COS30015 – IT Security

Individual Research Assignment

The RollJam Attack and Its Applications

Jai Lafferty (101610772)

Submitted: ADD SUBMIT TIME

Due: 27/10/19 17:30:00

INTRODUCTION

The RollJam attack is a radio frequency replay-based attack designed to defeat the rolling code security used in most modern vehicles’ keyless entry systems [1]. The technique was pioneered in 2015 by Sami Kamkar, presenting his findings at DEF CON 23 [2]. The biggest factor into the attack’s popularity was its surprising low cost, the original attack being performed with only $32 USD.

This report was initially inspired by the jamming-based attack (denial of service) on a SimpliSafe Home Security Alarm performed by the ‘LockPickingLawyer’ [3]. This attack was made possible by jamming the radio frequencies used by the alarm with an inexpensive radio transmitter, allowing an intruder silent access to the premises. With limited prior knowledge on radio frequencies, the question was posed whether a similar approach could work on vehicles also as they used radio frequencies for their keyless entry systems.

The rolling code encryption process is a method of transmitting an encoded data signal in which a replay attack is not possible [4]. Rather than send a repeated code for each action (ie, lock/unlock), a sequence counter was implemented after each instance of the transmitter being used to generate a new digital code for both transmitter and receiver. Performing a replay attack on a vehicle with rolling code transmission would be useless due to the original code having already been used and thus removed from the list of valid codes.

This forced the traditional replay attack to be modified if it was to be used on a rolling code equipped vehicle. The complete anatomy of the RollJam attack is as follows.

**Parties:** User, Interceptor, Vehicle

1. **Jam at a slightly deviated frequency**

Flooding the receiver’s (vehicle’s) receiving frequency stops it from being to differentiate the authentic code and the noise produced by the jammer, causing the vehicle to stop receiving entirely.

1. **Receive at a frequency with a tight receive filter bandwidth to evade jamming**

If the transmission frequency is known, being able to intercept the code is possible given the interceptor is receiving a very limited frequency and is filtering the jamming signal.

1. **First key press from user**

This first attempt to unlock the vehicle will not work due to the jamming signal. However, the interceptor is able to save this transmission.

1. **Second key press from user**

It was observed that the vast majority of users when faced with a malfunctioning transmitter would repeat the same transmission almost immediately. It is this social expectation that allows RollJam to be useful. Again, this transmission is jammed and not received, and again this transmission is saved by the interceptor. Jamming stops at this step.

1. **Replay the *first* transmission**

By replaying the first transmission, two major outcomes are met. Firstly, the user’s expectations of a working transmitter are met, allowing them access to their vehicle without suspicion. And secondly, still having the second code allows future use of that transmission as it is still on the valid list of codes.

This attack while in essence is relatively simple, can be difficult to implement due to changing variables such as frequencies and modulation types. Both to be closer examined in more depth later in this report.

Further, this report will primarily discuss the RollJam attack in relation to a specific scenario and specific hardware as the RollJam attack merely defines the steps required to exploit radio frequency based keyless entry.

**TARGET AND TOOLS**

The target for this attack is a 2007 BMW E87 120i. This target was used solely due to having ownership of this vehicle. Performing this attack on others’ vehicles is strongly ill-advised due to legal implications.

The target features a keyless access fob with an FCC ID of KR55WK49127 and an operating frequency of 315MHz (USA variant). The methodology section of this report covers how one can research the operating frequency without brute forcing.

The original RollJam device consisted of two CC1101 chips and a Teensy 3.1, rather than using tools which had these chips built in, Kamkar saved a lot of money at the expense of not having other prebuilt tools to his disposal [Fig1 - Appendix]. The device devised for the purpose of this report however utilised more expensive, yet easier to use hardware [Fig 2 - Appendix].

The following tools were used for this implementation:

1. **RTL-SDR Dongle (Realtek RTL2832U Chip)**

This Software Defined Radio dongle operates between 24-1766MHz [5] and allows the use of capturing and ‘listening’ of radio frequencies on a computer. Originally intended for TV tuning, this dongle has uses such as listening to FM Radio and receiving weather satellite images [6]. This tool is to be used in order to identify operating frequencies and confirming the transmission of codes from the attacker.

1. **YARD Stick One (TI CC1111 Chip) – Great Scott Gadgets**

The “YARD Stick One is a sub-1 GHz wireless test tool controlled by your computer.” [7] This dongle is capable of half-duplex transmitting and receiving with a large pool of available modulations. The largest advantage of this tool is the preinstalled RfCat firmware allowing an interactive python shell to interface with it out of the box [8].

1. **Raspberry Pi 2B**

Running a fresh install of the Raspbian Linux distro, the Raspberry Pi is a great low-cost option for computation. Note that a lot of the software tools necessary are only available on Linux. Further, using a Raspberry Pi allows the separation of personal computing and programming libraries avoiding the case of an unlikely collision.

1. **GQRX**

A Linux based RTL-SDR visualisation tool. Used for visually identifying transmissions.

1. **GNURADIO**

An open source software radio system [9]. Allows signal processing of radio frequencies. Used for demodulation of codes sent from the key fob.

1. **RfCat**

RfCat is the Python library that interfaces to dongles such as the YARD Stick One. Allows for either interactive programming in the Python shell or light scripting.

1. **Short Copper Cable**

Used as an antenna on the Raspberry Pi for jamming local signals.

METHODOLOGY

**Research & Reconnaissance**

Firstly, it is necessary to understand as much as possible about the target before continuing as not to waste time or effort. Finding the operating frequency of a car is relatively simple, the observed best way was to perform a search with the format “CAR\_MODEL remote fcc”. This would return the correct key in addition to the needed FCC ID. “An FCC ID is a unique identifier assigned to a device registered with the United States Federal Communications Commission.” [10]. In order for a wireless device to be sold within the US they must be registered with an FCC ID.

This allows the ability to search using fccid.io for specific engineering data about the key and it’s use, including Block Diagrams, Technical Descriptions and Schematics among other documents. The database also alerts you to different frequencies being used for different countries. In the case of the BMW E87, the fob operates at 868.4MHz and 434.2MHz in Europe, a largely different frequency to the 315MHz of the target model.

Before confirmation of the key fob’s operating frequency, the Raspberry Pi needs to be set up for use.

The following was installed on the Raspberry Pi:

* Raspbian Buster September 2019 (Kernel Version 4.19) [11]
* Gnuradio 3.7.13.5 [12]
* QT Graphical Toolkit 5 [13]
* GQRX 2.11 [14]
* Python 3.8 [15]
* Libusb 1.0.22b4 [16]
* Future 0.18.1 [17]
* Rpitx 2 [18]
* RfCat [8]

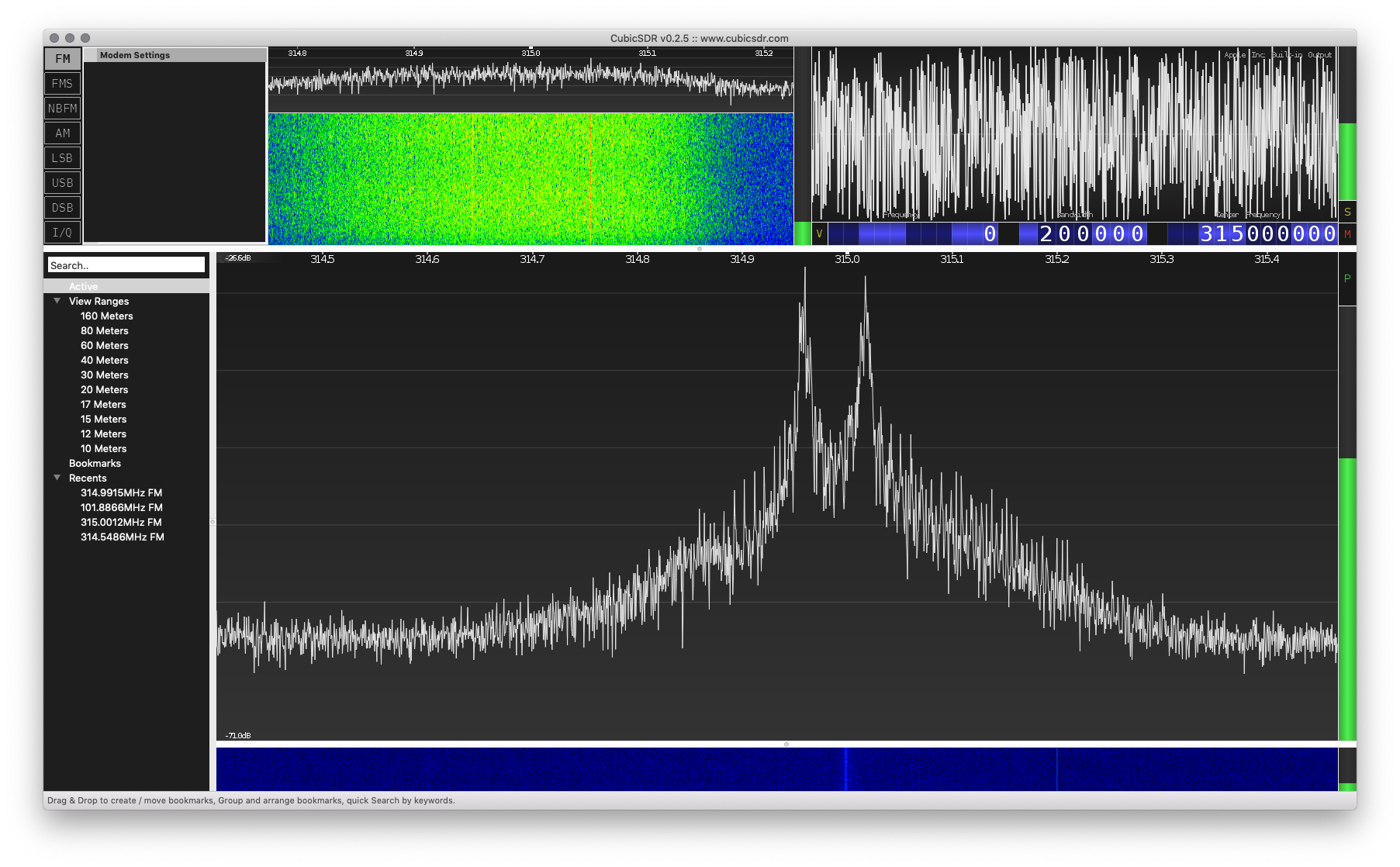
With the target’s operating frequency known we can test this using a visual spectrum analysis tool. Using GQRX centred around 315MHz with the attached RTL-SDR dongle, pressing the unlock button on the key fob should yield results similar to Figure 3, a spike in activity around 315MHz.

Determining the modulation type is next, the modulation type describes how 1s and 0s are to be sent over radio.

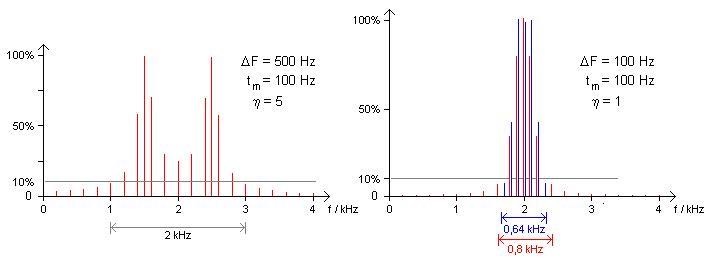
The most common methods used in car key fobs are:

* Amplitude Shift Keying (ASK)
  + “In amplitude shift keying, the carrier wave amplitude is changed between discrete levels (usually two) in accordance with the digital data.” [19]
* Frequency Shift Keying (FSK)
  + “In frequency shift keying, the carrier frequency is changed between discrete values.” [19]
  + Due to this, FSK is visually distinguishable from ASK as it operates on 2 frequencies rather than centering around the single frequency as ASK does.

Visually, it is possible discern whether the target’s modulation type is ASK or FSK.



*Figure 3 – Unlock code in Cubic SDR v0.2.5[[1]](#footnote-1)*

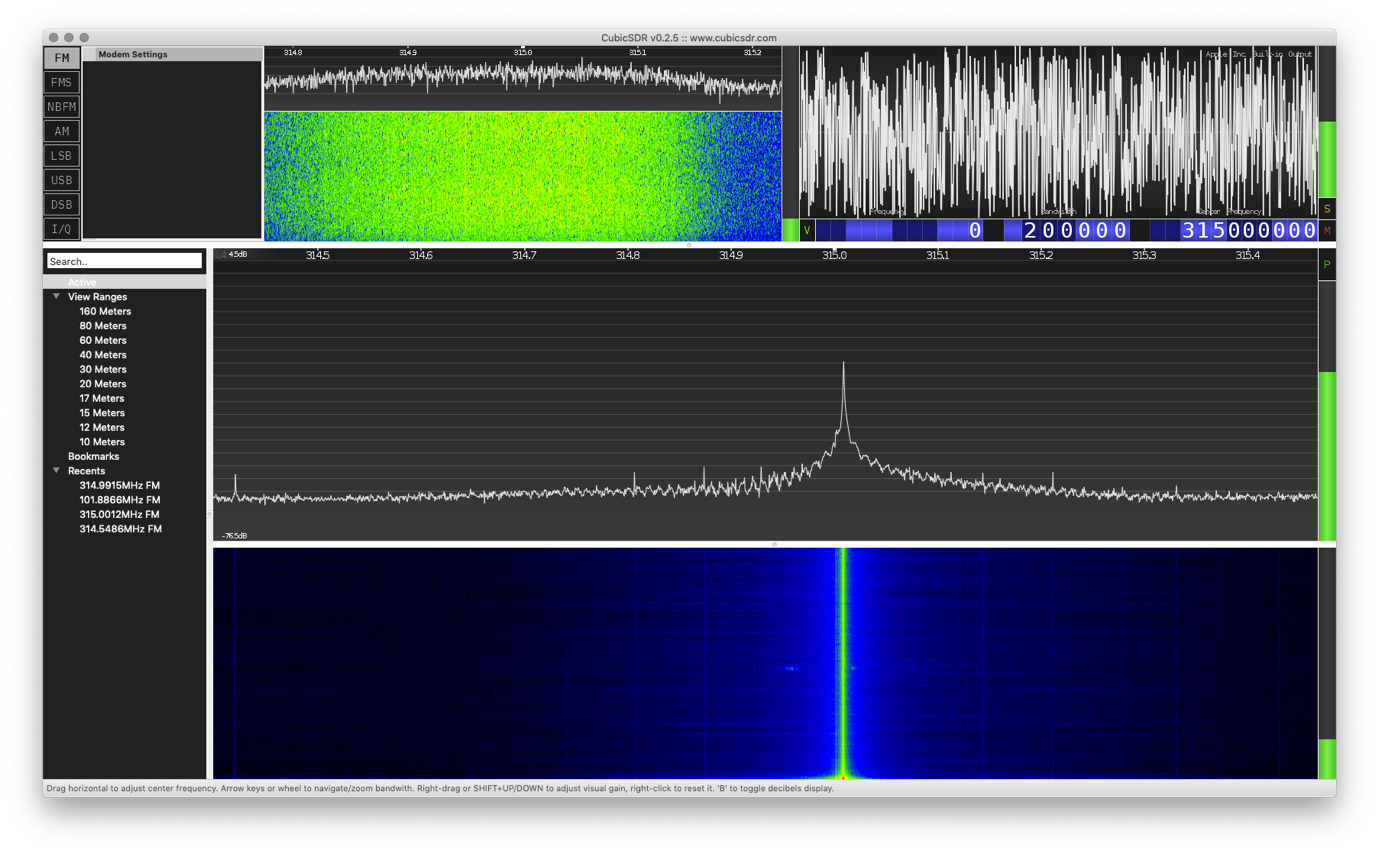


*Figure 4 – FSK (left) vs ASK(right) spectrum output [20]*

Evidently, the spectrum from the target key fob more closely aligns with FSK as outlined in Figure 4. Knowing the Modulation method allows the YARD Stick One to transmit the unlock code correctly.

**Jamming**

The Rpitx tool a general radio frequency transmitter [18]. This allows no special hardware other than a cable antenna attached to a GPIO pin. Given a frequency, this tool’s predefined *Carrier* setting will transmit a steady stream of data to that frequency. Figure 5 displays the jamming signal on the spectrum analyser.



*Figure 5 – Jamming Signal*

Akin to that of a DOS attack, this method of jamming renders the vehicle’s receiver useless as it’s unable to distinguish genuine transmissions from that of the jammer.

**Listening and Replaying**

As the RfCat library is written in Python, writing a script to listen and replay is relatively simple. The code used for this report is a modified/combined version of the *ghostlulzhacks* Python script [22] and the *trishmapow* Python script [23]. Changing device settings such as operating frequency and modulation type generates a relevant script to the target vehicle. Access to the code and demonstration video can be found via the link found in references [24].

Pseudocode for this script are as follows:

d = rfCatDevice *#YARD Stick One*

setup(d) *#Sets target frequency, modulation, bandwidth etc.*

captureData = []

*# capture*

**for** \_ in range(2): *#2 packets required*

data = d.receiveData()

encodedData = data.encodeAs(hex)

*#Jamming signal is "FFF..." and hence needs filtering*

**if** (encodedData.count(**'F'**) < 350): *#arbitrary number of f's based on packet length*

captureData.add(encodedData)

*# replay*

**for** i in range (2):

raw\_input() *#Required input to replay*

send\_packet = daptureData[1].toBitArray() *#pack data in sendable form*

d.RFxmit(send\_packet) *#send data*

It is important to have the settings of the YARD Stick One match that of the key fob otherwise irrelevant or incorrect data would be received. It is not largely important to actually decode the transmission as the script simply replays what it receives, regardless of the data it receives.

DISCUSSION & RECOMMENDATIONS

While this attack is not exactly trivial, it can be replicated with an actor of limited radio frequency experience given they have access to suitable tools and are willing to spend a short amount of time leveraging the vast amount of data on this topic available on the internet. This results in most modern vehicles being vulnerable to an attack such as this. While rolling code systems were put in place to defeat the standard replay attack, it is evident that further defences need to be developed.

Further, this attack is almost impossible to mitigate as an owner of these vulnerable

vehicles. With rolling code authentication being a feature of the vehicle, there is no

behavioural change that can guarantee the safety of an owner’s vehicle, the issue lies with

the product which may frustrate users.

Increasing key fob security has been done previously with methods such as data encryption

[]. While this does protect against “today’s common attack tricks such as brute-force key

guess attack(s),” encrypting the data portion of the data packet does not mitigate replay attacks where the data itself isn’t the focus of the exploit. The focus of the exploit lies with the authentication of the data packet.

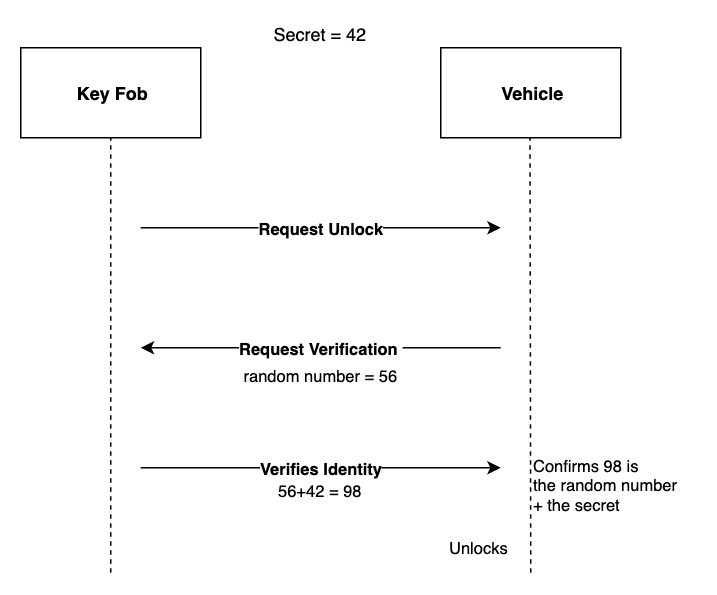
Authentication of the legitimate key fob can completed, and is recommended by this report, via either of the following methods:

* Challenge/Response based authentication
* Timer based authentication

**Challenge/Response Based Authentication**

Also recommended and implemented by the encryption method of S. Fook & V. Foo [], the Challenge/Response protocol suggests that, for example, the vehicle challenges an unlock request, challenging its identity forcing the key to respond in order for a successful unlock. Figure 6 describes this with a sequence diagram.

For a more detailed example, the request unlock would remain the same as the current request sent by car fobs, the difference being the car fob is programmed by the car to contain a secret unique number that is never transmitted but known by both parties. Not transmitting this secret. Authentication is achieved by the vehicle randomly generating a number and expecting in return a summed result of the random and secret numbers.



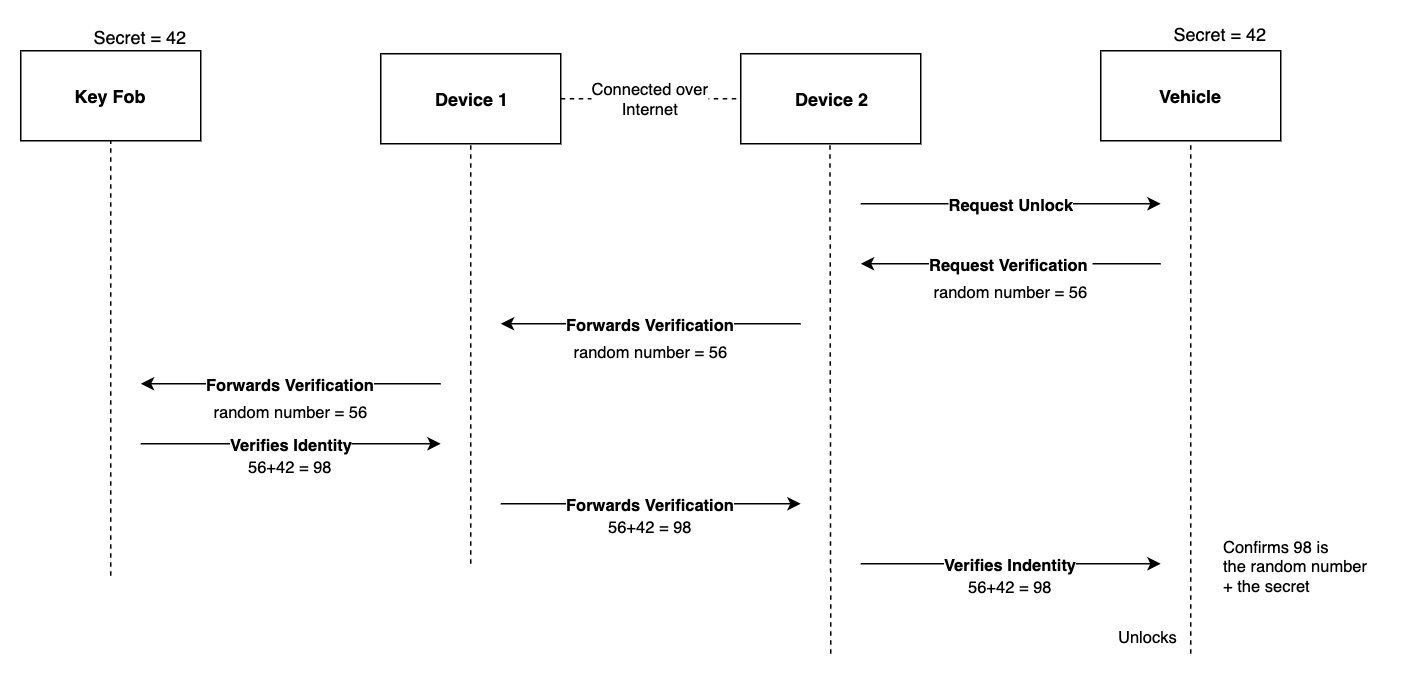
*Figure 6 – Challenge Response sequence diagram*

Exploiting this would require somehow knowing the secret key. Any jamming attempt that results in the user pressing unlock a secondary time would simply request another unlock from the vehicle, resetting the random number. This negates the ability to ‘store’ an unlock for later like the rolling code defence allows.

Where this method falls short however is in the case of interference. Tripling the amount of transmissions invited triple the surface area for interference to render a transmission unreadable. This makes the key fob significantly less reliable. Further, this now requires a 2-party man in the middle attack, outlined by Figure 7. While still possible, this attack has a greater barrier to entry with two internet connected devices and multiple scripts necessary to successfully gain entry. This has been demonstrated criminally in the UK []

This method of attack would work well in environments where the owner of the vehicle is far enough from the vehicle to notice a malicious actor waiting nearby the vehicle (eg. Shopping centre). Note that as this attack is a multi-layered replay attack, no decrypting or demodulation is required to gain access to the vehicle.

Further, implementing a rolling code in addition to the Challenge/Response protocol should sufficiently eliminate this type of attack due to actors needing the user to physically press the key fob’s unlock button to receive the challenge from the vehicle.

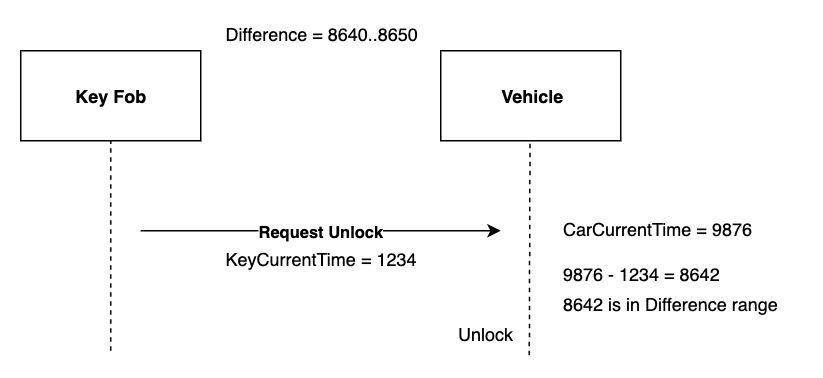


*Figure 7 – Challenge/Response Men in the Middle Attack*

**Timing Based Authentication**

Another reliable method of authentication would be using the system time of both the key and the vehicle. The time can be used as a guarantee that the physical button was pressed within an acceptable threshold of the car receiving the unlock request. This renders RollJam useless as the stored transmission will expire after what’s deemed a safe amount of time (assumes milliseconds). The KeeLoq product, used in many wirelessly unlocked devices, unveiled their implementation of Timing Based Authentication in Ultimate KeeLoq [] in 2014.

Figure 8 displays a sequence diagram of how timing-based authentication works.



In more detail, this method merely compared a delta value, the difference between devices’ times, and checks that there’s a strong probability the signal was sent on time.

Breaking this form of authentication requires knowing either devices’ clock value and being able to modify the packet sent. This method is considered extremely difficult due to firstly decrypting a packet sent from the fob in order to store its current time. Which needs to be done as fast as possible in order to get an accurate result. Once decrypted, a clock needs to be started as soon as possible to make sure whichever time is being sent in malicious packets is accurate. Further, perform RollJam as per normal. The stored packet will need to be altered with an updated time, being an addition of the first packets current time and elapsed time since decryption. Should the accuracy be in the range of milliseconds there is a small chance of unlocking the car.

Again, breaking this form of authentication is as close to impossible as is possible currently. Not only is it highly unlikely to be within the delta range of the vehicle, this required expert knowledge on demodulation, decryption and packet spoofing.

**Characteristics of Secure Authentication**

What both the aforementioned authentication techniques shared is primarily some form of synchronisation between the key fob and the vehicle. In the case of challenge/response authentication, the devices are synchronised to the secret number. Whereas the timer-based authentication synchronises the devices with the secret difference in local time.

Keeping this form of authentication secure relies on not transmitting data that could allow intruders to gain access to the secret value. This works in a similar way to private key encryption does, without the key all transmitted data is invalid.

In the event of a compromised key, system designers can factor in some sort of refreshing of the secret key. The challenge/response method could have the vehicle and fob reset their key to an agreed upon random number upon vehicle start, and the same could apply to the timer-based authentication, resetting the fob local time upon vehicle engine start.

REFERENCES

[1] A. Greenberg, "This Hacker's Tiny Device Unlocks Cars And Opens Garages", WIRED, 2015. [Online]. Available: https://www.wired.com/2015/08/hackers-tiny-device-unlocks-cars-opens-garages/. [Accessed: 21- Oct- 2019]

[2] S. Kamkar, DEF CON 23: Samy Kamkar - Drive it like you Hacked it. 2015 [Online]. Available: https://www.youtube.com/watch?v=UNgvShN4USU. [Accessed: 21- Oct- 2019]

[3] LockPickingLawyer, SimpliSafe Alarm Bypassed With a $2 Device From Amazon. 2019 [Online]. Available: https://www.youtube.com/watch?v=UlNkQJzw4oA. [Accessed: 21- Oct- 2019]

[4] Delphi Technologies Inc, "Rolling code encryption process for remote keyless entry system", EP1260942B1994.

[5] "Rtl-sdr - rtl-sdr - Open Source Mobile Communications", Osmocom.org, 2019. [Online]. Available: https://osmocom.org/projects/rtl-sdr/wiki/Rtl-sdr. [Accessed: 21- Oct- 2019]

[6] A. Wulff, "How to Download Live Images From Government Weather Satellites", Hackernoon.com, 2019. [Online]. Available: https://hackernoon.com/weather-sat-9620228789c8. [Accessed: 21- Oct- 2019]

[7] "YARD Stick One - Great Scott Gadgets", Greatscottgadgets.com, 2016. [Online]. Available: https://greatscottgadgets.com/yardstickone/. [Accessed: 21- Oct- 2019]

[8] "atlas0fd00m/rfcat", GitHub, 2019. [Online]. Available: https://github.com/atlas0fd00m/rfcat. [Accessed: 21- Oct- 2019]

[9] "What is GNU Radio? - GNU Radio", Wiki.gnuradio.org, 2017. [Online]. Available: https://wiki.gnuradio.org/index.php/What\_is\_GNU\_Radio%3F. [Accessed: 21- Oct- 2019]

[10] "FCC ID Search", FCC ID, 2019. [Online]. Available: https://fccid.io/. [Accessed: 21- Oct- 2019]

[11] "Download Raspbian for Raspberry Pi", Raspberry Pi, 2019. [Online]. Available: https://www.raspberrypi.org/downloads/raspbian/. [Accessed: 21- Oct- 2019]

[12] "InstallingGR - GNU Radio", Wiki.gnuradio.org, 2019. [Online]. Available: https://wiki.gnuradio.org/index.php/InstallingGR. [Accessed: 21- Oct- 2019]

[13] "Installing Qt5", PyPI, 2019. [Online]. Available: https://pypi.org/project/PyQt5/. [Accessed: 21- Oct- 2019]

[14] "Gqrx SDR for the Raspberry Pi – Gqrx SDR", Gqrx.dk, 2019. [Online]. Available: http://gqrx.dk/download/gqrx-sdr-for-the-raspberry-pi. [Accessed: 21- Oct- 2019]

[15] "Download Python", Python.org, 2019. [Online]. Available: https://www.python.org/downloads/. [Accessed: 21- Oct- 2019]

[16] "installing libusb", PyPI, 2019. [Online]. Available: https://pypi.org/project/libusb/. [Accessed: 21- Oct- 2019]

[17] "Installing future", PyPI, 2019. [Online]. Available: https://pypi.org/project/future/. [Accessed: 21- Oct- 2019]

[18] "F5OEO/rpitx", GitHub, 2019. [Online]. Available: https://github.com/F5OEO/rpitx. [Accessed: 21- Oct- 2019]

[19] "Introduction to Naval Weapons Engineering: Digital Communication Methods", Fas.org. [Online]. Available: https://fas.org/man/dod-101/navy/docs/es310/DigiComs/digicoms.htm. [Accessed: 21- Oct- 2019]

[20] R. M, "Frequency shift keying (FSK) in detail", indiastudychannel.com, 2013. [Online]. Available: https://www.indiastudychannel.com/resources/161009-Frequency-shift-keying-FSK-detail.aspx. [Accessed: 21- Oct- 2019]

[21] "CubicSDR | Cross-Platform and Open-Source Software Defined Radio Application", Cubicsdr.com, 2019. [Online]. Available: https://cubicsdr.com/. [Accessed: 21- Oct- 2019]

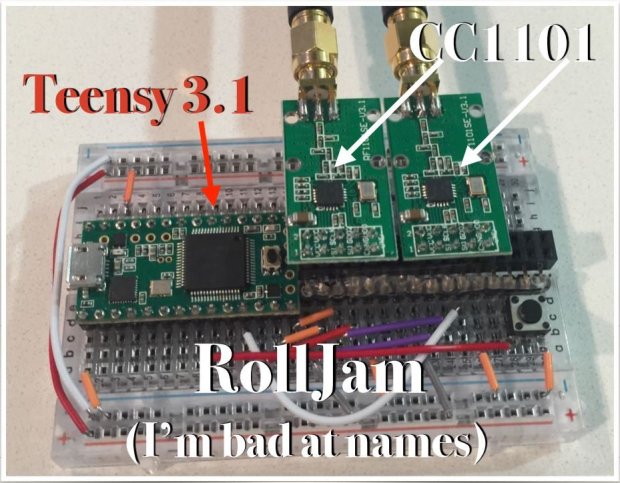
[22] "ghostlulzhacks/rolljam", GitHub, 2016. [Online]. Available: https://github.com/ghostlulzhacks/rolljam/blob/master/rf\_car\_replay.py. [Accessed: 21- Oct- 2019]

[23] "trishmapow/rf-jam-replay", GitHub, 2017. [Online]. Available: https://github.com/trishmapow/rf-jam-replay. [Accessed: 21- Oct- 2019]

[??] J. Steinberg, "Vulnerability In Car Keyless Entry Systems Allows Anyone To Open And Steal Your Vehicle", Forbes.com, 2017. [Online]. Available: https://www.forbes.com/sites/josephsteinberg/2015/05/12/vulnerability-in-car-keyless-entry-systems-allows-anyone-to-open-and-steal-your-car/#5029cd992442. [Accessed: 21- Oct- 2019]

[??] C. Toma, Introduction to Ultimate KEELOQ® Technology, 1st ed. Chandler, Arizona: Microchip Technology Inc., 2014 [Online]. Available: http://ww1.microchip.com/downloads/en/AppNotes/00001683A.pdf. [Accessed: 21- Oct- 2019]

APPENDIX



*Figure 1 – RollJam Hardware, Taken from Kamkars DEF CON Presentation [2]*

*Figure 2 – my implemtation (better caption)*

1. GQRX Screenshot recreated in MacOS with Cubic SDR [21] as to produce higher quality screen captures [↑](#footnote-ref-1)