Implementing and testing RollJam on Software-Defined Radios

Dario Stabili[1], Filip Valgimigli[2], Tobia Bocchi[2], Filippo Veronesi[1] and Mirco Marchetti[2]

[1]Alma Mater Studiorum - University of Bologna, Department of Computer Science and Engineering, 40126 Bologna, Italy
[2]University of Modena and Reggio Emilia, Department of Engineering "Enzo Ferrari", 41125 Modena, Italy

Abstract

In this paper, we present a comprehensive implementation of the RollJam attack using Software-Defined Radios (SDR), offering a detailed exploration of the practical aspects and implications of this wireless security vulnerability. The RollJam attack, initially introduced by Samy Kamkar in 2015, exploits weaknesses in rolling code-based keyless entry systems, allowing unauthorized access to vehicles and other secure environments.

Our research focuses on the development and deployment of a RollJam device leveraging SDR technology, enabling a cost-effective and versatile implementation for security researchers and practitioners. We discuss the intricacies of the attack methodology, including the jamming of radio frequency signals during key fob transmissions, recording and storing valid codes, and executing replay attacks to gain unauthorized access.

To provide a realistic evaluation of the RollJam attack's viability, we conduct experiments on a range of devices equipped with rolling code-based systems, and we analyze the effectiveness of the attack on various implementations and variations of keyless entry systems.

1. Introduction

The pervasive adoption of keyless entry systems in modern vehicles and access control mechanisms has undeniably enhanced user convenience, yet it has also ushered in new challenges in the realm of security. One of the main weakness of these systems is the susceptibility to hacking and electronic manipulation. As these systems rely on wireless signals and digital communication, they become targets for skilled hackers who may exploit weaknesses in encryption algorithm or intercept and duplicate access credentials. Among the array of vulnerabilities afflicting these systems, the RollJam attack [1] has emerged as a pivotal threat, leveraging weaknesses in rolling code technology to compromise the security of key fobs and access control devices. In particular, the RollJam attack is a wireless hacking technique that targets keyless entry systems based on rolling code technology. In a RollJam attack, a device intercepts and blocks the initial signals sent by a user attempting to unlock a car or open a secure entry point. While the user believes they have successfully accessed the system, the intercepted code is

ITASEC 2024: The Italian Conference on CyberSecurity, April 09–11, 2024, Salerno, Italy
dario.stabili@unibo.it (D. Stabili); filip.valgimigli@unimore.it (F. Valgimigli); tobia.bocchi@unimore.it (T. Bocchi); filippo.veronesi16@unibo.it (F. Veronesi); mirco.marchetti@unimore.it (M. Marchetti)
0000-0001-6850-334X (D. Stabili); 0009-0009-2852-2461 (F. Valgimigli); 0000-0002-7408-6906 (M. Marchetti)

not transmitted immediately. The attacker can then replay the intercepted code later to gain unauthorized access, exploiting the time gap between the intercepted and transmitted signals to compromise the security of the keyless entry system.

In this paper we present a comprehensive exploration of the RollJam attack, its practical implications, and the vulnerabilities it exploits in rolling code-based keyless entry systems. RollJam, first unveiled by security researcher Samy Kamkar in 2015, exploits the transmission and reception processes in these systems, allowing attackers to intercept and later replay valid codes for unauthorized access.

Our work builds upon the foundations laid by Kamkar's initial research, seeking to bridge the gap between theoretical understanding and practical implementation. We investigate the nuances of deploying RollJam devices, emphasizing the utilization of Software-Defined Radios (SDR) to execute and analyze the attack in real-world scenarios. Through a series of experiments and case studies, we evaluate the effectiveness of RollJam across diverse rolling code implementations over different frequencies, shedding light on the varying degrees of susceptibility exhibited by different systems. As an additional contribution, our implementation of the RollJam attack on SDR is publicly released [2] (upon request), enabling security researchers to develop novel solutions that can be easily tested against this type of attack.

The remainder of the paper is organized as follows. Section 2 discusses all related works in the field of Sub-GHz attacks, while Section 3 presents all the basic concepts required for understanding this work. Section 4 describes the RollJam attack in detail, showcasing its applicability on our range of test devices. Section 5 discusses our implementation of RollJam with support to SDRs, while the demonstration of the effectiveness of the attack against different control access systems over different radio signals is presented in Section 6. Finally, conclusions and future development are outlined in Section 7.

2. Related Work

Although rolling code schemes have been designed to enhance the security of RF-based communication systems, they have consistently demonstrated vulnerabilities since their first proposal. For example, Classic KeeLoq technology, predominantly utilized for garage doors, was compromised through cryptoanalysis [3, 4] and side-channel attacks targeting the key derivation scheme employed by the receiver [5]. Notably, a significant contribution to this field of study is presented in [6], where researchers identified several vulnerabilities in the keyless entry systems of most VW Group vehicles manufactured over a period exceeding two decades. Furthermore, they exposed vulnerabilities in rolling-code schemes different from KeeLoq, enabling attackers to derive session keys used in communication, thereby facilitating unauthorized cloning of key fobs. Despite enhancements made to the KeeLoq scheme since its first version, including the use of longer keys and stronger encryption algorithms, all iterations of KeeLoq and other RF-based secure communication schemes remain susceptible to low-level attacks such as jamming. In this type of attack, an attacker disrupts communication by transmitting a high-power signal on the same RF frequency, preventing the vehicle from receiving any legitimate signals.

A more sophisticated version of the basic jamming attack is the *selective jamming and replay* attack. In this scenario, the attacker not only jams the vehicle receiver antenna but also records

the signal transmitted by the key fob, effectively capturing a legitimate lock/unlock signal for future replay. An illustration of this attack is provided in [1], where the author demonstrates a brute-force method for compromising fixed-code RF communication and introduces the RollJam attack, which combines jamming with radio signal recording to disrupt communication between a car and its associated fob. Initially, the RollJam demonstration included a board equipped with antennas to support the attack, but currently, devices supporting RollJam are freely available on the market [7]. However, while RollJam-supporting boards are accessible, they often come with limited configuration options or require substantial modifications to replicate the attack entirely. Despite RollJam has been revised by security researchers from its first demonstration [8], existing implementations are often missing some key functionalities that hinders their application in a real scenario, thus limiting their direct applicability and requiring a huge effort to make them applicable.

Specifically, the implementation outlined in [9] solely comprises the record and replay phases of the attack, each as distinct projects. Consequently, integrating the jamming phase into the attack is necessary to align it with a RollJam attack scenario. In contrast, our implementation encompasses all essential stages for executing a RollJam attack within a unified workflow. This comprehensive approach enables researchers and security practitioners to directly utilize our implementation for evaluating a system's susceptibility to the RollJam attack. Other publicly accessible repositories primarily focus on discussing the attack intricately [10], demonstrating implementations tailored for individual boards or RF modules [11, 12], or supporting a fixed array of antennas [13, 14, 15].

Another detailed and publicly available implementation of the RollJam attack can be found in [16]. However, we have identified some critical issues that require significant modification before it can be deployed for testing purposes. One particular issue that we address in our implementation pertains to the nature of the jamming signal. In [16], the jamming signal is generated from a fixed sequence of bits ($[1, 0, 0, 1, 0]$) rather than utilizing a random sequence. Consequently, the implementation available in [16] can be easily circumvented by applying simple interference mitigation techniques once the jamming signal is known [17]. Another critical issue found in [16] is the requirement of manually activating the different phases of the attack, thus limiting the automatic setup of a test environment. In the implementation presented in this paper, the three phases of the RollJam attack are included in a single workflow, allowing the attacker to jam the signal and record and replay the rolling codes programmatically. Finally, in the recording phase of the implementation presented in [16], there is no demodulation of the signal, resulting in the preservation of both the signal and all background noise. In our implementation, however, since we include demodulation of the signal during the recording of the rolling code, we only save the actual data transmitted by the key fob, discarding any unnecessary background noise. We also note that by saving the demodulated signal and using modulation in the replay phase, we can send a clearer signal to the immobilizer, thus increasing the efficiency of the attack itself.

Compared to the current state-of-the-art, the implementation presented in this paper offers security researchers and practitioners a straightforward tool for testing any system against the RollJam attack in an automated fashion. Moreover, developing the RollJam attack to support multiple SDRs allows researchers to test their system in different conditions without requiring a dedicated hardware platform.

3. Background Knowledge

3.1. Radio-Frequency signals and modulation

An radio-frequency (RF) signal is a signal that is coherently generated, radiated by a transmit antenna, propagated through air or space, collected by a receive antenna, and then amplified and information extracted [18]. One of the key characteristics distinguishing RF signals from infrared and visible light is that an RF signal can be generated with coherent phase, and information can be transmitted in both amplitude and phase variations of the RF signal. Such signals can be easily generated up to 220 GHz. However, since the hardware necessary to generate higher frequency radio signals is more expensive than the one required to generate low-frequency radio signals, it is common to find various communication protocols based on Sub-GHz RF. Radars, telephony, FM radio, TV broadcast and RFID are some of the most common communication systems based on Sub-GHz RF. However, to carry information within the RF it is necessary to modify the carrier sinusoid of the baseband frequency by varying the amplitude, phase, and/or frequency of the sinusoid. This action is called *modulation*. Historically, modulations can be distinguished in two families: *analog* modulation, where the carrier signal is modified (in either amplitude, phase, or frequency) to carry the modulating signal; and *digital* modulation, where the carrier signal is created considering the data to transfer, resulting in a more sharp difference between the data and the carrier signal. Creating a digitally-modulated signal is based on both transmitter and receiver being synchronized in reading the state of the waveform of the signal, with the time between the two readings defining the bandwidth of the signal. According to the characteristic of the carrier signal being modulated, we can distinguish between three types of digital RF signals.

3.1.1. Frequency Shift Keying

Frequency Shift Keying (FSK) is the simplest form of digital modulation, where digital data is transmitted by varying the frequency of the carrier signal. In FSK, binary data is represented by two distinct frequencies, typically denoted as the 0 and 1 states. When transmitting a binary 0, the carrier signal is modulated to one specific frequency, and for a binary 1, it shifts to another predetermined frequency. The change in frequency corresponds to the different states of the binary data, enabling efficient and reliable transmission.

3.1.2. Phase Shift Keying

Phase Shift Keying (PSK) is a digital modulation scheme that transmit data by varying the phase of the carrier signal. In PSK, different phase states represent distinct symbols or bits of information. Commonly, Binary Phase Shift Keying (BPSK) uses two phase states, 0 and 180 degrees, corresponding to binary values 0 and 1, respectively. More advanced variants, such as Quadrature Phase Shift Keying (QPSK) and Quadrature Amplitude Modulation (QAM), employ multiple phase states to transmit multiple bits per symbol, thereby increasing data transmission efficiency.

3.1.3. Amplitude Shift Keying

Amplitude Shift Keying (ASK) is is a digital modulation scheme that transmit data by varying the amplitude of a carrier signal. In ASK, different amplitude levels represent distinct binary states. Typically, two amplitude levels are used, with one level representing binary 0 and another representing binary 1. The amplitude of the carrier signal is modulated based on the binary data to be transmitted, resulting in a signal that switches between the predetermined amplitude levels.

3.2. Keyless Entry Systems Security

As automotive technology advances, traditional mechanical car keys have evolved into sophisticated electronic components, known as key fobs. These fobs serve as radio transmitters, allowing tasks like unlocking doors and starting the engine with a single button press. They are closely tied to immobilizers, security devices in vehicles that prevent unauthorized access. Immobilizers often use RFID technology, embedding a transponder within the key fob's shell. When the key is near the vehicle or the *open* button is pressed, a confidential code is transmitted to the in-vehicle immobilizer, unlocking the doors and starting the engine upon detecting the keys inside. To maintain code confidentiality, modern systems employ cryptography, often utilizing challenge-response protocols in systems like Passive Key Entry and Start (PKES). PKES systems use a bidirectional challenge-response scheme within a small operating range, typically about one meter. When the key is near the vehicle, it responds to the challenge received by the vehicle, unlocking it if the response is correct. However, PKES systems, lacking the need for user interaction like a button press on the key fob, are vulnerable to relay attacks, where signals are relayed wirelessly, allowing authentication without direct physical interaction between the vehicle and key.

An alternative to Keyless Entry systems is Remote Keyless Entry (RKE), which relies on one-way data transmission from the key fob to the vehicle. Users initiate RF transmission by pressing a button, sending a code to unlock the vehicle's doors. This signal operates on widely available radio frequencies like 433 MHz or 868 MHz in Europe, or 315 MHz in North America, covering ranges of tens to hundreds of meters. RKE allows users to remotely control locking, unlocking, and additional features like the anti-theft alarm or trunk opening. Originally using RF signals with a "fixed code", modern RKE systems employ "rolling code" strategies for increased security. These systems incorporate cryptography and a counter value that increments with each button press. The counter value, along with other parameters, is encoded in a plaintext message to generate the rolling code signal. Upon receiving this code, the immobilizer compares the counter value with its internal value. If the counter is correct, the signal is considered fresh and valid; otherwise, it is rejected. This mechanism effectively prevents replay attacks, as a valid signal cannot be replayed.

One standard implementation for rolling codes in automotive security is KeeLoq [19], a proprietary algorithm designed for generating rolling codes on key fobs. Despite KeeLoq being compromised by cryptoanalysis [3, 20] and side-channel attacks on the key derivation scheme [5, 4], it's important to note that these attacks require a deep understanding of automotive systems and make strong assumptions about system availability. Although seemingly unrealistic at the

time, they remain a significant future threat and should be continuously monitored.

However, KeeLoq and other rolling code schemes used in RKEs systems are still vulnerable to attacks like the *RollJam* attack, which doesn't require access to the vehicle's system and can be deployed from outside the vehicle to completely disrupt RF communication between the key fob and the immobilizer.

4. The RollJam attack

The RollJam [1] attack is a wireless attack that targets keyless entry systems, specifically those using rolling code technology. The attack takes advantage of vulnerabilities in the implementation of rolling code systems, which are commonly used in key fobs for vehicles, garage door openers, and other wireless access control systems. A basic (yet similar to the RollJam) attack to RKEs is based on a simple RF jamming between the key fob of the car key and the immobilizer, preventing any signal from being received. RollJam, on the other hand, is based on a combined *eavesdrop-and-jam* approach, where the attacker also monitors the rolling code signal (carrying either the *lock* or the *unlock* action to the immobilizer) on the target RF while jamming communication. This enables the attacker to store a valid code that can be used later to either lock the vehicle after a burglary or unlock it. The only limitation of this attack is that, after the eavesdropped signal is replayed, it is not possible to create other valid messages without eavesdrop them, thus limiting the time window for the attack. The full RollJam attack comprises the following phases:

- **Jamming phase:** The attacker uses a device to jam the radio frequency signal between the key fob and the target device (e.g., vehicle). When the user presses the button on the key fob to unlock the car or open a gate, the jamming signal interferes with the transmission, preventing the target device from receiving the code.
- **Record phase:** While jamming the signal, the attacker records the transmitted code. The target device, unaware of the interference, does not receive and process the first code, thus leaving it in a vulnerable state. The attacker now has a stored valid code that the target device did not receive due to the jamming.
- **Replay phase:** At a later time, the attacker can replay the valid code mimicking the behavior of the legitimate key fob. The target device, still unaware of the interference during the initial transmission, accepts the replayed code and grants unauthorized access to the attacker.

The RollJam attack highlights a weakness in some rolling code implementations, where the target device does not properly handle situations where the transmitted code is jammed. This vulnerability allows an attacker to replay a previously intercepted code to gain unauthorized access to the target device.

5. RollJam Implementation

In the implementation of the RollJam attack described in this section, we utilized a Universal Software Radio Peripheral (USRP) $B200$ model from Ettus Research as our primary SDR

device [21]. The USRP $B200$ offers a versatile range of specifications, operating within a frequency range spanning from 70 MHz to 6 GHz. It supports full-duplex communication, thus allowing simultaneous transmit and receive operations. However, it's worth noting that our implementation also supports SDRs without full-duplex capabilities, providing flexibility for researchers to utilize alternative SDR devices. We implemented the RollJam attack using the GNU Radio toolkit [22]. GNU Radio is an open-source software development toolkit that offers signal processing blocks for building radio systems. It facilitates the design, simulation, and deployment of radio communication systems using SDR platforms. GNU Radio provides a graphical user interface, known as GNU Radio Companion, for designing signal flow graphs. Users can connect various signal processing blocks to create custom radio systems, performing functions such as modulation, demodulation, filtering, encoding, decoding, and more. GNU Radio supports a wide range of SDR devices, making it adaptable to various radio communication applications. It is extensively used in academic research, hobbyist projects, and industrial applications for prototyping, testing, and implementing radio communication systems. While all phases of the RollJam attack are included in the final implementation within the same project, we will present and discuss each phase of the attack individually.

5.1. Jamming phase

In the Jamming phase of the RollJam attack we transmit a strong and narrow-band noise signal to disrupt the normal RF communication between the key fob and the vehicle immobilizer. The objective of the attack is to hinder the immobilizer on the vehicle from receiving the signal while enabling the attacker to record it for later use. Hence, it's crucial to set the center frequency of the jamming signal close to the operating frequency of the key fob, facilitating the attacker's ability to filter out the jamming signal and extract the desired RF signal. In our implementation, the noise gain is set at approximately 50% of the maximum gain (approximately 40 dB on the SDR driver) to prevent harmonics, thereby allowing for straightforward signal extraction through filtering during the recording phase.

5.2. Record phase

In the Record phase of the RollJam attack, we capture the signal transmitted from the key fob, which we previously prevented from reaching the immobilizer. This process involves demodulating the received signal at the precise operating frequency of the key fob to extract the data containing the user's command. In our implementation, we also opt to demodulate the received signal based on the modulation scheme employed by the key fob (ASK or FSK). This approach allows us to store only the encoded binary data instead of the entire signal. All operations in the Record phase occur while the jamming signal is still being transmitted, necessitating the use of either a full-duplex or two half-duplex devices.

5.3. Replay phase

In the Replay phase of the RollJam attack, we replay the previously recorded signal over the same frequency to replicate the behavior of the legitimate key fob. Since our implementation relies on the demodulation of the received signal, in this phase we modulate the bitstream

(using the same modulation) to generate a new signal. All the operations in the Replay phase are performed without the jamming signal.

6. RollJam demonstration

In this section, we will present three different types of tests on our implementation of the RollJam attack. Specifically, we will assess the range and effectiveness of the jamming attack and analyze the actual memory footprint of saving the demodulated signal compared to saving the entire signal in both ASK and FSK modulation. It is worth noting that since PSK modulation is typically used for WLANs, RFID, and Bluetooth communication, testing the RollJam against this type of modulated signal falls outside the scope of this work.

All tests are conducted on a laptop equipped with an Intel $i7 - 1165G7$ processor, 16 GB of RAM running Arch Linux and GNU Radio 3.10.9.2. Our SDR setup includes an Ettus USRP B200 board [21] paired with an Ettus Vert 2450 antenna [23].

6.1. Jamming range

To assess the effectiveness of the jamming signal designed in our implementation, we tested its operational range in two different scenarios. The first scenario represents an attack where the jamming signal is placed in close proximity to the target vehicle (e.g., inside a vehicle parked near the target), while the second scenario represents an attack where the driver is close to the vehicle while the jamming signal is not fixed in position. In both test scenarios, we deployed a jamming signal with the same characteristics discussed in Section 5.1.

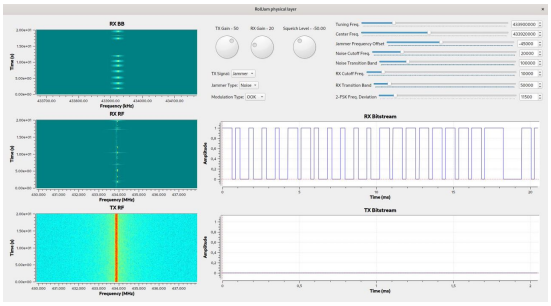
The results of these tests highlight that placing the jamming signal 2 meters from the target vehicle can prevent the vehicle from receiving signals transmitted by the key fob. Conversely, if the key fob is in close proximity to the vehicle (approximately 1 meter), the jamming signal is only effective up to 5 meters from the vehicle.

6.2. ASK record and replay

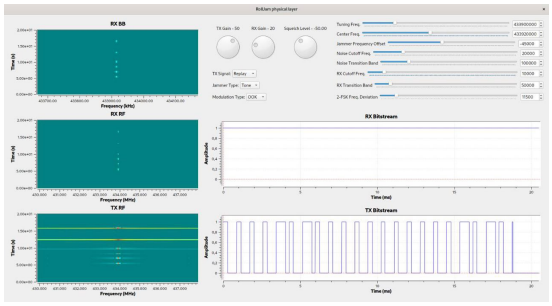
The results of the test on the record and replay phases of the ASK signal are depicted in Figures 1a and 1b, respectively. Both figures present the same plots to aid in identifying the different steps of the attack and verifying its effectiveness. In the left vertical section of Figures 1a and 1b, there are three waterfall plots. From top to bottom, the plots labeled *RX BB* and *RX RF* showcase the incoming signal received by the SDR (baseband and radio frequency, respectively), while the *TX RF* plot showcases the signal emitted from the SDR. It should be noted that the baseband representation of the received signal can be extracted from the RF signal by downconverting the received signal to the baseband and filtering the final output to extract the transmitted data.

In the other vertical section of both Figures, a series of configuration parameters used in our implementation and both received and transmitted bit streams are depicted from top to bottom.

Focusing on the demodulation and record phase of our RollJam implementation (Figure 1a), it is evident that we can accurately demodulate the incoming signal (RX BB, RX RF, and RX Bitstream plots) while the jamming signal is actively transmitted by the SDR (TX RF). In the



(a) ASK signal demodulation and record.



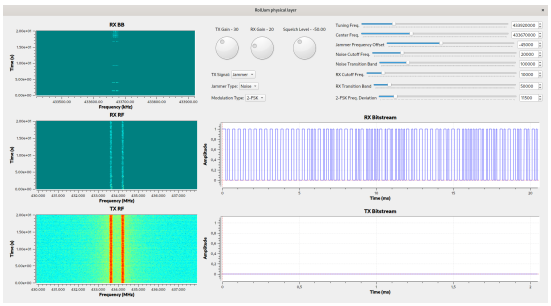
(b) ASK signal modulation and replay.

modulation and replay phase, it is clear that the bitstream corresponding to the demodulated signal is being transmitted from the SDR (TX Bitstream and TX RF plots).

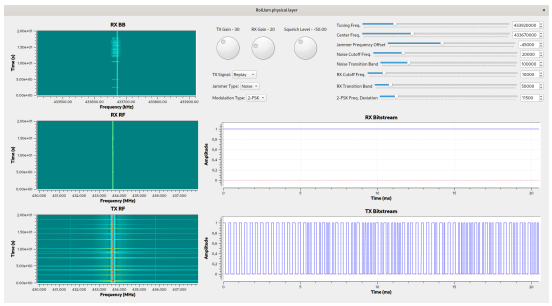
The overall size of the recorded ASK signal (without demodulation) is 30 MB (3818880 samples at a sampling rate of 1 Mbps), while the saved ASK signal (with modulation) is only 450 kB, which is equal to 1.5% of the recorded signal.

6.3. FSK record and replay

The results of the test on the record and replay phases of the FSK signal are depicted in Figures 2a and 2b, respectively. Both figures present the same plots as described for the ASK tests.



(a) FSK signal demodulation and record.



(b) FSK signal modulation and replay.

From the analysis of the demodulation and record phase (Figure 2a), it is apparent that the signal is transmitted over two channels (as clearly observable from the RX RF window), although it is possible to demodulate it using only one of the two channels. It should be noted, however, that this behavior is specific to our test vehicle and, despite not being the classical implementation of rolling code authentication over FSK, it is now a common practice. Moreover, it is noticeable that in this scenario, the jamming signal needs to be sent over both channels to effectively prevent the vehicle from receiving the signal.

In the modulation and replay phase, however, we can send the signal over one of the two channels to complete the attack. The overall size of the recorded FSK signal (without demodulation) is 57.43 MB (7527600 samples at a sampling rate of 1 Mbps), while the saved FSK signal (with modulation) is only 1.6 MB, which is equal to 2.78% of the recorded signal.

7. Conclusions

In this study, we have described an implementation of the RollJam attack using Software Defined Radios (SDRs) and investigated the practical implications of security vulnerabilities in rolling code-based authentication systems. In our experimental evaluation, we demonstrate the effectiveness and signal demodulation capabilities of our implementation, showcasing its suitability for a variety of rolling code systems using Amplitude Shift Keying (ASK) and Frequency Shift Keying (FSK) modulations. Notably, our practical assessment, conducted with our SDR setup, illustrates the RollJam attack's effectiveness at distances of up to 5 meters from the target system, with the key fob in close proximity to the vehicle. Placing the jamming signal as close as 2 meters prevents any signal from being received. Additionally, we have successfully optimized signal storage requirements, achieving a reduction of over 98% and 96% compared to traditional methods, through adept demodulation techniques for both ASK and FSK modulated signals, respectively. In an effort to promote collaborative progress in security research, we have made our RollJam attack implementation publicly available to the scientific community upon request [2]. This work aims to empower other researchers in strengthening defenses against wireless exploits, thus contributing to the continuous improvement of security protocols.

Acknowledgments

This work was partially supported by projects SERICS (PE00000014) under the MUR National Recovery and Resilience Plan funded by the European Union - NextGenerationEU and "FuSeCar" funded by the MUR Progetti di Ricerca di Rilevante Interesse Nazionale (PRIN) Bando 2022 - grant 2022W3EPEP

References

[1] S. Kamkar, Drive it like you hacked it, 2015. URL: https://samy.pl/defcon2015/.

[2] ACES - Automotive, Cyber-Physical and Embedded Security Group, RollJam implementation, 2024. URL: https://github.com/SECloudUNIMORE/ACES/tree/master/SDR-RollJam-access.

[3] S. Indesteege, N. Keller, O. Dunkelman, E. Biham, B. Preneel, A practical attack on keeloq, in: N. Smart (Ed.), Advances in Cryptology – EUROCRYPT 2008, Springer Berlin Heidelberg, Berlin, Heidelberg, 2008, pp. 1–18.

[4] W. Aerts, E. Biham, D. Moitie, E. Mulder, O. Dunkelman, S. Indesteege, N. Keller, B. Preneel, G. Vandenbosch, I. Verbauwhede, A practical attack on keeloq, J. Cryptology 25 (2012) 136–157. doi:10.1007/s00145-010-9091-9.

[5] M. Kasper, T. Kasper, A. Moradi, C. Paar, Breaking keeloq in a flash: On extracting keys at lightning speed, in: Proceedings of the 2nd International Conference on Cryptology in Africa: Progress in Cryptology, AFRICACRYPT '09, Springer-Verlag, Berlin, Heidelberg, 2009, p. 403–420. URL: https://doi.org/10.1007/978-3-642-02384-2_25. doi:10.1007/978-3-642-02384-2_25.

[6] F. D. Garcia, D. Oswald, T. Kasper, P. Pavlidès, Lock it and still lose it - on the (in)security of

automotive remote keyless entry systems, in: Proceedings of the 25th USENIX Conference on Security Symposium, SEC'16, USENIX Association, USA, 2016, p. 929–944.
[7] F. Maggi, A. Guglielmini, Rfquack: A universal hardware-software toolkit for wireless protocol (security) analysis and research, 2021. arXiv:2104.02551.
[8] Z. Depp, H. B. Tulay, E. Koksal, Enhanced Vehicular Roll-Jam Attack using a Known Noise Source, in: Symposium on Vehicles Security and Privacy (VehicleSec), 2023, 2023, pp. 1–6.
[9] s23s0n, jam-replay-rf, 2019. URL: https://github.com/s34s0n/jam-replay-rf.
[10] CR11CS, Rolljam-315mhz-433mhz, 2022. URL: https://github.com/CR11CS/RollJam-315MHz-433MHz.
[11] mcore1976, cc1101-tool, 2023. URL: https://github.com/mcore1976/cc1101-tool.
[12] eliddell1, Rolljam, 2018. URL: https://github.com/eliddell1/RollJam.
[13] exploitagency, rfcat-rolljam, 2017. URL: https://github.com/exploitagency/rfcat-rolljam.
[14] ghostlulzhacks, rolljam, 2016. URL: https://github.com/ghostlulzhacks/rolljam.
[15] AndrewMohawk, Rfcathelpers, 2015. URL: https://github.com/AndrewMohawk/RfCatHelpers.
[16] lucaercoli, rolling-code-grabber, 2020. URL: https://github.com/lucaercoli/rolling-code-grabber.
[17] H. Al-Tous, I. Barhumi, N. Al-Dhahir, Atomic-norm for joint data recovery and narrow-band interference mitigation in ofdm systems, in: 2016 IEEE 27th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC), 2016, pp. 1–5. doi:10.1109/PIMRC.2016.7794716.
[18] M. Steer, Microwave and RF Design: Networks \, Open textbook library, North Carolina State University Libraries, 2019. URL: https://books.google.it/books?id=fnAiyAEACAAJ.
[19] T. Eisenbarth, T. Kasper, C. Paar, S. Indesteege, Keeloq, Springer US, Boston, MA, 2011, pp. 671–673. URL: https://doi.org/10.1007/978-1-4419-5906-5_587. doi:10.1007/978-1-4419-5906-5_587.
[20] A. Bogdanov, Cryptanalysis of the keeloq block cipher, Cryptology ePrint Archive, Paper 2007/055, 2007. URL: https://eprint.iacr.org/2007/055, https://eprint.iacr.org/2007/055.
[21] Ettus Research, USRP B200, 2024. URL: https://www.ettus.com/all-products/UB200-KIT/.
[22] GNU Radio Project, GNU Radio - The Free and Open Software Radio Ecosystem, 2001. URL: https://www.gnuradio.org/.
[23] Ettus Research, Vert 2450 antenna, 2024. URL: https://www.ettus.com/all-products/vert2450/.