vulnerabilities in vehicle access systems and insecure BLE devices.

Discovery: Prior work has utilized stationary sensor networks for traffic analysis on roadways by tracking previously known in-vehicle wireless devices [29,37,46]. In our work, we focus on identifying previously unknown automotive remote control systems using mobile BLE scanners. Researchers have also shown the benefit of using public crowdsourced wardriving databases [4] for locating wireless access points [14,34]. However, these scans are collected from crowdsourced mobile devices that scan infrequently. So it is difficult to identify the locations that a moving vehicle frequently visits. Bhaskar et al. [12] demonstrated that it is possible to precisely locate BLE devices in gas pumps, but this required frequent scanning, and supplementing wardriving with finer-precision warwalking data. In our work, we collect our own frequent scans of BLE devices. We combine this with OpenStreetMap-based geobounding of freeways to find popular remote-control systems. We also show that public wardriving databases sometimes can reveal specific locations where owners park their vehicles.

Application analysis: Researchers have discovered vulnerabilities in the web APIs of cellular-based vehicle security systems [28], enabling attackers worldwide to remotely control vehicles. Our work focuses on weaknesses in systems controlled locally via BLE. Prior work also analyzed a wide variety of vulnerabilities in companion applications for BLE IoT devices. Researchers have fingerprinted BLE devices using UUIDs [38,39,50,51], identified pairing and authentication vulnerabilities [39,50,51], and found threats from colocated BLE applications running on the same device [38].

Attacks on vehicle access and communication systems: Researchers have found a variety of vulnerabilities in vehicle access systems, particularly keyless entry and immobilizer systems, which are typically exploited through passive eavesdropping and replay attacks [18,30,47,48]. Others have also looked at breaking the cryptography of well-established keyless entry systems, such as Keeloq [20,25] and Hitag2 [19,41]. Several attacks have also been found by sniffing messages and injecting spoofed messages through different entry points such as the OBD2 port [17,40], ECUs [22,27], and infotainment systems [26]. Our work is the first to discover there are also significant vulnerabilities in widely deployed aftermarket remote control systems.

Insecure BLE in IoT: A variety of vulnerabilities have been identified in BLE-controlled IoT devices. Researchers found vulnerabilities in the BLE security model [24,33,45]. In particular, Just Works pairing introduces many problems, shifting the security burden to the application developers and device manufacturers. We observe a similar problem in BLE remote control systems, where developers chose the lowest BLE security mode (no encryption, no authentication) and handrolled their own pairing, authentication, and encryption mechanisms, leading to the vulnerabilities that we cover in this work.

9 Conclusion

The failures we observed in the aftermarket BLE remote control ecosystem serve as a cautionary tale about the pitfalls of using BLE to establish a direct local link between an offline device and a smartphone. Automotive remote control systems can not use BLE's built-in security modes because multiple smartphones that may not be locally paired need to be able to control the vehicle. Developers end up disabling BLE's built-in security by downgrading it to the lowest security mode, and "rolling their own" application-layer security to provide these features. Unfortunately, they inadvertently end up misusing security primitives. Additionally, vulnerabilities in application-layer BLE protocols could leave millions of modules vulnerable with no way to automatically update them all at once. Instead, every user needs to manually update their own module, if they even know that they have it. This essentially creates permanent N-days because manual patching will never be 100%. These vulnerable cars poison the supply chain and can end up being deployed all over the world. In wardriving data, we found vulnerable cars from the United States operating in Central and South America, Asia, Europe, and Oceania.

10 Acknowledgements

We would like to thank our shepherd and the anonymous reviewers from USENIX Security 2026 for their insightful comments and suggestions. We also thank the reviewers from USENIX Security 2025 whose feedback led to this manuscript. This work was supported in part by National Science Foundation grant no. CNS-2239163.

A Ethics Discussion

This section provides a detailed discussion of the Ethics of this work, including: (1) how this work may affect each of the stakeholders in the aftermarket remote control ecosystem, and (2) how we followed a principled ethical approach to using BLE wardriving datasets for this work.

A.1 Stakeholders

Vehicle owners using vulnerable aftermarket remote control products: During the course of our research, we did not experiment on any vehicle without the explicit consent of the owner. However, due to publication of the vulnerabilities we found, vehicle owners with affected products can potentially be exposed to increased risk of property theft/damage in the short term. However, our disclosures to vendors have led them to developing patches. We are also doing everything we can to ensure the patches are made available before the paper's publication (i.e., by embargoing the paper).

Drivers who are unaware of vulnerable products in their vehicles: Drivers without knowledge that they have a vulnerable BLE remote control system installed in their vehicle—particularly systems installed by dealerships—may face prolonged risk as vendors cannot instantly patch these devices without the driver's involvement. However, these hidden vulnerabilities can eventually be addressed by a combination of increased public awareness through the publication of our work and vendors or government agencies directly contacting vehicle owners.

Aftermarket product vendors and installers face potential loss of revenue/income from the publication of these vulnerabilities. However, we have begun disclosure processes well in advance of submission, and we have worked closely with these vendors to improve the security of their products. Car dealerships using and/or profiting from these products Dealerships who rely on these vulnerable products as a secondary revenue source also face potential loss of revenue. Publication of this work could lead to public distrust of these installations. However, we believe that the benefit of raising awareness of these vulnerabilities outweighs these downsides, which we try to mitigate with our disclosure process. Also, if dealers help their customers patch the vulnerable systems that could potentially help them regain trust in the product.

Society at large benefits from this work as it brings to light significant security concerns that raises awareness of this overlooked segment of the automotive remote control market. In the long run, this may contribute to increased consumer awareness and a more secure aftermarket ecosystem.

Summary: We acknowledge that our work carries certain risks and the impact of our revelation can be far-reaching. However, we have taken a principled approach to our research work to ensure that no individual's rights were violated, and every possible action is taken to mitigate potential harm to the aforementioned stakeholders.

A.2 BLE Dataset Analysis

We use both our own BLE advertisement collection and the WiGLE public crowdsourced BLE dataset to identify and analyze the scale and spread of aftermarket BLE systems.

Our Bluetooth measurements of KARR and CarLink deployment do not involve any human subjects. The data is collected passively, by capturing Bluetooth beacons from the remote control systems. These data are collected do not involve human subjects. These beacons only contain the serial number of the device, the voltage of the car battery, and for some models, the mode the device is in.

WiGLE is a public data set, and we carefully analyze and report on the data from WiGLE based on the principles outlined in the Menlo Report:

Beneficence: WiGLE enabled us to discover the geographic spread of vulnerable BLE remote control systems. By identifying these regions we determined what regions of the world car owners need to be particularly cautious of if their new or used vehicle has a vulnerable BLE remote control system. Also, we only present aggregated data in our paper and ensure no raw data about individual remote control systems is included in the paper's artifacts.

Respect for Persons: Although BLE beacons are broadcast publicly, we were careful to ensure that our analysis and artifacts do not expose any details about individual vulnerable systems (e.g., their geolocation). This work may entice attackers to use WiGLE to find vulnerable targets, but they would then be violating the WiGLE Terms of Service.

Justice: We identified specific regions that could be targeted in our study. However, we report on all risk-prone regions equally, even covering the spread of car security systems from North America around the rest of the world. We also do not target a specific aftermarket vendor with this study.

Respect for Law and Public Interest: The WiGLE data enabled us to determine that the scale and spread of the systems was so significant, and included many cars where the owner may not know they even have a vulnerable aftermarket remote system, that we needed to make this information public so everyone can patch their systems. We also worked closely with WiGLE administrators to ensure that our research data collection is compliant with their terms of service.

Tangible Harms: WiGLE data is primarily used in an aggregate way for large-scale data analytics at the resolution of entire cities, states, and counties. An individual whose data is used in our study is exposed to no additional harm than what is discussed in the subsequent sections. There may however be harms if an attacker were to use WiGLE to find where vulnerable vehicles are to attack.

Violations of Human Rights: We believe our data collection and analytics processes do not constitute any human rights violations.

A.2.1 Decision

Taking into account all of the potential benefits and harms of the work, we decided to proceed with publication due to a

combination of the following three issues: (1) “Respect to Persons” the attacks we discovered are trivial and could already be in use today, (2) "Justice" many users do not currently know they are vulnerable to these attacks because a dealer installed an aftermarket BLE remote control system in their vehicle, and users have to manually update their BLE remote control systems to defend against an attack, (3) "Beneficence" vendors are more likely to fix the problem quickly and robustly because of the attention that this work will provide to these vulnerabilities.

B Open Science

We provide as many code artifacts as possible to reproduce our measurement results. This includes all the scripts we used to analyze the data and generate the figures and tables in this work. We will provide restricted access to the data used to generate the plots, especially for evaluating the feasibility of targeting vulnerable systems. The concern with releasing all our data is that it would be unethical to widely make available a list of targets before the vendors, whom we disclosed to, have had time to deploy patches. We can make that data available however on request if researchers demonstrate a good faith effort that they will not use it for attacks.

We also include a code artifact that demonstrates the vulnerability on the KARR remote control system. The secret key is scrubbed from our artifact, and we worked with the vendor to also scrub off the key from as many APK mirror sites as possible. This is necessary to prevent easy use of this key to do the attack, especially since millions of vehicles now have to be manually patched to protect against it.

The artifact repository is available at https://zenodo.org/records/20656773.

C Scale estimation formula

In a sequentially allocated address space, the intervals between consecutive sorted addresses follows geometric distribution. Let X be a random variable representing the interval size between two consecutive sorted addresses in a uniform random sample of the address space, the probability mass function is

$$P(X = k) = (1-p)^{k-1}p$$

for $k = 1,2,3,\ldots$, where p is the success probability of a trial. In our case, p represents any given address being included in our sample. With n samples in a total population of size N, we have

$$p = \frac{n}{N}$$

The key insight is that the probability of observing an interval of size 1 between consecutive sorted addresses is equal to the sampling probability p.

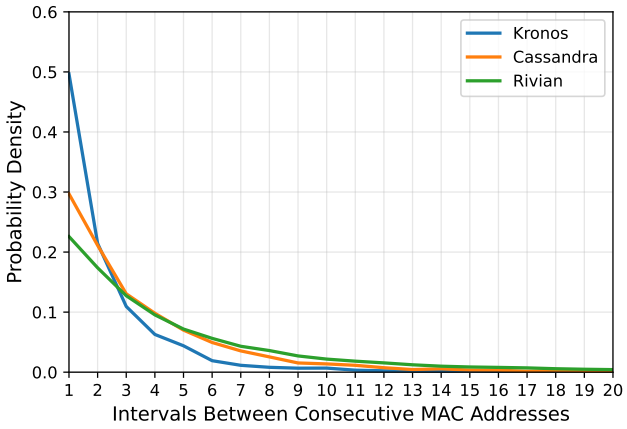


Figure 7: Interval probability density

$$P(X = 1) = (1-p)^{1-1}p = p$$

We know $P(X = 1)$ by measuring the frequency of observing an interval $X = 1$ in our samples.

$$P(X = 1) \approx \frac{\text{count}(\text{interval} = 1)}{n-1}$$

Where $n-1$ is the total number of intervals between n samples. Rearranging the equation, we can estimate the total population N:

$$N = \frac{n}{p} = \frac{n}{P(X=1)} \approx \frac{n}{\left(\frac{\text{count}(\text{interval}=1)}{n-1}\right)}$$

If manufacturers allocate addresses sequentially, the interval distribution should exhibit geometric characteristics: decreasing frequencies as interval size increases, with small intervals being more common than large intervals. From figure 7, we observe clear geometrical distributions in the probability density functions of these products.

For KARR and CarLink, the estimated populations are as follows:

$$N_{KARR} \approx \frac{725,808}{\left(\frac{357,602}{725,808-1}\right)} \approx 1,473,136$$

$$N_{CarLink} \approx \frac{8,825}{\left(\frac{2,439}{8,825-1}\right)} \approx 31,927$$

For Rivian, our dataset represents 69% of the total WiGLE data, but we cannot treat this as a random sample because WiGLE provides data sorted by MAC address rather than randomly sampled. Additionally, each Rivian vehicle contains 4 BLE sensors. Therefore, in our estimation, we account for these factors by dividing by our download ratio 0.69 and the number of sensors per vehicle 4. We observed 20,593 intervals

of size 1 out of 92,443 samples. We estimate the total number of Rivian vehicles to be as follows:

$$\begin{aligned} N_{rivian} &\approx \frac{92,443}{\left(\frac{20,593}{92,443-1}\right)} \times \frac{1}{0.69} \times \frac{1}{4} \\ &\approx 414,977 \times 1.45 \times 0.25 \\ &\approx 150,867 \end{aligned}$$

References

[1] APKMirror. https://www.apkmirror.com/.

[2] APKPure. https://apkpure.com/.

[3] Car Connectivity Consortium Members. https://carconnectivity.org/our-members/.

[4] WiGLE: Wireless Network Mapping. https://www.wigle.net/.

[5] Recommendation for Pair-Wise Key-Establishment Schemes Using Discrete Logarithm Cryptography. Nist special publication, National Institute of Standards and Technology, 2018.

[6] CCC Digital Key: The Future of Vehicle Access. Technical report, Car Connectivity Consortium, 2022.

[7] Bluetooth Low Energy CCC Digital Key R3 Application Note. Technical report, NXP Semiconductors, 2024.

[8] ILSpy: Open-source .NET assembly browser and decompiler. https://github.com/icsharpcode/ILSpy, 2024.

[9] Open Street Maps, Motorway map. https://wiki.openstreetmap.org/wiki/Tag:highway%3Dmotorway, 2025.

[10] D. Antonioli, N. O. Tippenhauer, and K. Rasmussen. Key Negotiation Downgrade Attacks on Bluetooth and Bluetooth Low Energy. *ACM Transactions on Privacy and Security*, July 2020.

[11] Battery University. BU-303: Confusion with Voltages. https://batteryuniversity.com/article/bu-303-confusion-with-voltages, 2021.

[12] N. Bhaskar, M. Bland, K. Levchenko, and A. Schulman. Please Pay Inside: Evaluating Bluetooth-based Detection of Gas Pump Skimmers. In *Proceedings of the 28th USENIX Security Symposium, SEC'19*, pages 373–388. USENIX Association, August 2019.

[13] N. T. Courtois, G. V. Bard, and D. Wagner. Algebraic and Slide Attacks on KeeLoq. In *Proceedings of the 15th International Workshop on Fast Software Encryption, FSE 2008*, pages 97–115. Springer-Verlag, February 2008.

[14] A. Di Luzio, A. Mei, and J. Stefa. Mind Your Probes: De-Anonymization of Large Crowds Through Smartphone WiFi Probe Requests. In *Proceedings of the 35th Annual IEEE International Conference on Computer Communications, INFOCOM'16*, pages 1–9. IEEE, April 2016.

[15] T. Eisenbarth, T. Kasper, A. Moradi, C. Paar, M. Salmasizadeh, and M. T. Shalmani. On the Power of Power Analysis in the Real World: A Complete Break of the KeeLoq Code Hopping Scheme. In *Proceedings of the 28th Annual Conference on Cryptology: Advances in Cryptology, CRYPTO 2008*, pages 203–220. Springer-Verlag, August 2008.

[16] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, pages 226 —- 231. AAAI Press, 1996.

[17] I. Foster, A. Prudhomme, K. Koscher, and S. Savage. Fast and Vulnerable: A Story of Telematic Failures. In *Proceedings of the 9th USENIX Workshop on Offensive Technologies, WOOT'15*, pages 15–23. USENIX Association, August 2015.

[18] A. Francillon, B. Danev, and S. Capkun. Relay Attacks on Passive Keyless Entry and Start Systems in Modern Cars. In *Proceedings of the Network and Distributed System Security Symposium, NDSS'11*, pages 332–346. The Internet Society, February 2011.

[19] F. D. Garcia, D. Oswald, T. Kasper, and P. Pavlidès. Lock It and Still Lose It – On the (In)Security of Automotive Remote Keyless Entry Systems. In *Proceedings of the 25th USENIX Security Symposium, SEC'16*, pages 929–944. USENIX Association, August 2016.

[20] S. Indesteege, N. Keller, O. Dunkelman, E. Biham, and B. Preneel. A Practical Attack on KeeLoq. In *Proceedings of the 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques, EUROCRYPT'08*, pages 1–18. Springer-Verlag, April 2008.

[21] M. K. Jangid, Y. Zhang, and Z. Lin. Extrapolating Formal Analysis to Uncover Attacks in Bluetooth Passkey Entry Pairing. In *Proceedings of 30th Annual Network and Distributed System Security Symposium, NDSS'23*. The Internet Society, February 2023.

[22] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, and S. Savage. Experimental Security Analysis of a Modern Automobile. In *Proceedings of the IEEE Symposium on Security and Privacy, SP'10*, SP '10, pages 447–462. IEEE Computer Society, May 2010.

[23] G. Kwon, J. Kim, J. Noh, and S. Cho. Bluetooth Low Energy Security Vulnerability and Improvement Method. In *Proceedings of the International Conference on Consumer Electronics-Asia, ICCE-Asia'16*, October 2016.

[24] K. Lounis and M. Zulkernine. Bluetooth Low Energy Makes "Just Works" Not Work. In *Proceedings of the 3rd Cyber Security in Networking Conference, CSNet'19*, pages 99–106. IEEE, October 2019.

[25] K. Marneweck. An Introduction to KEELOQ® Code Hopping. Technical report, Microchip Technology Inc, 1996.

[26] S. Mazloom, M. Rezaeirad, A. Hunter, and D. McCoy. A Security Analysis of an In Vehicle Infotainment and App Platform. In *Proceedings of the 10th USENIX Conference on Offensive Technologies, WOOT'16*, page 232–243. USENIX Association, August 2016.

[27] A. Mohammed, Y. Man, R. Gerdes, M. Li, and Z. B. Celik. Relay Attacks on Passive Keyless Entry and Start Systems in Modern Cars. In *Proceedings of the 4th International Workshop on Automotive and Autonomous Vehicle Security, AutoSec'22*. The Internet Society, February 2022.

[28] K. Munro. Gone in Six Seconds: Exploiting Car Alarms. https://www.pentestpartners.com/security-blog/gone-in-six-seconds-exploiting-car-alarms/, March 2019.

[29] A. B. M. Musa and J. Eriksson. Tracking Unmodified Smartphones Using Wi-Fi Monitors. In *Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems, SenSys'12*, pages 281–294. Association for Computing Machinery, November 2012.

[30] NCC Group. Technical Advisory — Tesla BLE Phone-as-a-Key Passive Entry Vulnerable to Relay Attacks, May 2022.

[31] Reddit. Tesla NFC Key Card Security Vulnerability Demonstration (RKiE). https://www.reddit.com/r/teslamotors/comments/v86pc1/tesla_nfc_key_card_security_vulnerability/, 2022.

[32] Reddit. I just bought a new Jetta. Does anyone know what this is? https://web.archive.org/web/20250804184434/https://www.reddit.com/r/AskMechanics/comments/1ijd8pc/i_just_bought_a_new_jetta_does_anyone_know_what/, 2025.

[33] M. Y. Sadaoui, O. Salem, and A. Mehaoua. Security Assessment of Bluetooth Just Works Pairing Method: Vulnerabilities and Enhancements. In *Proceedings of 22nd IEEE International Conference on E-health Networking, Application & Services, Healthcom'23*, pages 107–112. IEEE, December 2023.

[34] P. Sapiezynski, R. Gatej, A. Mislove, and S. Lehmann. Opportunities and Challenges in Crowdsourced Wardriving. In *Proceedings of the 15th ACM Internet Measurement Conference*, pages 267–273, New York, NY, USA, October 2015. Association for Computing Machinery.

[35] L. Seitz, G. Selander, E. Wahlstroem, S. Erdtman, and H. Tschofenig. Authentication and Authorization for Constrained Environments Using the OAuth 2.0 Framework (ACE-OAuth). RFC 9200, Aug. 2022.

[36] M. Shi, J. Chen, K. He, H. Zhao, M. Jia, and R. Du. Formal Analysis and Patching of BLE-SC Pairing. In *Proceedings of the 32nd USENIX Conference on Security Symposium, SEC'23*. USENIX Association, August 2023.

[37] J. P. Singer, A. E. Robinson, J. Krueger, J. E. Atkinson, and M. Myers. Travel Time on Arterials and Rural Highways: State-of-the-Practice Synthesis on Arterial Data Collection Technology. Technical report, United States Federal Highway Administration, 2013.

[38] P. Sivakumaran and J. Blasco. A Study of the Feasibility of Co-located App Attacks against BLE and a Large-Scale Analysis of the Current Application-Layer Security Landscape. In *Proceedings of the 28th USENIX Conference on Security Symposium, SEC'19*, pages 1–18. USENIX Association, August 2019.

[39] P. Sivakumaran, C. Zuo, Z. Lin, and J. Blasco. Uncovering Vulnerabilities of Bluetooth Low Energy IoT from Companion Mobile Apps with Ble-Guuide. In *Proceedings of the 18th ACM ASIA Conference on Computer and Communications Security, ASIACCS'23*, pages 1004–1015. Association for Computing Machinery, July 2023.

[40] C. Valasek and C. Miller. Adventures in Automotive Networks and Control Units, 2014.

[41] R. Verdult, F. D. Garcia, and J. Balasch. Gone in 360 seconds: Hijacking with Hitag2. In *Proceedings of the 21st USENIX Conference on Security Symposium, SEC'12*, pages 37–52. USENIX Association, 2012.

[42] VERSPRITE. How hackers control & steal vehicles remotely. https://versprite.com/vs-labs/hacking-remote-start-system/.

[43] M. von Tschirschnitz, L. Peuckert, F. Franzen, and J. Grossklags. Method Confusion Attack on Bluetooth Pairing. In *Proceedings of 42nd IEEE Symposium on Security and Privacy, SP'21*, pages 1332–1347, May 2021.

[44] VOXX International. Complete Installation & Programming Guide: Telematics Communicator System ASCL6. https://web.archive.org/web/20250319100348/https://static.voxxintl.com/docs/common/ASCL6/ASCL6_IM.pdf.

[45] J. Wang, F. Hu, Y. Zhou, Y. Liu, H. Zhang, and Z. Liu. BlueDoor: Breaking the Secure Information Flow via BLE Vulnerability. In *Proceedings of the 18th International Conference on Mobile Systems, Applications, and Services, MobiSys'20*, MobiSys '20, pages 286–298. Association for Computing Machinery, June 2020.

[46] Y. Wang, Y. Malinovskiy, Y. Wu, and U. K. Lee. Error Modeling and Analysis for Travel Time Data Obtained from Bluetooth MAC Address Matching. Technical report, Washington State Department of Transportation, 2011.

[47] L. Wouters, E. Marin, T. Ashur, B. Gierlichs, and B. Preneel. Fast, Furious and Insecure: Passive Keyless Entry and Start Systems in Modern Supercars. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, May 2019.

[48] X. Xie, K. Jiang, R. Dai, J. Lu, L. Wang, Q. Li, and J. Yu. Access Your Tesla without Your Awareness: Compromising Keyless Entry System of Model 3. In *Proceedings of 30th Annual Network and Distributed System Security Symposium, NDSS'23*. The Internet Society, February 2023.

[49] Y. Zhang, J. Weng, R. Dey, Y. Jin, Z. Lin, and X. Fu. Breaking Secure Pairing of Bluetooth Low Energy Using Downgrade Attacks. In *Proceedings of the 29th USENIX Conference on Security Symposium, SEC'20*. USENIX Association, August 2020.

[50] Q. Zhao, C. Zuo, J. Blasco, and Z. Lin. PeriScope: Comprehensive Vulnerability Analysis of Mobile App-defined Bluetooth Peripherals. In *Proceedings of the 17th ACM ASIA Conference on Computer and Communications Security, ASIACCS'22*, pages 521–533, May 2022.

[51] C. Zuo, H. Wen, Z. Lin, and Y. Zhang. Automatic Fingerprinting of Vulnerable BLE IoT Devices with Static UUIDs from Mobile Apps. In *Proceedings of the 26th ACM SIGSAC Conference on Computer and Communications Security, CCS'19*, pages 1469–1483. Association for Computing Machinery, November 2019.