Overview of some automotive RKE systems

Pierre Pavlidès

OWASP Gothenburg Day 2016

November 24, 2016

Slides at http://r.rogdham.net/26

Pierre Pavlidès

`@rogdham`

usenix
THE ADVANCED COMPUTING SYSTEMS ASSOCIATION

Lock It and Still Lose It—On the (In)Security of Automotive Remote Keyless Entry Systems

Flavio D. Garcia and David Oswald, *University of Birmingham;* Timo Kasper, *Kasper & Oswald GmbH;* Pierre Pavlidès, *University of Birmingham*

https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/garcia

Before we start

BBC | News | Sport | Weather | Shop | Earth | Travel | Capital | Culture | More | Search

NEWS

Home | Video | World | UK | Business | Tech | Science | Magazine | Entertainment & Arts | Health | World News TV | More

Technology

'Millions' of Volkswagen cars can be unlocked via hack

By Chris Baraniuk
Technology reporter

12 August 2016 | Technology

WIRED

ANDY GREENBERG SECURITY 08.10.16 4:29 PM

A NEW WIRELESS HACK CAN UNLOCK 100 MILLION VOLKSWAGENS

Agenda

Agenda

Car security mechanisms

Going inside the car:
→ Mechanical lock
→ RKE system
→ Smart key

Starting the engine and drive away:
→ Be inside the car first
→ Immobiliser (transponder)
→ Mechanical lock / ignition button

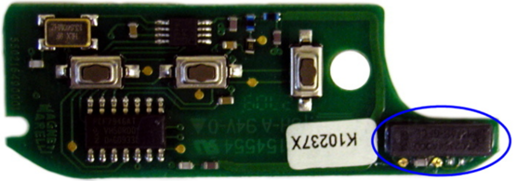
Once you are in the car, game over
→ Attacks on transponder
→ Access the onboard computer

In this talk: attacks on RKE systems

NOT in this talk: newest car "features" such as acces over Internet

Remote overview

Remote overview



RKE overview

RKE systems use a very simple protocol:

- Owner pushes a button
- The remote sends a message to the car
- The car reacts accordingly

Only a single one-way message

$\rightarrow$ Usual security protocols (e.g. challenge/response) don't work

Since it seems simple, let's create an RKE protocol from scratch

Action: open/close the doors, etc.

Tell the car if we want to close or open the doors:

$$< btn >$$

Issue

In a parking lot, we will open *all* cars at once!

Car or remote identifier

Add the car or remote identifier:

$$< UID || btn >$$

Using the remote identifier allows to invalidate specific remotes

Attack scenario

1. Eavesdrop a close signal
2. Create an open signal

Integrity

Crypto block	Key	Issue
Signature	Asymetric	Costly battery-wise
MAC	Symetric	Key management

Generate random shared key key_{UID} when the remote is linked with the car

$$< UID || \overbrace{btn}^{M} || \mathrm{MAC}_{key_{UID}}(M) >$$

Does not matter if M includes the UID

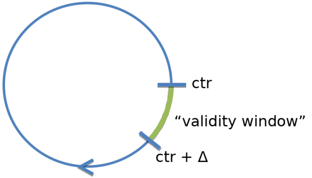
Attack scenario

Replay attacks

Token	Issue
Random[1]	Car needs to remember all used tokens
Time-based[2]	Precise clock in remote (and car)
Counter	Overflow, desynchronisation

Use a counter with a validity window

$$< UID || \overbrace{btn, ctr}^{M} || \mathrm{MAC}_{key_{UID}}(M) >$$



Rolling code (https://en.wikipedia.org/wiki/Rolling_code)

A rolling code [...] is used in keyless entry systems to prevent replay attacks

[1] e.g. anti-CSRF (ESAPI. . .)

[2] e.g. Google Authenticator (RFC 6238)

Any remaining problems?

$$< UID || \overbrace{btn, ctr}^{M} || \mathrm{MAC}_{key_{UID}}(M) >$$

Confidentiality
$\rightarrow$ Not sure we need it

Integrity: MAC

Availability
$\rightarrow$ Repeat the rolling code several times, hope for the best
$\rightarrow$ Worst case scenario: owner uses mechanical lock

Agenda

The VW systems

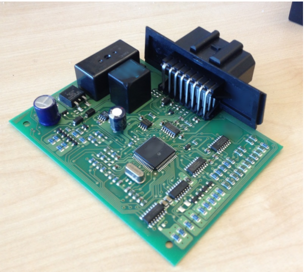
Amarok, (New) Beetle, Bora, Caddy, Crafter, e-Up, Eos, Fox, Golf 4, Golf 5, Golf 6, Golf Plus, Jetta, Lupo, Passat, Polo, T4, T5, Scirocco, Sharan, Tiguan, Touran, Up

AA1, Q3, R8, S3, TT, others

Alhambra, Altea, Arosa, Cordoba, Ibiza, Leon, MII, Toledo

ACity Go, Roomster, Fabia 1, Fabia 2, Octavia, Superb, Yeti

Dump ECU firmware and start reverse engineer assembly



```
; =============== S U B R O U T I N E =======================================

sub_F5C4:                                        ; CODE XREF: sub_E31D+5C↑p
                pshd
                pshx
                leas    -$C,sp
                anda    #$3F ; '?'
                clrx
                addd    #$8000
                bcc     loc_F5D2
                inx

loc_F5D2:                                        ; CODE XREF: sub_F5C4+B↑j
                std     4,sp
                ldd     $14,sp
                ldx     $12,sp
                subd    $E,sp
                sbex    $C,sp
```

VW-1 scheme

Used approx. until 2005

$$< \mathrm{f}(\mathit{UID})||\mathrm{g}(\mathit{ctr})||\mathit{btn} >$$

f and g are deterministic functions

Issues

No shared secret, no intergrity checks...
$\rightarrow$ Security through obscurity (obfuscation)

VW-2 scheme

Used approx. since 2004 (VW-2)

$$< start_2 || \mathrm{AUT64}_{key_2}(UID, ctr, btn) || btn >$$

$start_2$ is a fixed prefix
$\mathrm{AUT64}$ is a cipher

Issue

Use encryption for integrity

Issue

One worldwide unique key

VW-2, VW-3, VW-4 schemes

Used approx. since 2004 (VW-2) / 2006 (VW-3) / 2009 (VW-4)

$$\begin{cases} \text{VW-2:} & < start_2 || \mathrm{AUT64}_{key_2}(UID, ctr, btn) || btn > \\ \text{VW-3:} & < start_3 || \mathrm{AUT64}_{key_3}(UID, ctr, btn) || btn > \\ \text{VW-4:} & < start_4 || \mathrm{XTEA}_{key_4}(UID, ctr, btn) || btn > \end{cases}$$

Issue

Still worldwide unique key

The VW systems

$$\begin{cases} \text{VW-1:} & < \mathrm{f}(\mathit{UID}) || \mathrm{g}(\mathit{ctr}) || \mathit{btn} > \\ \text{VW-2:} & < \mathit{start}_2 || \mathrm{AUT64}_{\mathit{key}_2}(\mathit{UID}, \mathit{ctr}, \mathit{btn}) || \mathit{btn} > \\ \text{VW-3:} & < \mathit{start}_3 || \mathrm{AUT64}_{\mathit{key}_3}(\mathit{UID}, \mathit{ctr}, \mathit{btn}) || \mathit{btn} > \\ \text{VW-4:} & < \mathit{start}_4 || \mathrm{XTEA}_{\mathit{key}_4}(\mathit{UID}, \mathit{ctr}, \mathit{btn}) || \mathit{btn} > \end{cases}$$

Attack scenario

Possible to clone a remote if we capture a single rolling code

Impact

Most VW group vehicles after 2000
Except the ones using Golf 7 (MQB) platform

Agenda

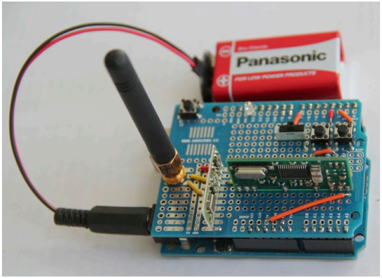
The Hitag2 system

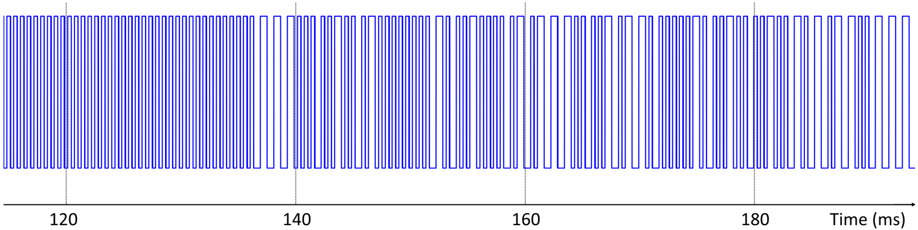
We verified our findings in practice by building a key emulator and then unlocking and locking the vehicles with newly generated rolling codes:

Manufacturer	Model	Year
Alfa Romeo	Giulietta	2010
Chevrolet	Cruze Hatchback	2012
Citroen	Nemo	2009
Dacia	Logan II	2012
Fiat	Punto	2016
Ford	Ka	2009, 2016
Lancia	Delta	2009
Mitsubishi	Colt	2004
Nissan	Micra	2006
Opel	Vectra	2008
Opel	Combo	2016
Peugeot	207	2010
Peugeot	Boxer	2016
Renault	Clio	2011
Renault	Master	2011

The vehicles in the above list are our own and also from colleagues and friends who volunteered.

Analysis: black box reverse-engineering

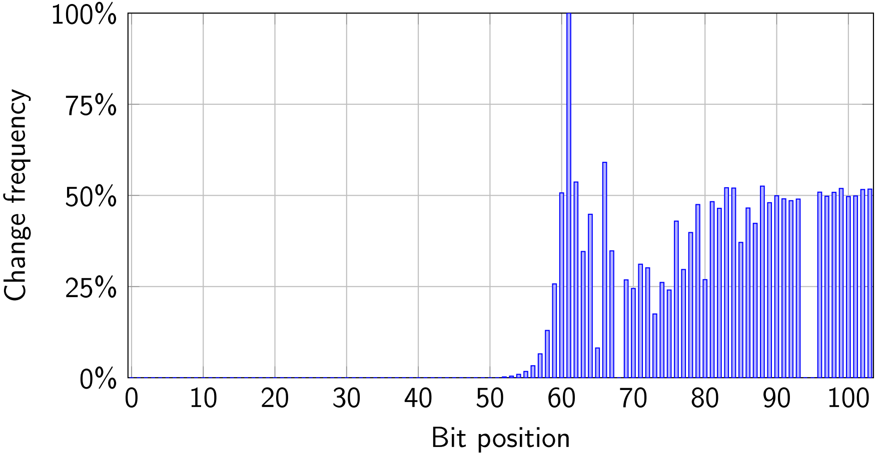




Analysis: bitstreams

```
00 01 53 d0 11 6b 21 b1 d1 d2 3b e6 b7
00 01 53 d0 11 6b 21 b7 d1 df 62 16 15
00 01 53 d0 11 6b 21 bb f3 96 2d 8e a5
00 01 53 d0 11 6b 21 bd f2 9a 77 3a 40
00 01 53 d0 11 6b 21 c3 d5 c6 57 22 7d
00 01 53 d0 11 6b 21 c5 d5 ce 2b 22 0f
00 01 53 d0 11 6b 21 c9 f5 4b d5 ee 94
00 01 53 d0 11 6b 21 cf f5 c6 c5 0a eb
00 01 53 d0 11 6b 21 d3 75 ee 77 7e 99
00 01 53 d0 11 6b 21 d5 75 e7 15 92 18
00 01 53 d0 11 6b 21 d8 55 f3 fb 2a 77
00 01 53 d0 11 6b 21 de 55 f7 c1 de bb
00 01 53 d0 11 6b 21 e3 71 d8 6e 06 fa
00 01 53 d0 11 6b 21 e5 75 dd 3d ca 62
00 01 53 d0 11 6b 21 e8 55 5b 24 76 6c
00 01 53 d0 11 6b 21 ee 51 d4 76 fa 3f
00 01 53 d0 11 6b 21 f3 d1 d7 31 62 7e
00 01 53 d0 11 6b 21 f5 d1 d4 71 d6 8f
00 01 53 d0 11 6b 21 f9 f1 cf 86 9a 03
00 01 53 d0 11 6b 21 ff f0 ce ed 36 c2
```

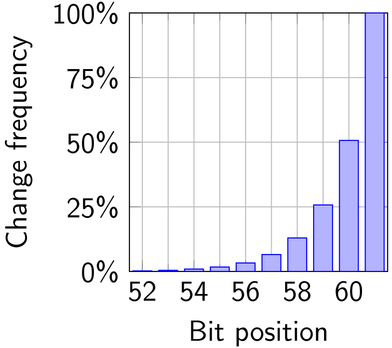
Frequency analysis: changes

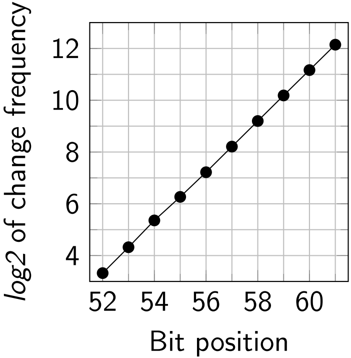


Computed over $> 2^{12}$ rolling codes from the same remote

→ First 51 bits don't change on this remote

Frequency analysis: closer look





Exponential increase

→ Counter!

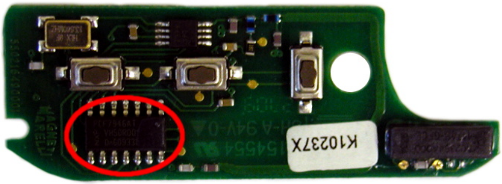
Blackbox analysis

$$< start||UID||btn||ctr||ks||chk >$$

Unknown keystream *ks*
Checksum *chk*: XOR all other bytes

We need some external information on how *ks* is computed

A closer look at the remote



Philips Semiconductors

Product Profile PCF7946AT

Security Transponder plus Remote Keyless Entry, HITAG2^{PLUS}

Features

- Compatible with Security Transponder, PCF7936AS.
- Rolling Code Generator for keyless entry
- 14-pin SO package

Transponder

- 64/32 bit mutual authentication
- 32 bit unique device identification number
- Fast authentication, 39ms

General Description

The HITAG2^{PLUS} is a high performance monolithic Security Transponder and Remote Keyless Entry Chip ideally suited for car immobiliser applications that incorporate keyless entry functions.

The HITAG2^{PLUS} transponder circuitry is compatible with the Security Transponder PCF7936AS to support mixed systems using a HITAG2^{PLUS} and a standard Security Transponder, PCF7936AS at the same time.

The Transponder circuitry meets the security and

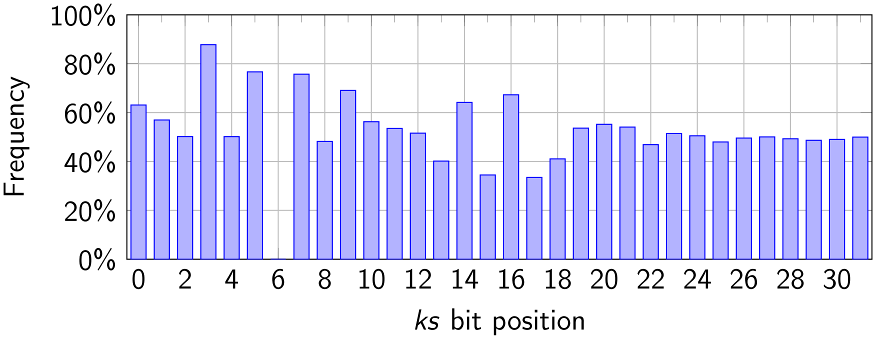
Hitag2

Stream cipher used in transponders

$$\text{Hitag2}(\text{key} = (48 \text{ bits}), \text{serial} = (32 \text{ bits}), \text{iv} = (32 \text{ bits}))$$

Reverse engineered, several attacks to recover the key from outputs

Re: frequency analysis of remote outputs



Blackbox analysis

$$< start||UID||btn||ctr||ks||chk >$$

Hitag2 is used, so:

$$ks = ???(\ \mathrm{Hitag2}(\mathrm{key} = ???, \mathrm{serial} = ???, \mathrm{iv} = ???)\ )$$

UID is 32 bits, just like the *serial*...

$$ks = ???(\ \mathrm{Hitag2}(\mathrm{key} = ???, \mathrm{serial} = UID, \mathrm{iv} = ???)\ )$$

But what to use for the *key*?

Mixed remotes: RKE + transponder

Philips Semiconductors

Product Profile — PCF7946AT

Security Transponder plus Remote Keyless Entry, HITAG2^{PLUS}

Features

- Compatible with Security Transponder, PCF7936AS.
- Rolling Code Generator for keyless entry
- 14-pin SO package

Transponder

- 64/32 bit mutual authentication
- 32 bit unique device identification number
- Fast authentication, 39ms
- 48 bit Secret Key

General Description

The HITAG2^{PLUS} is a high performance monolithic Security Transponder and Remote Keyless Entry Chip ideally suited for car immobiliser applications that incorporate keyless entry functions.

The HITAG2^{PLUS} transponder circuitry is compatible with the Security Transponder PCF7936AS to support mixed systems using a HITAG2^{PLUS} and a standard Security Transponder, PCF7936AS at the same time.

The Transponder circuitry meets the security and

Transponder reader

Transponder memory

Identifier	P0	53D0116B	R	
PSW / ISK low	P1	4D494B52	R	W
NA / ISK high	P2	00004F4E	R	W
Configuration	P3	00AA4854	R	W
User page 0	P4	4D494B52	R	W
User page 1	P5	00004F4E	R	W
User page 2	P6	6515F2D5	R	W
User page 3	P7	00000000	R	W

Test case for Hitag2

Same value as in the test case of the Hitag2 code:

User page 0	P4	4D494B52
User page 1	P5	00004F4E

```
// "MIKRON"      =  O  N  M  I  K  R
// Key           = 4F 4E 4D 49 4B 52     - Secret 48-bit key
// Serial        = 49 43 57 69           - Serial number
// Random        = 65 6E 45 72           - Random IV
state = hitag2_init(rev64(0x524B494D4E4FUL), rev32(0x69574349),
                    rev32(0x72456E65));
```

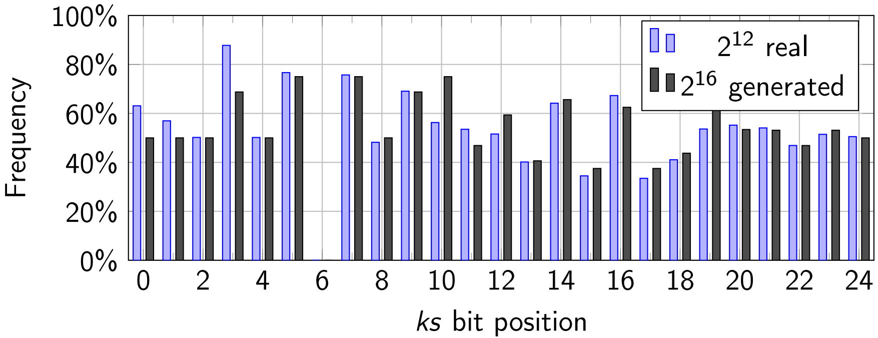
→ That was a blank remote

Progress so far

Assuming we have everything right:

$$ks = ???(\ \mathrm{Hitag2}(\mathrm{key} = \texttt{0x4f4e4d494b52}, \mathrm{serial} = UID, \mathrm{iv} = ???)\ )$$

Remember Hitag2 output is not random? → Generate random *ivs*



Progress so far

Confident that we are right:

$$\mathit{ks} = ???(\ \mathrm{Hitag2}(\mathrm{key} = \texttt{0x4f4e4d494b52}, \mathrm{serial} = \mathit{UID}, \mathrm{iv} = ???)\ )$$

Even better: no output mangling:

$$\mathit{ks} = \mathrm{Hitag2}(\mathrm{key} = \texttt{0x4f4e4d494b52}, \mathrm{serial} = \mathit{UID}, \mathrm{iv} = ???)$$

$\rightarrow$ All is left is the *iv*

Finding the IV

There is some function φ so that $iv = \varphi(key, UID, btn, ctr)$

Assume that $iv = \varphi(btn, ctr)$

Assume linear: $\varphi(btn, ctr) = \alpha.btn + \beta.ctr + \gamma$

Set $ctr = 0$ (4 of our 2^{12} rolling codes since *ctr* is 10 bits long)
$\Rightarrow \varphi(btn, 0) = \alpha.btn + \gamma$
Bruteforce α and γ

We have hits for $btn = 2$, $\alpha = 1$, $\gamma \in \{\texttt{0x4000}, \texttt{0x8000}, \texttt{0xc000}, \texttt{0x10000}\}$
$\rightarrow$ Success \o/

Finding the IV

$$\varphi(btn, ctr) = btn + \beta.ctr + \gamma, \gamma \in \{\texttt{0x4000}, \texttt{0x8000}, \texttt{0xc000}, \texttt{0x10000}\}$$

Now use arbitrary *ctr*, bruteforcing β
Gives $\beta = \texttt{0x10}$

Why $\gamma \in \{\texttt{0x4000}, \texttt{0x8000}, \texttt{0xc000}, \texttt{0x10000}\}$?
$\rightarrow$ *ctr* is in fact more than 10 bits!

$$\varphi(btn, ctr) = btn + \texttt{0x10}.ctr$$

$$\varphi(btn, ctr) = ctr || btn$$

$\rightarrow$ We know everything!

Hitag2 scheme

$$< start||UID||btn||\overbrace{ctr\&0\text{x}3\text{ff}}^{10\text{ bits}}||ks||chk >$$
$$ks = \text{Hitag2}(\text{key} = key_{UID}, \text{serial} = UID, \text{iv} = ctr||btn)_{[0:31]}$$
$$chk = \text{xor}(\text{other bytes})$$

In our case: key_{UID} = `0x4f4e4d494b52` (default Hitag2 key)
But this is the shared secret key of that remote

$\rightarrow$ Contrary to VW schemes, good key divesification

Hitag2 scheme: attacks

To clone the remote, we need the *UID*, *ctr* and the key_{UID}

Need 1 rolling code to get the *UID* and *ctr*
Cannot recover key_{UID} (48 bits) from less than 2 rolling codes (32 bits *ks*)

Attack in the paper:
- 4 to 8 rolling codes
- 10min on a laptop

How to get enough rolling codes

Wait enough time eavesdroping...

How to get rolling codes quicker?

$$< start||UID||btn||ctr||ks||chk >$$

Send noise on *chk*, the car ignore that rolling code
$\rightarrow$ Owner presses button again
$\rightarrow$ Or not
Attacker wins in both cases!

Conclusion

Agenda:

1. RKE from scratch
2. The VW systems
3. The Hitag2 system

Responsible disclosure:
- VW Group Dec 2015
- NXP Semiconductors Jan 2016

Poor crypto is bad, poor key management is worse

Weak ciphers still used in new vehicles

Improvement in ciphers over time (VW-x)

This may explain several mysterious theft cases without signs of forced entry

Q&A