J Cryptogr Eng (2016) 6:155–169
DOI 10.1007/s13389-016-0127-4

CrossMark

CHES 2015

Improved cryptanalysis of the DECT standard cipher

Iwen Coisel[1] · **Ignacio Sanchez**[1]

Received: 14 December 2015 / Accepted: 26 February 2016 / Published online: 17 March 2016

Abstract The DECT Standard Cipher (DSC) is a 64-bit key stream cipher used in the Digital Enhanced Cordless Telecommunications (DECT) standard to protect the confidentiality of the communications. In this paper, we present an improved cryptanalysis of DSC that is more effective than the prior best known one introduced by Nohl–Tews–Weinmann (NTW). Our known plaintext attack requires less than 2.8 min of voice communication (2^{14} keystreams) compared to the more than 11 min (2^{16} keystreams) of the NTW attack to retrieve the key with an equivalent success rate. Our attack yields similar improvements using control data encrypted with the first 9 bits of the keystream, instead of voice, reaching a success rate of 50 % by analysing 2^{13} keystreams in comparison with the 2^{15} keystreams required by the NTW attack to achieve equivalent results. We have successfully applied our attack in a controlled environment against actual DECT communications. Furthermore, we have tuned our attack to work more effectively when the knowledge of plaintext is not 100 % accurate. On the basis of these results, we think that the most appropriate measure to secure the privacy of DECT voice communications is the effective roll-out of the DSC-2 cipher in DECT equipment.

Keywords DECT · Privacy · Security · Cryptanalysis · Stream cipher · Cryptography · Encryption · DSC

1 Introduction

Digital Enhanced Cordless Telecommunication (DECT) is an ETSI standard [5] used in cordless telephony, widely deployed worldwide both in residential and enterprise environments.[1] In traditional residential scenarios, the DECT base station (Fixed Part or FP) is directly connected to the analogue telephone line and provides telephony and a battery recharge point for one or more wireless handsets (Portable Parts or PP). This is the scenario that is typically found today, where the DECT cordless phones have replaced a significant number of classic wired phones.

In enterprise environments DECT cordless phones are often integrated into Unified Communication Systems. These systems integrating several types of voice and data communications, such as VoIP and videoconferencing, start to become available in the consumer market. Nowadays it is common to find low-cost DECT cordless phones able to handle both landline and VoIP communications. Despite the massive adoption of mobile telephony, the DECT standard has reinforced its position as one of the main wireless communication protocols in Smart Home ecosystems.

When encryption is not used, the DECT voice communications are vulnerable to remote eavesdropping attacks, as demonstrated by Lucks et al. [7] employing special purpose hardware. More recently, Sanchez et al. [12] demonstrated that eavesdropping of non-encrypted DECT voice communications can be effectively performed using widely available

I. Coisel and I. Sanchez have contributed equally and are presented in alphabetical order.

✉ Iwen Coisel
iwen.coisel@jrc.ec.europa.eu

Ignacio Sanchez
ignacio.sanchez@jrc.ec.europa.eu

[1] European Commission, Joint Research Centre (JRC), Via Enrico Fermi 2749, 21027 Ispra, VA, Italy

[1] According to ETSI the number of DECT devices sold reaches 820 million with a proliferation of 100 million new devices per year.

low-cost Software Defined Radios (SDR). To mitigate these vulnerabilities and to protect the confidentiality of the communications, the DECT standard foresees the usage of a proprietary stream cipher, the DECT Standard Cipher (DSC). Currently, the privacy of the personal voice communications of hundreds of millions of citizens depends on the security of this encryption algorithm.

The DSC algorithm is referenced in the DECT standard but the details of its design and implementation were only made available under non-disclosure agreements. In [11], Nohl et al. reverse engineered DSC from a hardware implementation and published its internal details along with a reference software implementation. The internal details of the DECT Standard Authentication Algorithm (DSAA), used in DECT for authentication purposes, were also available under non-disclosure agreements only. The details of the DSAA became known to the academic community when Lucks et al. [7] published a cryptanalysis of the cipher, reverse engineering it from a hardware implementation.

Published cryptanalysis of the DSC cipher is quite limited, especially when compared to other encryption algorithms such as A5/1 used in mobile telephony. Nohl et al. [11] proposed the first cryptanalysis of the algorithm, the Nohl–Tews–Weinmann (NTW) attack, capable of recovering the DSC key after analysing large quantities of encrypted traffic. Molter et al. [10] were able to take advantage of weak random number generators present in some DECT phones to brute-force the long-term key and decrypt phone calls. In [9] McHardy et al. demonstrated an active replay attack against DSC able to decrypt a recorded call by interactively recovering the keystreams used to encrypt it.

In [1] we presented an attack able to retrieve the long-term key of a DECT system and derive subsequent session keys to decrypt voice communications. While this attack is efficient (approx. 2^{10} operations), it requires that an attacker eavesdrops the key establishment protocol that takes place between the handset and the base station as part of the pairing process of the DECT device.

In [3] we demonstrated that the long-term key can also be retrieved by physically attacking the base station and tampering with the memory chip. In this work we also highlighted the lack of perfect forward secrecy in DECT phones. In possession of the long-term key an attacker can decrypt any previous communication that might have been recorded (up to the previous long-term key establishment).

The present article is an extension of the improved cryptanalysis of the DSC cipher that we have presented in [2], which requires substantially less plaintext material than the NTW attack to be equally effective, making it more viable in practical scenarios. In this paper, we provide additional information and we describe an enhancement of our attack able to provide better experimental results in scenarios where the knowledge of the plaintext is not 100 % accurate.

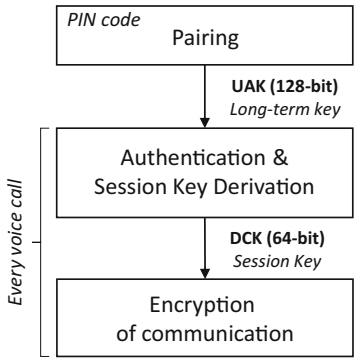


Fig. 1 Pairing, authentication and session key derivation processes in DECT

The paper is structured as follows: We start by introducing the DECT standard and the related underlying cryptographic mechanisms in Sect. 2 followed by a detailed description of the DECT Stream Cipher in Sect. 3. In Sect. 4, we describe the techniques and tools we have developed to support the experimental part of our research. In Sect. 5 we describe the Nohl–Tews–Weinmann attack and its results. Our attack against DSC is introduced in Sect. 6 followed by a description of its working implementation in Sect. 7. In Sect. 8 we present and analyse the results obtained by our approach. Finally, we present the conclusions and we outline future research lines on the topic in Sect. 9.

2 Overview of the cryptographic mechanisms defined in the DECT standard

The DECT standard [6] includes a set of protocols and algorithms designed to secure the communications. Figure 1 depicts the three main security processes in DECT and the relationship between them.

The DECT long-term key derivation protocol that we will refer to as the pairing protocol, is the process defined in the DECT Generic Access Profile (GAP) standard that allows an FP and a PP to agree on a 128-bit long-term key called User Authentication Key (UAK). This protocol takes place when a new PP is being registered into an FP and it is a key element of the GAP profile to ensure interoperability between manufacturers. The UAK value is permanently stored by FP and PP becoming a shared secret that will be used by subsequent authentication and session key derivation processes. Often, when FP and PP are purchased as part of the same set, the devices come already paired from the manufacturer sharing the same UAK value. The pairing process is described in detail in Sect. 2.1.

The session key derivation process is used to determine a 64-bit random session key, known as the Derived Cipher Key (DCK), that will be used to encrypt the communication. Alternatively to this process, the manufacturer could decide

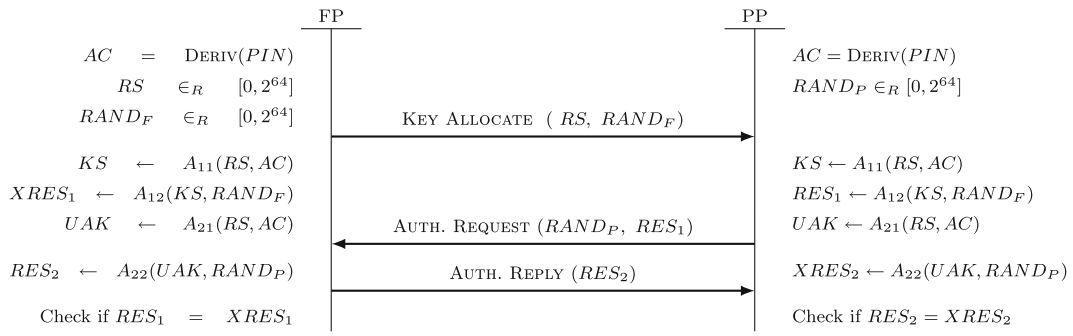


Fig. 2 Key allocation protocol

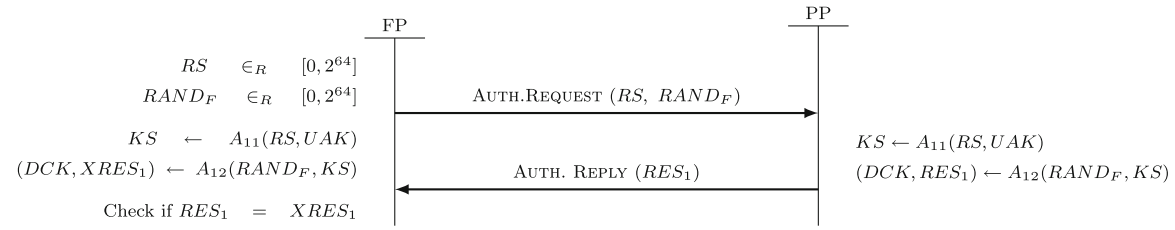


Fig. 3 Authentication of a DECT PP

to use a 64-bit Static Cipher Key (SCK) for encryption purposes, set during the manufacturing process, at the expense of breaking the interoperability of the solution.

2.1 UAK pairing protocol

The main building block of the pairing protocol is the DSAA algorithm that takes as input a key of 128 bits and a second value of 64 bits to output a block of 128 bits. Four different wrappers are defined in the standard around DSAA, denoted as A_{11}, A_{12}, A_{21}, and A_{22}. The difference between them lies in the specific processing made over the output, as defined below.

A_{11}: directly outputs the 128 bits without any post-processing;
A_{12}: discards the first 32 bits of the output, and the remaining bits are split in one block of 64 bits and one of 32 bits;
A_{21}: the output is XORed with a fixed value identical for all DECT equipment;
A_{22}: discards the first 96 bits and outputs the remaining 32 bits as a single output.

The pairing process is depicted in Fig. 2. The UAK is derived by both peers from the several nonces exchanged and an Authentication Code (AC). The latter is determined from a 4-digit PIN code that is permanently stored in the FP and manually inserted by the operator in the PP to initiate the process.

At the end of the procedure the FP checks that the $XRES_1$ value it calculated is equal to the RES_1 value received from PP. The PP executes a similar check over the $XRES_2$ and RES_2 values. If the checks succeeded the new UAK calculated by both peers is accepted and stored permanently.

2.2 Authentication and session key derivation

The PP Authentication procedure is described in Fig. 3. The output of the A_{12} algorithm contains the RES_1 value transmitted to the FP that uses it to authenticate PP comparing it with its own calculation.

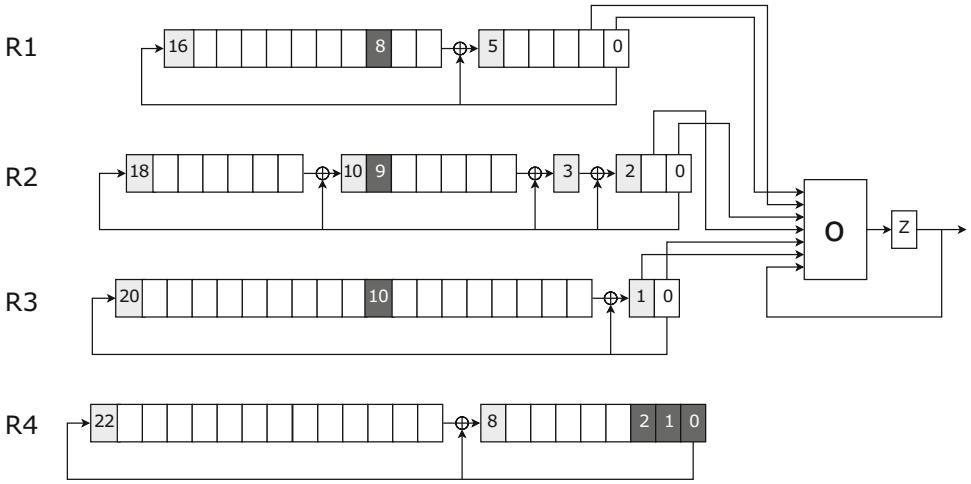
Should the communication be encrypted using a DCK, the second output of A_{12} will be used as the 64-bit session key for the DSC cipher to encrypt the communication.

3 The DECT standard cipher

The DECT Standard Cipher (DSC) is a proprietary 64-bit key stream cipher designed as part of the DECT ETSI standard [5]. DSC is based on four irregularly clocked Linear Feedback Shift Registers (LFSRs) R_1, R_2, R_3, R_4 in Galois configuration and a non-linear output combiner O with memory z. The cipher is initialized with a 35-bit long Initialization Vector and a 64-bit key, respectively, denoted IV and KEY.

Each pair of IV and KEY is used to produce 720 bits of keystream that are split in two keyStream segments of 360 bits each. The first segment is used to encrypt the DECT frames sent by the FP, whereas the second one is used to encrypt the frames sent by the PP. In each case, the first 40 bits are used to encrypt the C-Channel data (that contains

Fig. 4 DSC stream cipher



control data), whilst the rest of the bits are used to encrypt the B-Field (digitally encoded voice). More details can be found in the corresponding DECT ETSI standard [5]. The DSC is depicted in Fig. 4 and formally described in the following subsections.

3.1 The DSC internal configuration

The internal state of the DSC stream cipher is composed of 81 bits spread amongst the four LFSRs and the memory bit of the output combiner. The three first registers, R1, R2 and R3, are used to provide input to the output combiner, as will be described in detail in Sect. 3.2. The last register, R4, is used exclusively to control the clocking of the other registers after each round of the DSC. The four LFSR, depicted in Fig. 4, are formally described below together with their respective feedback polynomials:

- Registry R1, 17 bits: $x^{17} + x^6 + 1$
- Registry R2, 19 bits: $x^{19} + x^{11} + x^4 + x^3 + 1$
- Registry R3, 21 bits: $x^{21} + x + 1$
- Registry R4, 23 bits: $x^{23} + x^9 + 1$

The initialization of the internal state, also called key loading procedure, is executed by introducing bit by bit the 64 bits of the IV (the 35-bit IV padded to 64 bits with 0s) concatenated with the 64-bit key, XORing them with the most significant bit of each register. After each insertion, the four registers are clocked a single time (no irregular clockings are performed in this step).

After the key loading procedure is completed, 40 "empty" rounds are executed during which the generated keystream bits are discarded. Once the initialization procedure is completed, DSC will start producing keystream bits. Each round i, starting by round 0, produces a keystream bit denoted as z_i. Keystream bit z_i is output at the end of the round i. After each round, including the initialization rounds, the R4 register is clocked three times, whereas the three main registers, R1, R2 and R3, are either clocked two or three times depending of the values of specific bits (displayed with a dark grey background in Fig. 4) of the three other registers. The number of times each register is clocked in a specific round, denoted irr_cl_i, is determined as follows, where $x_{i,j}$ denotes the j-th less significant bit of the register i:

$$irr\_cl_1 = 2 + (x_{4,0} \oplus x_{2,9} \oplus x_{3,10})$$
$$irr\_cl_2 = 2 + (x_{4,1} \oplus x_{1,8} \oplus x_{3,10})$$
$$irr\_cl_3 = 2 + (x_{4,2} \oplus x_{1,8} \oplus x_{2,9})$$

3.2 The output combiner

The output combiner O is a cubic function taking as inputs the two least significant bits of the three main registers, R1, R2 and R3, and the bit from the memory slot, denoted z in Fig. 4, corresponding to the previous bit of the keystream. In every round after the key loading procedure, the bit stored in the memory slot will be output as a keystream bit and the new bit produced by the output combiner will replace it in the memory slot.

The 6 bits from the three main registers taken as input by the output combiner constitute what is called in the following the *status* of the DSC, which is specific to a round number and it depends on both KEY and the IV.

The output combiner is formally defined as follows, where $S = (x_{1,0}, x_{1,1}, x_{2,0}, x_{2,1}, x_{3,0}, x_{3,1})$ is the status of the DSC for the current round:

$$\begin{aligned} O(S, z) = {} & x_{1,1}x_{1,0}z \oplus x_{2,0}x_{1,1}x_{1,0} \oplus x_{1,1}z \oplus x_{2,1}x_{1,0}z \\ & \oplus x_{2,1}x_{2,0}x_{1,0} \oplus x_{3,0}x_{1,0}z \oplus x_{3,1} \oplus x_{3,1}z \\ & \oplus x_{1,1}x_{1,0} \oplus x_{3,0}x_{2,0}x_{1,0} \oplus x_{2,0}x_{1,1} \oplus x_{3,1}x_{1,0} \\ & \oplus x_{2,1} \oplus x_{3,0}z \end{aligned}$$

The main purpose of the function is to break the linearity of the three main registers. Over the 128 possible combination of inputs (64 possible statuses plus 2 possible values of the memory slot), half of them output 0 whilst the others output 1.

At first sight the function seems to be balanced since any modification of a given status would modify in average the output bit in 50 % of the cases. However, we have noticed that the output bit remains the same in 56.25 % of the cases when only the bits of a single register are modified. When bits of at least two registers are modified, the outputs are again balanced. Furthermore, we have noticed that the probability that the output bit remains the same is dependent on the input. We will not elaborate further on this last fact since we do not use it in our attack.

3.3 Notations used in the rest of the paper

In order to facilitate the readability of the paper, we introduce some notations that will be used in the following sections.

$S_{sc}(\text{KEY}, \text{IV})$, for $sc = (c_1, c_2, c_3)$, is the status of the DSC, initialized with KEY and IV, when the three main registers are, respectively, clocked c_1, c_2, and c_3 times. $S_l(\text{KEY}, \text{IV})$, is the status of the DSC, initialized with KEY and IV, that has been used to produce the bit output at the end of the round l. When it does not bring confusion, we do not mention KEY and IV.

tc_i denotes an hypothesis about the number of clocks of the i register, while $c_{i,l}$ denotes the actual number of clocks of the i register at the round l.

We abusively call the couple (IV, KS), composed by a keystream KS and the associated initial vector IV, a sample of plaintext (or wlog plaintext), as the "real" plaintext can be recovered from the ciphertext and the keystream. $\mathscr{P} = \{(IV, KS)\}$ denotes the set of available samples of plaintext.

We extend the XOR operator and define the XOR operation of two statuses where the bits of the statuses are mutually XORed together.

4 Tools and techniques developed

The cryptanalysis of the DSC made by Nohl et al. [9] and the one described in this paper are both known plaintext attacks. As a consequence, the keystreams produced by the targeted key must be known. While it is feasible to simulate such data, it is not easy to recover it from a real DECT communications. In order to do so we had to develop new tools that enabled us to intercept and decrypt DECT communications allowing us to collect actual keystreams produced by the DSC hardware implementations. This section describes the several tools and techniques [1,3,12] we have developed to support the experimental part of the article.

4.1 Passive eavesdropping of non-encrypted communications

In the cryptanalysis presented in this paper we have used data taken from actual DECT communications. These data have been acquired using the Software-defined Radio DECT sniffer developed internally and described in [12]. The DECT sniffer was used in combination with a USRP B200 board from Ettus Research and a standard VERT-900 antenna. The implementation is capable of eavesdropping actual DECT communications saving the sniffed DECT packets in a PCAP file.

The acquisition of the data was done by placing both an FP and a PP (the target) together with the DECT SDR sniffer inside an anechoic chamber. The DECT system was connected to a land-line to simulate an actual phone call. We have collected data from a wide variety of DECT phones from several manufacturers.

The analysis of the captured communications revealed that some phones do not encrypt at all their communications giving to an eavesdropper direct access to the transmitted voice. Other phones encrypt only the B-Field data but not the C-Channel communications. Only recordings of fully encrypted communications (both C-Channel and B-Field) were used to test our DSC cryptanalysis attack.

4.2 The pairing attack

As a first step towards being able to test with actual data our DSC cryptanalysis, we set up a technique to extract the set of (keystream, IV) and the corresponding 64-bit key used by the DECT target system to encrypt the communication. In order to do so, we eavesdrop the pairing process of a given pair of FP and PP to follow the protocol and derive the negotiated UAK key. Having the UAK key allowed us to derive subsequent DCK by following the session key derivation protocol. Implementing this technique we noticed that the DECT pairing protocol is not secure. In the following, we briefly introduce our attack that is able to determine the UAK by observing the values exchanged in cleartext during the protocol. An interested reader should refer to the full article [1] for more details.

An eavesdropper monitoring the radio link can collect all values involved in the computation of RES_1 and UAK with the exception of AC. The latter is deterministically computed from the PIN, which is unknown to an attacker since it is not exchanged during the protocol. However, the PIN is the only unknown in the computation of RES_1, as described below:

$$RES_1 = A_{12}(A_{11}(RS, \text{DERIV}(PIN)), RAND_F)$$

Given that RES1 can be collected by an attacker, an exhaustive search can be conducted trying all the possible

PIN codes (typically 10^4 values) to find the one that verifies the previously described equation.

Once in possession of the right PIN code, the attacker can derive UAK, which is equal to $A_{21}(RS, AC)$. At this point, the attacker is able derive any future DSC session keys using UAK to follow the execution of PP authentication protocols and decrypt with them any communication.

4.3 Physical extraction of the key

In order to test our cryptanalysis attack with actual encrypted communications generated by phones that had not been previously paired (e.g. FP and PP that were purchased as part of the same set), we developed a technique to recover the UAK directly from the memory of the FP. The full details about this process are provided in an article written together with David Shaw [3]. This technique has proven to be a powerful physical attack that we successfully tested on a total of seven different models of DECT cordless phones from four different manufacturers, four of them manufactured in 2014.

The phones analysed in this work were equipped with a 24cxxx EEPROM chip that operates with the I2C protocol. Since then, we have analysed two additional phones, one of them equipped with a chip operating with the SPI protocol. In all cases the EEPROM memory is used by the device to store several parameters related to the configuration of the DECT network, such as the RFPI or, more importantly in the case of this study, the UAK and IPUI values of each PP registered in the FP.

In order to extract the contents of the EEPROM memory, we used an open source device known as the *Bus Pirate* to connect to the I2C bus of the main circuit board. This device is a multi-purpose low-cost hardware tool that is specifically designed to analyse, operate, and assist in the usage and reverse engineering of electronic boards and integrated circuits.

The methodology we followed consisted in identifying the addressing mode of the memory chip, scanning the I2C bus (or the SPI one), locating the addresses and dumping the whole contents of the SMD EEPROM memory into a binary file. The specific location of the UAK within the memory map can be discovered by following the PP authentication, which was eavesdropped, with each possible 128-bit value found in the EEPROM to find the one that matches the response of the PP to the challenge.

In the referred article we highlight the lack of perfect-forward secrecy in the DECT protocol. Indeed, using an attack able to determine the UAK such as the one we presented, it is possible for a third party to decrypt any present, future or past communication for as long as no other pairing is made. Our article proposes some measures to mitigate this issue, even if the likelihood that a DECT system is compromised this way is low, given that physical access to the FP is required to extract the cryptomaterial.

5 The Nohl–Tews–Weinmann attack

The Nohl–Tews–Weinmann (NTW) attack is a known-plaintext attack which, given a set of plaintext $\mathscr{P}$, is able to recover the 64-bit DSC key faster than an exhaustive search over the 2^{64} possible keys.

The first phase of the attack determines a set of affine linear equations that specify relations about the key bits. In the second phase the remaining bits of the key are brute-forced to obtain the 64-bit key.

Due to the linearity of DSC, each bit of the registers can be defined as a linear combination of bits of the key and the IV, for a given number of clocks. The goal of the attack is to guess enough statuses of the DSC for several sets of clocks to obtain a sufficient number of equations to be able to perform an exhaustive search over the remaining bits of the key.

5.1 Guessing a single status

In this section, we describe how the NTW attack manages to retrieve a single status of the DSC. In order to do so, we first must explain how the clocks c_i of the three main registers can be guessed for a given round. This part of the attack takes advantage of the linearity of the DSC stream cipher, more precisely of the fact that the following equality holds for any triplet of clocks $sc = (c_1, c_2, c_3)$:

$$S_{sc}(\text{KEY}, \text{IV}) = S_{sc}(\text{KEY}, 0) \oplus S_{sc}(0, \text{IV})$$

The attack is also based on the following fact: Assuming that a register is clocked twice with a probability of 50 %, the clock c_i for the round l is distributed according to a shifted binomial distribution with mode $2.5l + 100$. Based on the distribution, the most probable triplet(s) of clocks can be selected. As an example, the most probable triplets for the first round are those where each clock is either 102 or 103.

Furthermore, when the registers are clocked accordingly to the values in sc at the end of the round l the equality $S_{sc}(\text{KEY}, \text{IV}) = S_l(\text{KEY}, \text{IV})$ holds. In addition, the generated status and the bits of the keystream always verify the equation $O(S_{sc}(\text{KEY}, \text{IV}), z_{l-1}) = z_l$. Such equation is denoted $eqn(sc, l)$ in the rest of the paper.

The key and $S_{sc}(\text{KEY}, \text{IV})$ are unknown, whereas the IVs and the bits of the keystream are assumed to be known as it is a known-plaintext attack. Therefore, it is possible to test the equation $eqn(sc, l)$ with the 64 possible statuses to identify the subset $\mathcal{S}$ of them that verify it. The correct status will necessarily belong to this subset of 32 candidates. The influence of the IV in the status can be computed as $\tilde{s} =$

$S_{sc}(0, \text{IV})$. Assuming that sc is the correct triplet of clocks, $S_{sc}(\text{KEY}, 0)$ will be in the subset $\tilde{\mathcal{S}} = \{\tilde{s} \oplus s^*; \forall\, s^* \in \mathcal{S}\}$. On the contrary, if the triplet sc is not the triplet of clocks of round l, the correct status still has a probability of 50 % to be in this set according to Nohl et al. [11].

Consequently, the correct status has more than 50 % chance of being in this set, whereas any other possible status has just a 50 % probability to be inside. This experiment can be seen as a Bernoulli trial where a success is the presence of the status in the list of candidates. Therefore, if repeated a sufficient number of times, the most frequent status should be the one of $S_{sc}(\text{KEY}, 0)$, given the existence of this bias.

5.2 Determination of more equations

The determination of a single status provides six linear equations that related together bits of the key. However, the reduction of the key space would still be insufficient, as the brute-force of the remaining 58 bits of the key would require too much computational efforts.

In order to determine more equations for the DSC key, the NTW attack extends this principle for each possible combination of clocks in a large range (35 in their article) considering several bits of the keystream (19 in their article). For each triplet of clocks, a frequency table is generated to store the "score" of each potential candidate status.

Once all samples have been processed, the value of a bit of a given register for a given clock is estimated according to all the frequency tables where this bit is involved in. By doing so, 108 bits of the DSC status are derived leading to a total of 108 equations. A subset of these equations is selected according to a certain rank (see [11] for more details) and the solvability of the obtained system. When enough equations have been selected (e.g. around 30) the remaining bits are brute-forced.

5.3 Results of the Nohl–Tews–Weinmann attack

Nohl et al. have conducted their known-plaintext attack against both simulated C-Channel and B-Field data considering different quantities of plaintext, evaluating the probability that the system of equations defined is valid according to the correct key. Their results are summarised in Table 1, for both C-Channel and B-Field, considering the several sizes used for the system of equations (10 to 40) and the different quantities of plaintext analysed. Full details about the results obtained by the NTW attack can be found in the original article [11].

To reach a probability of success of 50 % against the C-Channel to define 30 equations, the attack requires at least 32,768 plaintexts. Against the B-Field, this attack reaches a probability of success of 28 % for 30 equations with approximately 65,536 different samples of plaintext. Slightly better results can be found in the paper of Weiner et al. [13] following an optimisation based on a new key ranking procedure. As an example, the success probability for 32,768 available keystreams and 22 equations goes from 71 to 90 %.

Table 1 Success rate of the C-Channel and B-Field attack

Number of plaintext	C-Channel		
	8192 (%)	16,384 (%)	32,768 (%)
10 Equations	2	30	96
20 Equations	0	2	78
30 Equations	0	1	48
40 Equations	0	0	11
Number of plaintext	B-Field		
	16,384 (%)	32,768 (%)	65,536 (%)
10 Equations	2	30	92
20 Equations	0	2	65
30 Equations	0	0	28
40 Equations	0	0	4

6 A theoretical model of an improved cryptanalysis

Instead of considering separately each bit of the internal status of the DSC as done in the NTW attack, our attack directly processes the entire status for a given range of clocks. By doing so, all the irrelevant candidates due to the equality of some bits and the feedback of the registers are discarded in a preliminary step. Furthermore, the underlying theoretical model used to score the potential candidates has been refined, leading to more accurate results. Before entering into details we summarise below the full process of our attack.

The first stage of the attack aims to retrieve the 6-bit statuses of the DSC for each triplet of clocks of a given range of length len_c. As the statuses of two consecutive clocks share three bits, the final combination of statuses is $3(len_c + 1)$-bit long. A frequency table containing all these possible combinations, called candidates, is generated to store their *score*. All the equations $eqn(sc, l)$ relevant for this range of clocks are evaluated for each possible candidate. The candidates that verify a given equation increase their score by a value, called weight, specific to this particular equation. This weight is computed according to the probability that the correct status belongs to the specific subset of candidates that satisfy the corresponding equation for a random pair of IV and keystream.

Once all the plaintext samples are processed, all the candidates are ordered according to the score they have obtained. $3(len_c + 1)$ linear equations linking together the bits of the DSC key can be derived from each candidate. Starting with the first candidate, the remaining bits of the key (i.e.

$64 - 3(len_c + 1)$ bits) are then brute-forced in the last step of the attack.

6.1 Computation of the weights

In a preliminary phase, the weight of each possible equation $eqn(sc, l)$ for the selected range of clocks is calculated. Only the equations with non-null weights will be evaluated during the attack. This preliminary step is also performed in the NTW attack. However, we have noticed experimentally that their values were not accurate. Based on this observation, we have refined the theoretical model by including the non-homogeneous behaviour of the output combiner.

The weight of an equation is based on the probability that the output combiner outputs the same bit when it takes as input either $S_{(tc_1,tc_2,tc_3)}$ or S_l. As a reminder, the equation $eqn((tc_1, tc_2, tc_3), l)$ is said verified if

$$O(S_l, z_{l-1}) = O\big(S_{(tc_1,tc_2,tc_3)}, z_{l-1}\big).$$

Obviously, if the registers are ,respectively, clocked tc_1, tc_2, and tc_3 times in the round l, then this equation is verified with a 100 % probability and the correct status will necessarily be in the subset of candidates. In the following, we assume that the probability that a register is clocked only two times at the end of a round is 50 %. We also assume that the clocking decision of each register is independent from the other registers. The probability that a single register i is clocked exactly tc_i times after the round l is

$$Pr\big[c_{i,l} = tc_i\big] = \binom{40 + l}{tc_i - (80 + 2l)} 2^{-(40+l)}.$$

The probability, denoted p_1, that the three main registers are, respectively, clocked tc_1, tc_2, and tc_3 times after the round l, is defined as follows. For the sake of clarity we omit (tc_1, tc_2, tc_3) and l in the notation of the probabilities:

$$p_1 = \prod_{i=1}^{3} Pr\big[c_{i,l} = tc_i\big].$$

When there is at least one clock difference between (tc_1, tc_2, tc_3) and the actual set of clocks, one could expect that the correct status is in the subset of candidates with a probability of 50 %, as stated in the article [11]. However, due to the particular behaviour of the output combiner, previously described in Sect. 3.2, the probability that the equation remains verified is not exactly 50 % when the clocks differ. Indeed, the equation is verified with a probability of 56.25 % if two out of the three targeted clocks are correct. The global probability of success can, therefore, be refined.

For clarity we introduce two intermediate probabilities, the probability p_2 that only one register is not clocked the targeted number of times and the probability p_3 that at least two registers are not clocked the targeted number of times.

$$p_2 = \sum_{i=1}^{3} (1 - Pr[c_{i,l} = tc_i]) \prod_{\substack{j \neq i; \\ j=1}}^{3} Pr[c_{j,l} = tc_j]$$

$$p_3 = \sum_{i=1}^{3} \left[Pr[c_{i,l} = tc_i] \prod_{\substack{j \neq i; \\ j=1}}^{3} (1 - Pr[c_{j,l} = tc_j]) \right] + \prod_{k=1}^{3} (1 - Pr[c_{k,l} = tc_k])$$

The probability that a given equation is verified can now be expressed as

$$Pr[eqn(sc, l)] = p_1 + 0.5625 * p_2 + 0.5 * p_3$$

As an example, the equation $((102, 102, 102), 1)$ corresponding to the production of the keystream bit z_1, is verified in 50.338 % of the cases. Following the approach of [8], the weight $w(eqn(sc, l))$ associated with an equation is computed as the logarithmic likelihood of the probability, namely

$$w(eqn(sc, l)) = log\left(\frac{Pr[eqn]}{1 - Pr[eqn]}\right).$$

The weight of all possible equations according to a given range of clocks and a given range of keystream bits are precomputed using these results.

6.2 Determination of the best candidates

The precomputed weights are now used in the first step of the attack to guess the most probable statuses of $S_{sc}(\text{KEY}, 0)$ for the largest possible range of clocks $\mathcal{R}$ and a given set of bits of the keystream $\mathcal{K}$. $\mathcal{T}$ denotes the table that contains the $2^{3(len_c+1)}$ possible combinations of bits for the given range of clocks. The score of each candidate is equal to the addition of all the weights of the equations verified by this particular candidate.

For all triplet of clocks $sc \in \mathcal{R}^3$ and all rounds $l \in \mathcal{K}$, the equations $eqn(sc, l)$ are evaluated for all the possible candidates in $\mathcal{T}$. The statuses that verify a given equation are those potentially corresponding to $\tilde{s} = S_{sc}(\text{KEY}, IV)$. The influence of the IV in the status can be removed thanks to the DSC linearity, as $s^* = S_{sc}(0, IV)$ can be computed. Therefore, the candidates that shall receive the weight of an equation are $s = \tilde{s} \oplus s^*$. As in the NTW attack, the more plaintexts are used, the higher the chances are that the correct status belongs to the subset of the ones with the higher scores.

6.3 Exhaustive search over the remaining bits

A system of $3(len_c + 1)$ linear equations with 64 unknowns can be defined from each candidate in the table $\mathscr{T}$. All the equations are independent as long as the length of the range of clock used stays below 17 since it is the size of the smaller register. The DSC 64-bit key will be a solution of the system assuming the right candidate has been selected. $3(len_c + 1)$ bits of the key are determined by these equations whilst the remaining ones need to be brute-forced. We apply a Gaussian reduction to the selected system to ease this brute-force step.

In order to carry out the last step, we have developed a CPU SIMD-based optimised implementation that loads the system of equations and performs the exhaustive search over the remaining bits of the key. For each possible combination of key bits to be explored it determines the remaining ones using the system of n equations, reducing the search to 2^{64-n} possible keys. A portion of keystream longer than $64 - n$ bits is compared against the output of DSC executed using the candidate key and the IV that belongs to that specific keystream. The process will explore the entire key space until a match is found. Our implementation reaches approximately a rate of 500 million keys/sec in a Core i7 (AVX) workstation.

7 Improved implementation of the cryptanalysis

The attack described in the previous section is impractical given the computational resources that would be required to explore all possible combinations of clocks in a large enough range to derive a sufficient number of equations to make viable the bruteforce step.

In this section we present a probability of success vs time trade-off to hasten considerably the execution time whilst maintaining a good success rate.

7.1 Efficiency consideration

The parameter that ultimately influences the most the efficiency of the attack is the length len_c of the clock range. Increasing it not only speeds the final brute-force step, but also increases the number of equations n_{eq} to be evaluated as well as the size of the candidate table $\mathscr{T}$, both cubic functions in len_c as formally defined below:

$$n_{eq} = len_c^3.len_k$$
$$|\mathscr{T}| = 2^{3(len_c+1)}$$

To reduce the workload, we split the range of clocks in sub-ranges and apply our attack to them. Combining the best candidates of each of the sub-ranges frequency table allows the determination of the same amount of status bits as in the full range. Unfortunately, this technique also reduces the probability of success due to the loss of contribution from the equations related to clocks that overlap these sub-ranges. Adding some redundancies in the definition of the sub-ranges can be a compromise between computational resources required to carry out the attack and probability of success. However, the small gain in the probability of success that we have observed is not worth compared to the loss of efficiency introduced by the overlaps. Indeed, following this approach more sub-ranges would be required to obtain the same amount of linear equations.

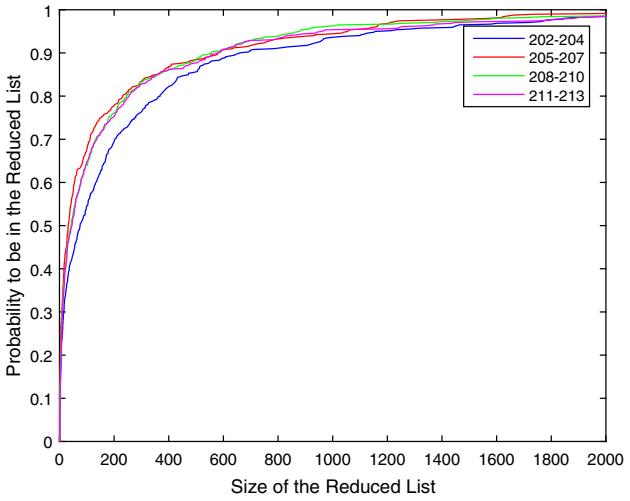
7.2 A time-accuracy trade-off

We have considered a time-accuracy trade-off using several sub-ranges of reasonable lengths (typically 3 or 4 clocks each) without any overlapping between them, over which we apply the attack previously defined in Sect. 6. A frequency table is generated for each of these sub-ranges. Then a joint table of the two first sub-ranges is created and populated with the cartesian product of the N_T most promising candidates from each sub-table. The score associated with these new candidates is initially set to the sum of the scores taken from the two original candidate tables. The attack described in Sect. 6.2 is reapplied to this new subset considering the equations that only involve set of clocks that spread across these two sub-ranges. Note that the equations already processed in the prior stage are not evaluated a second time. The most N_R promising candidates from the merged frequency table are again extracted and combined with the ones of the next sub-range and so on until all the sub-ranges have been processed.

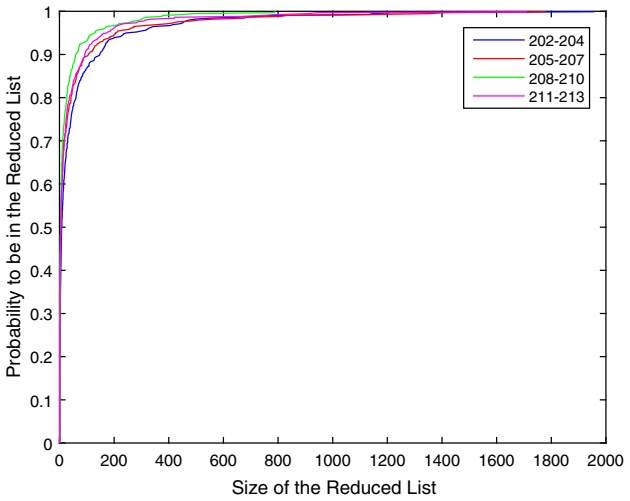
N_T and N_R are critical parameters as they allow to hasten the experiment to the detriment of the success probability. Small values decrease the number of candidates that will be evaluated against the equations for a wider range of clocks. At the same time, the chances that the correct candidate belongs to the reduced list and that the attack is successful are decreased. Indeed, if the algorithm fails to capture the right status in a reduced list of a sub-range, it is certain that it will not be present in the final joint table.

To find out the most convenient thresholds for the selection of the most promising candidates, we have experimentally determined the probability that the correct candidate is in the reduced list. We have conducted these experiments for different quantities of available plaintext, for several sub-ranges against both the C-Channel and the B-Field. Figure 5 presents the probabilities that the correct candidate is in the top N_T of the corresponding sub-ranges for 16,384 and 32,768 available plaintexts when attacking the B-Field.

As can be observed in these figures, the increase of available plaintexts increases the chances that the correct candidate is in the reduced table. However, it should be kept in



(a) 16384 plaintexts



(b) 32768 plaintexts

Fig. 5 Probability that the correct candidate is in the top N_T for several sub-ranges attacking the B-Field

mind that it also increases the time required for the evaluation of all the equations. As a consequence, the size of the reduced table should be selected taking into consideration the quantity of available plaintext, the desired probability of success and the available time to conduct the attack. An interesting fact is that, even if the right candidate was not in a good position in one or more tables during the experiments (see Sect. 8 for more details on these experiments), it can still be the first one in the final one, due to the fact the wider the range, the more equations will contribute to the determination of the most probable candidate.

7.3 Selection of the relevant equations

Following the approach previously described, we divide the full range in four sub-ranges of three clocks each. For each sub-range we only consider the keystream bits for which the associated equations have a relevant weight for this range of clocks. Indeed, the impact of a certain keystream bit on a given sub-range is directly dependent on the shifted binomial distribution of the clocks. As an example, the first bit of the keystream has much more impact on the range [102, 104] than on the range [111, 113], as the distribution of the clocks is centred in 102.5. Therefore, by evaluating in every sub-range only the relevant set of equations we optimise the performance of the attack.

For a given set of clocks and a given bit of keystream, the bias of the corresponding equation is the difference between the probability that such equation is verified and the expected probability of 50 %. Given a range of clocks and a keystream bit, the associated accumulated bias is the sum of the bias of each possible equation from the given range of clocks for the specific bit of the keystream. Figure 6 represents graphically the computed accumulated bias for different keystream bits and sub-ranges of clocks. Based on these results, we have selected the list of pertinent equations to be evaluated in each sub-range.

It can be observed in Fig. 6 that the accumulated bias for a given keystream bit decreases proportionally to the position of the bit considered. For example, the accumulated bias of the bit 41 (first bit of the keystream related to the B-Field) is almost half compared to the first bit of the keystream. This is a direct result of the irregular clocking and the increasing uncertainty about the specific combination of clocks that generated that bit. Therefore, the probability that an equation related to this bit gets verified, decreases as well. That is the reason why more plaintext samples are required to conduct a successful attack using B-Field data instead of C-Channel data.

8 Experimental results of our attack

In order to test experimentally our cryptanalysis attack, we have conducted several experiments, both with simulated and actual data, aimed at validating and benchmarking it. Two scenarios using data originated from the C-Channel and the B-Field have been considered respectively for the ranges of clocks [102, 113] and [202, 213].

For all the experiments, we have followed the approach described in the previous section where the full range is divided in four sub-ranges of three clocks each and the first 9 bits of the corresponding segment of the keystream are used for the attacks.

8.1 Results based on simulated data

In this section we present our experimental results using simulated data. We have generated a total of 200 random DSC

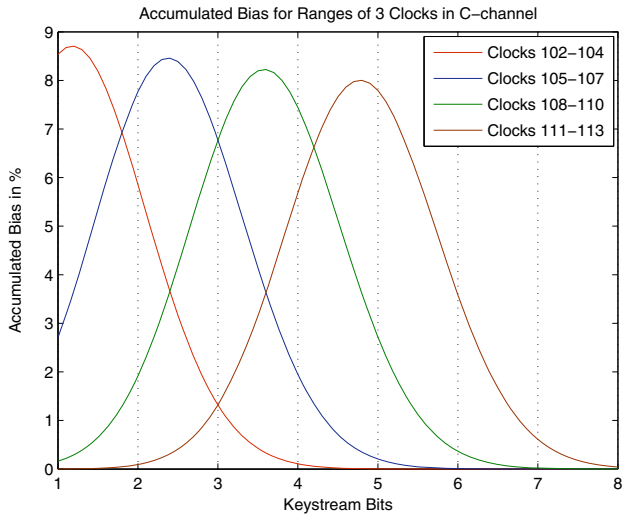


Fig. 6 Accumulated bias for sub-ranges of 3 clocks

keys and for each one we have created several sets of plaintext samples (IV and keystream) of different sizes. For each of the 200 keys, the first IV of the first sample was generated randomly and the subsequent IVs incrementally, mimicking the behaviour of actual DECT devices.

Each set of plaintext represents a recording of an encrypted DECT conversation, where each packet contains the IV in clear and the payload encrypted with a unique keystream. The total amount of plaintext is directly linked to the number of minutes of the encrypted voice call that had to be recorded to obtain them. For the cryptanalysis, the values N_T and N_R were, respectively, set to 200 and 50.

An interesting finding in our experiments is that when the attack was unsuccessful, often only a few bits of the best candidate differed from those of the correct candidate. Most of the time these wrong bits were the ones at the edge of the range (e.g. the bits for 102 and 113 for the range [102, 113]). Therefore, the probability of success of the attack can be increased by discarding these status bits, at the cost of reducing the number of equations and increasing the time required for the final exhaustive search. For example, discarding the two bits at the edge of the candidate reduces the number of linear equations from 39 to 33.

Table 2 Success rate of the C-Channel and B-Field attack for the respective range [102, 113] and [202, 213]

Number of plaintext	C-Channel		
	4096 (%)	8192 (%)	16,384 (%)
9 Equations	35	85	98
21 Equations	16	73	97
33 Equations	6	55	95
39 Equations	2	33	84
Number of plaintext	B-Field		
	8192 (%)	16,384 (%)	32,768 (%)
9 Equations	19	69	94
21 Equations	10	57	90
33 Equations	3	36	82
39 Equations	1	21	66

Table 2 displays the percentage of time the correct status was output in first position by our attack so that our results can be compared on a fair basis with the ones of the NTW attack [11].

Our results can additionally be improved (up to 10 % additional keys retrieved) when the brute-force step sequentially evaluates the next possible candidates output in the final table. For example, while the success probability is about 69 % when using 39 equations attacking the B-Field with 32K plaintexts, it increases to 76 % by considering the final output list. Considering 33 equations, our attack guesses the correct candidate with a probability slightly higher than 50 % when using 8192 plaintexts from the C-Channel. In this case, the final exhaustive search step is able to retrieve the correct key in less than 5 s using our CPU SIMD-based implementation in an i7 multi core (AVX) workstation. To reach a more or less equivalent success rate, the NTW cryptanalysis attack requires at least four times more plaintext material. The success probability can be raised to more than 70 % using only 21 equations at the price of a longer but still reasonable exhaustive search of less than 5 h.

Although at a first sight it seems much more interesting to attack the C-Channel rather than the B-Field, based on the number of requested plaintext, it shall be noted that only a limited amount of DECT packets are typically sent per second (around 5) carrying C-Channel data, in comparison with the 100 that are sent per second carrying B-Field data during

a voice communication. Consequently, 20 times more plaintext is produced in the B-Field which makes the attack more realistic in this scenario in terms of minimum call duration required to gather enough cryptomaterial.

As an example, to reach a 55 % chance to retrieve the key using C-Channel plaintext, 27 min of conversation have to be recorded, whereas the NTW attack would require more than 1 h and 50 min to reach an equivalent success rate. Considering a B-Field attack, our attack achieves a success rate of 36 % using just 163 s of communication when the NTW attack needs more than 11 min to reach equivalent results.

8.2 Results based on actual data

In order to validate our findings in practice we have used several DECT cordless phones, from several manufacturers, that we have previously verified and found to be encrypted following the approach described by Sanchez et al. [12] and briefly introduced in Sect. 4.1.

In a first round, we validated our attack assuming 100 % accuracy in the prediction of the plaintext. In order to do so, we have extracted the UAK of the targeted phones using our pairing attack as briefly described in Sect. 4.2. Thanks to this long-term key we were able to obtain the session key and consequently the plaintext required to test our attack.

Our first attempts to recover the key were performed analysing 5 min of DECT encrypted calls (corresponding to 32K samples) from several handsets of different commercial brands available in the domestic market. In our first round of experiments our attacks had a success rate of over 66 % targeting the encrypted voice. A second round of tests analysing 10 min of calls were 100 % successful.

8.3 Predictability of C-channel plaintext

The C-channel messages transported in the A-field of certain DECT packets are used to exchange control data between FP and PP. For example, C-channel messages are used during the initialisation of a phone call to authenticate the FP and PP and derive a DSC session key (DCK). C-channel is also used to exchange other messages related to the encryption process or to transmit information about the duration of the ongoing phone call. The set of control messages that can be exchanged are defined in the ETSI DECT standard [4].

Once the DCK has been negotiated during the initialisation of the call, FP and PP can agree to encrypt C-Channel data. When the C-Channel is encrypted, an attacker can still exploit the predefined structure of these messages to predict parts of their content. Furthermore, the periodicity and position of a given C-Channel message in a communication can provide additional hints about its content.

To evaluate in practice the feasibility of predicting the plaintext of C-channel messages, we have analysed the communications of different phones from several manufacturers that we know use encryption to protect voice communications.

Many phones turn on the encryption of the C-channel as soon as the DCK is negotiated, typically within the first 2 s of the communication. It is worth noting that during this brief period both voice and C-channel information are transmitted in cleartext. An attacker can analyse the content of the packets involved in the C-channel communication before the encryption is activated to gain knowledge about the plaintext and the communication pattern. Once the encryption is activated for the C-channel, the attacker can use that knowledge to predict the first 9 bits of the plaintext that will be encrypted in the packets involved. Moreover, we noted that many phones send periodically encrypted C-channel messages with predictable content during a phone call. The predictability of this content, which depends on the branch and model of the DECT system that is used, can enable an attacker to collect enough keystream material to carry out a successful attack.

To our surprise, some phones that fully encrypt the voice communication (B-field) do not encrypt the C-channel at all. In these cases, the C-Channel communication cannot be used as a source of plaintext to carry out the DSC cryptanalysis attack, which must rely exclusively on B-field data, much more difficult to predict. Therefore, even though leaving the C-channel communications unencrypted implies that an attacker can eavesdrop the data exchanged, often these data are not particularly relevant or sensitive. As suggested by Nohl et al. in [11], leaving the C-channel communication unencrypted paradoxically contributes to enhance the privacy of the voice communication by removing an easy source of predictable plaintext.

8.4 Partially-known plaintext attack

Our experimental results have shown that our attack can be successfully carried out using B-Field data as source of plaintext. However, B-Field plaintext cannot be easily predicted given that it contains the digitally encoded voice of the two speakers. Given that in DECT the voice of each speaker will be transmitted independently from the other one, we could assume that periods of silence will occur during a standard conversation in both directions. As pointed out by [11], these segments of silence can be a useful source of plaintext with perfect silence producing all ones in the digitally encoded voice.

We have analysed several actual DECT voice communications recorded in a quiet environment, thus containing silence that could occur in realistic conditions, in order to understand up to what extend the plaintext could be predicted. We have found out that depending on the equipment and the back-

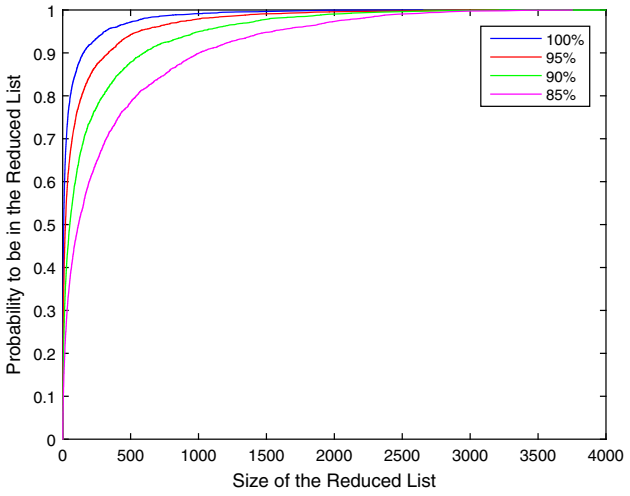


Fig. 7 Probability that the correct candidate belongs to the reduced list

Table 3 Success rate of the B-Field attack for the range [202, 213]

Accuracy of plaintext	32,768 Plaintexts			
	100 %	95 %	90 %	85 %
9 Equations	96	92	71	55
21 Equations	91	78	57	37
33 Equations	85	65	42	21
39 Equations	81	56	28	11
Accuracy of plaintext	65,536 Plaintexts			
	100 %	95 %	90 %	85 %
9 Equations	100	100	100	92
21 Equations	100	100	96	81
33 Equations	99	98	87	70
39 Equations	99	94	85	63

ground noise, the accuracy of the recovered keystream, under the assumption that the plaintext was pure silence, ranges from 85 to 95 %.

When one of the bits of the keystream derived from the prediction of the plaintext is wrong, the probability that the correct status belongs to the candidate list drops to 50 % for all the equations where this bit is involved. We could consider that the vote cast by those equations is randomly distributed among the possible candidates. This fact reduces the overall probability that the candidate belongs to the reduced list of a sub-ranges.

To verify this statement, we have experimentally determined the probability that the correct candidate is included in the reduced list of the sub-range [202, 204], for several degrees of inaccuracy in the prediction of the keystream. The results of the experiment are shown in Fig. 7. This observed loss of accuracy can be compensated by increasing the size of the reduced list at the cost of the overall efficiency. Analysing more plaintexts can also contribute to compensate this loss. To measure the impact in the probability of success, we have benchmarked our attack using simulated B-Field data assuming the knowledge of 32,768 and 65,536 plaintexts with several degrees of accuracy. The results presented in Table 3 show the success probabilities to retrieve the session key among the final output list of candidates.

We have also used actual DECT encrypted voice communications to validate that our attack works in practice even when the plaintext is only partially known. When mostly silence is transmitted, our prediction of plaintext is accurate at approximately 90 %. All our attempts using 65K plaintexts defining 39 linear relations amongst the key bits were successful. The success rate of the tests we have conducted using 32K plaintexts are consistent with the results of Table 3.

8.5 Taking advantage of the codec behaviour

We have made an interesting discovery analysing data decrypted from actual DECT voice communications. The distribution of the zeros, bits differing from digitally encoded pure silence, is not uniform over the 9 bits of the keystream that are required for our attack. We have noticed unexpected patterns in groups of 4 bits, probably derived from the way the G726 codec encodes the voice. The Least Significant Bit (LSB) in each of these 4-bit group has a higher probability to be a zero.

We have recorded several hours of pure silence in an anechoic chamber to evaluate the observed patterns. Around 90.7 % in average of the bits of the plaintext transmitted by the FP were ones. The two LSB in the 4-bit groups, more precisely the fourth and the eighth bits of the byte, had a probability of around 80 % to be zeros.

The fact that some bits of the keystream are more likely to be zeros than others should be taken into account during the computation of the weight. We have thus introduced such possibility in the probability computation. The updated probabilities are the following where $err[l]$ denotes the probability that the l-th bit of the keystream is wrong:

$$\begin{aligned} p_{1\_new} &= p_1 * (err[l-1] * err[l] \\ &\quad + 0.5 * (1 - err[l-1])) \\ p_{2\_new} &= p_2 * (0.5625 * err[l-1] * err[l] \\ &\quad + 0.5 * (1 - err[l-1]) \\ &\quad + 0.4375 * err[l-1] * (1 - err[l])) \end{aligned}$$

Note that when at least two registers are not clocked the targeted number of times the probability that the equation is verified remain of 50 % even if the prediction about the

Table 4 Success rate of the attack using silence for the range [202, 213] using 32K plaintext

	Old model (%)	New model (%)
9 Equations	71	79
21 Equations	61	67
33 Equations	43	50
39 Equations	39	43

plaintext is wrong. The new probability that a given equation is verified is now:

$$Pr[eqn(sc, l)] = p_{1\_new} + p_{2\_new} + 0.5 * p_3$$

We have conducted a series of tests to compare the success rate of the old and the new probability models for both 32K and 65K of plaintext of B-Field communication. In the latter case, the success rates were not significantly different whereas in the first case, the updated model performed better than the old one. The results are displayed in Table 4.

9 Conclusions and future developments

In this paper we have presented an improved cryptanalysis attack against the DECT Standard Cipher, leveraging the clock guessing approach introduced by the NTW attack. Compared to it, our approach offers higher success probability, requiring four times less amount of keystream material.

Our attack is able to retrieve the key with a probability of success of 55 % using 33 linear equations and 2^{13} keystreams of C-channel following an exhaustive search over 2^{31} keys. In comparison to it, the NTW attack requires 2^{15} keystreams to reach a probability of success of 48 % after exploring 2^{34} keys using 30 equations. Attacking encrypted voice our attack provides consistent results requiring 2.8 min of voice communication (corresponding to 2^{14} keystreams of B-field) to reach a probability of success of 36 % using 33 equations and searching 2^{31} keys. In a similar scenario the NTW requires 10.9 min of communication (corresponding to 2^{16} keystreams of B-field) to reach 28 % of probability of success with 30 equations and an exhaustive search over 2^{34} keys.

We have also successfully applied our attack to actual voice communications eavesdropped from commercial DECT phones that use encryption. Furthermore, we have explored experimentally the use of silence in the voice communication as source of predictable plaintext and demonstrated that our attack can still work with a non-perfect prediction of the plaintext, albeit with a lower probability of success. In this scenario we have been able to improve our attack, taking advantage of the patterns we have observed in the distribution of probabilities over the bits of the keystream to reduce the influence of this loss of accuracy in the prediction of the plaintext, improving the success rate of our attack under these conditions.

The results of our work demonstrate that DSC cipher is weaker than previously thought and that passive attacks against the privacy of encrypted DECT communications are feasible in practice. Whilst continuous renegotiation of the DSC key during a communication could help to mitigate the privacy risk, we recommend it to be done at least every 30 s to prevent an attacker from being able to retrieve enough cryptomaterial to conduct a successful attack. However, it is our opinion that only the effective roll-out of the DSC-2 cipher now available in the DECT standard can be seen as a definitive solution to the privacy risk introduced by the vulnerabilities of the DSC cipher.

References

1. Coisel, I., Sanchez, I.: Practical interception of DECT encrypted voice communication in unified communications environments. In: IEEE Joint Intelligence and Security Informatics Conference, JISIC 2014, pp. 115–122. IEEE (2014)
2. Coisel, I., Sanchez, I.: Improved cryptanalysis of the DECT standard cipher. In: Cryptographic Hardware and Embedded Systems, CHES 2015, pp. 269–286. Springer, Berlin (2015)
3. Coisel, I., Sanchez, I., Shaw, D.: Physical attacks against the lack of perfect forward secrecy in DECT encrypted communications and possible countermeasures. In: International Wireless Communications and Mobile Computing Conference, IWCMC 2015, pp. 594–599. IEEE (2015)
4. ETSI: Digital Enhanced Cordless Telecommunications (DECT), Common Interface (CI), Part 5: Network (NWK) layer (2011). http://www.etsi.org/deliver/etsi_en/300100_300199/30017505/02.04.00_40/en_30017505v020400o.pdf
5. ETSI: ETSI DECT Official Website (2013). http://www.etsi.org/technologies-clusters/technologies/dect/
6. ETSI: Digital Enhanced Cordless Telecommunications (DECT), Common Interface (CI), Part 7: Security features (2015). https://www.etsi.org/deliver/etsi_en/300100_300199/30017507/02.05.07_20/en_30017507v020507a.pdf
7. Lucks, S., Schuler, A., Tews, E., Weinmann, R.P., Wenzel, M.: Attacks on the DECT authentication mechanisms. In: Topics in Cryptology, CT-RSA 2009, pp. 48–65. Springer, Berlin (2009)
8. Maximov, A., Johansson, T., Babbage, S.: An improved correlation attack on A5/1. In: Proceedings of the 11th International Conference on Selected Areas in Cryptography, SAC'04, pp. 1–18. Springer, Berlin (2005)
9. McHardy, P., Schuler, A., Tews, E.: Interactive decryption of DECT phone calls. In: Proceedings of the fourth ACM Conference on Wireless Network Security, pp. 71–78. ACM (2011)

10. Molter, H.G., Ogata, K., Tews, E., Weinmann, R.P.: An efficient FPGA implementation for an DECT brute-force attacking scenario. In: Fifth International Conference on Wireless and Mobile Communications, 2009, ICWMC'09, pp. 82–86. IEEE (2009)
11. Nohl, K., Tews, E., Weinmann, R.P.: Cryptanalysis of the DECT standard cipher. In: Fast Software Encryption, pp. 1–18. Springer, Berlin (2010)
12. Sanchez, I., Baldini, G., Shaw, D., Giuliani, R.: Experimental passive eavesdropping of digital enhanced cordless telecommunication voice communications through low-cost software defined radios. Secur. Commun. Netw. **8**(3), 403–417 (2014)
13. Weiner, M., Tews, E., Heinz, B., Heyszl, J.: FPGA implementation of an improved attack against the DECT standard cipher. In: Proceedings of the 13th International Conference on Information Security and Cryptology, ICISC'10, pp. 177–188. Springer, Berlin (2011)