Improved Cryptanalysis of the DECT Standard Cipher

Iwen Coisel, Ignacio Sanchez*

European Commission, Joint Research Centre (JRC)
Institute for the Protection and the Security of the Citizen (IPSC)
Digital Citizen Security Unit
Via Enrico Fermi 2749, 21027 Ispra (VA), Italy
{iwen.coisel, ignacio.sanchez@jrc.ec.europa.eu}

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 approach which is more effective than the Nohl-Tews-Weinmann (NTW) attack and requires four times less plaintext material. Under the best conditions, our known plaintext attack requires only 3 minutes of communication compared to 10 minutes for the NTW attack. Our approach is able to quickly recover the secret key with a success rate of more than 50% by analysing 2^{13} keystreams and performing an exhaustive search over 2^{31} keys. Additionally, the attack was successfully conducted against real intercepted DECT traffic where the plaintext was only 90% accurate. To the best of our knowledge, the approach we present in this paper is the most effective cryptanalysis published so far against the DSC cipher.

Keywords: DECT, privacy, security, cryptanalysis, stream cipher

1 Introduction

The Digital Enhanced Cordless Telecommunications standard (DECT) is an ETSI standard used in cordless telephony, widely deployed worldwide both in residential and enterprise environments[1]. In traditional residential scenarios, the DECT base station is directly connected to the analogue telephone line and provides telephony and a battery recharge point for one or more wireless handsets. 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 the corporate Unified Communication Systems. These systems integrate several types of voice and data communications, such as VoIP and Videoconferencing,

* Both authors have contributed equally and are presented in alphabetical order.

[1] The number of DECT devices sold reaches 820 million with a proliferation of 100 million new devices per year.

which have recently become available in the consumer market. Nowadays it is common to find low-cost DECT cordless phones able to handle both land-line 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. [3] employing special purpose hardware. More recently, Sanchez et al. [7] demonstrated that eavesdropping of non-encrypted DECT voice communications can be effectively performed using widely available 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 the DSC encryption algorithm.

Although the details of the DSC algorithm were never made openly available in the public standard[2], Nohl et al. [6] reverse engineered it from a hardware implementation and published the details of the algorithm along with a reference software implementation.

The literature detailing the security of the DSC encryption algorithm is quite limited, especially when compared to other encryption algorithms such as A5/1 used in mobile telephony. In [5] McHardy et al. demonstrated an active replay attack against DSC, and they were able to decrypt a recorded call by interactively recovering the keystreams used to encrypt it. Nohl et al. [6] proposed the first (and only) cryptanalysis method, the Nohl-Tews-Weinmann (NTW) attack, capable of recovering the DSC key after analysing large quantities of encrypted traffic. Their work constitutes the foundation of the present paper.

In [1], Coisel and Sanchez have presented an attack aiming at retrieving the long term key of the DECT device. While this attack is efficient (approx. 2^{10} operations), it requires that a key establishment protocol happens between the handset and the base station. Such protocol typically never takes place in a residential environment as the devices are paired in advance by the manufacturers.

In this paper we present an improved cryptanalysis of the DSC encryption algorithm that is faster than the NTW attack and requires substantially less plaintext material to be effective, making its usage viable in practical scenarios.

The paper is structured as follows. In section 2 we review in detail the DSC stream cipher and its functions. In section 3 we describe the Nohl-Tews-Weinmann attack and its results. In section 4 we introduce our attack against DSC, and in section 5 we describe the working implementation. In section 6 we present and analyse the results obtained by our approach, and we compare with the results obtained by the Nohl-Tews-Weinmann method. Finally, in section 7 we present the conclusions and we outline future research lines on the topic.

[2] The internal detail and reference implementation of the DSC algorithm is only available under non-disclosure agreements.

2 The DECT Standard Cipher

The DECT Standard Cipher (DSC) is a proprietary stream cipher designed as part of the DECT ETSI standard [2]. Although the DSC details were never made public in the DECT standard, Nohl et al. [6] presented a reference software implementation that they obtained through a reverse engineering procedure.

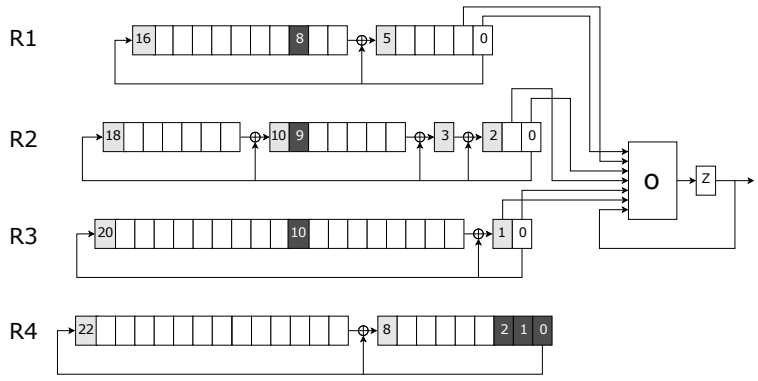


Fig. 1: DSC Stream Cipher

Concisely, the DSC is a 64-bit key stream cipher based on 4 irregularly clocked linear feedback shift registers (LFSRs) $R_1, \ldots, R_4$ in Galois configuration and a non-linear output combiner O with memory z. The three first registers are used to produce the keystream, i.e. the symmetric key used in the One-Time Pad algorithm to encrypt the plaintext. The fourth LFSR register is used, in combination with the others, to control the clocking of the three main registers. As will be described in detail later on, the DSC is initialized with a 35 bits 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 (KSS) of 360 bits each. The first KSS is used to encrypt the DECT frames sent by the base station (Fixed Part), and the second to encrypt the frames sent by the phone (Portable Part). In each case, the first 40 bits are used to encrypt the C-Channel data (that contains 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 DECT ETSI standard [2]. The DSC cipher is depicted in Figure 1 and formally described in the following subsections.

2.1 The DSC Internal Configuration

The internal status of the DSC stream cipher is composed of 81 bits spread amongst the 4 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 Section 2.2. The last register, R4, is used exclusively to control the irregular clocking of the other registers after each round of the DSC. The four LFSR, depicted in Figure 1, are formally described below together with their respective feedback polynomials.

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

The initialization of the internal status, also called key loading procedure, is done 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 clocking during this step).

After the key loading procedure is completed, 40 "empty" rounds are executed during which the generated keystream bits are discarded. After each round, the register R4 is regularly clocked three times. The three main registers, R1, R2 and R3, are irregularly clocked two or three times depending of the values of specific bits (displayed with a dark grey background in Figure 1) of the three other registers. The number of times each register is clocked for a specific round, denoted irr_cl_i, is defined as follows, where $x_{i,j}$ denotes the j-th less significant bit, starting from 0, of the register i.

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

Once the initialization procedure is completed, the round 0 of the DSC starts and produces the first bit of the keystream, denoted z_0. More generally, the keystream bit z_i is output at the end of the round i. All the keystream bits are generated by the output combiner as detailed in the next subsection.

2.2 The Output Combiner

The output combiner O is a non-linear function, more precisely a cubic function. It takes as inputs the two least significant bits of the three main registers, R1, R2 and R3, and the bit from the memory slot corresponding to the previous bit of the keystream. From now on, these 6 bits, taken from the three main registers, constitute what is called in the following the *status* of the DSC, and it is obviously dependent on the round number as well as the key and the IV.

Every round, the output combiner generates a bit that is stored in the memory slot, denoted z in Figure 1. The bit previously stored in the memory slot will be output as a keystream bit, provided the initial 40 rounds were already completed. The new bit stored in the memory slot will be used as part of the input of the output combiner in the next round.

The output combiner is 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.

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

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.

Any modification of a given status would modify in average the output bit in 50% of the cases. However 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 yet it is an interesting line of research that may potentially improve the success rate of the attack.

2.3 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 3 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 real clock of the register i, while $c_{i,l}$ denotes the real clock of the register i 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 also extend the XOR operator and define the XOR operation of two statuses where the bits of the statuses are mutually XORed together.

3 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 certain amount of affine linear equations that specify relations about the key bits. In a second phase, the remaining bits of the key are brute-forced in order to obtain the 64-bit key.

Due to the linearity of the 4 DSC registers, each bit of each register, for a given number of clocks, can be defined as a linear combination of the bits of the key and the bits of the initial vector. Guessing correctly a bit $x_{i,j}$ for a given clock determines a linear equation. The goal of the attack is to guess correctly the status of the DSC and the corresponding clocks (c_1, c_2, c_3) in order to obtain a sufficiently large number of equations. Once n independent equations have been derived, the 64-bit key can be recovered following an exhaustive search over the remaining 2^{64-n} values.

3.1 Guessing Correctly a Status

In this section, we describe how the NTW attack manages to retrieve the status of the DSC. In order to do so, first we 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, or more precisely, the fact that the following equality holds for any triplet of clocks $sc = (c_1, c_2, c_3)$.

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

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. The equality $S_{sc}(\text{KEY}, \text{IV}) = S_l(\text{KEY}, \text{IV})$ obviously holds if the registers are clocked accordingly to the values in sc at the end of the round l. The generated status and the bits of the keystream also 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.

Both the key and $S_{sc}(\text{KEY}, \text{IV})$ are unknown. However, the IVs and the bits of the keystream are known (as it is a known-plaintext attack). Therefore it is possible to evaluate this equation with the 64 possible statuses. The correct status will necessarily belong to the subset $\mathcal{S}$ of the 32 candidate statuses that verify this particular equation. Using the corresponding IV, $\tilde{s} = S_{sc}(0, \text{IV})$ can be computed. As a conclusion, if sc is the correct triplet of clocks, then $S_{sc}(\text{KEY}, 0)$ is 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 the round l, the correct status still has a probability of 50% to be in this set according to Nohl et al. [6]. Consequently, the correct status has more than 50% of chance to be in this set, whilst this probability is 50% for all the other statuses.

This experiment can be seen as a Bernoulli trial where a success is the presence of the status is in the list of candidates. If repeated sufficiently, the most frequent status should therefore be the one of $S_{sc}(\text{KEY}, 0)$ due to this bias.

3.2 Determination of More Equations

If the evaluated equation $eqn(sc, l)$ is properly selected, then the previous step has successfully determined 6 equations for the DSC key, one for each bit of

the status. 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 time. 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 clock, a frequency table is generated to store the "score" of each potential candidate status.

Once all the 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 are defined leading to 108 equations. A subset of these equations is selected according to a certain rank (see [6] for more details) and the solvability of the obtained system. When enough equations have been selected (around 30) the remaining bits are brute-forced.

3.3 Results of the Nohl-Tews-Weinmann Attack

Nohl et al. have conducted their known-plaintext attack against both the C-Channel and the B-Field considering the availability of different quantities of plaintext. They consider the probability that the system of equations defined is valid regarding the correct key. Their experiments consider different sizes for the system, from 10 to 40 equations. For comparison purposes, we summarise in Table 1 the most relevant results for the C-Channel and the B-Field respectively for different quantities of available plaintext. Full details about the results obtained by the NTW attack can be found in the original article [6].

	C-Channel			B-Field		
Number of plaintext	8192	16384	32768	16384	32768	65536
10 equations	2%	30%	96%	2%	30%	92%
20 equations	0%	2%	78%	0%	2%	65%
30 equations	0%	1%	48%	0%	0%	28%
40 equations	0%	0%	11%	0%	0%	4%

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

To reach a probability of success of 50% against the C-Channel in order 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. [8] 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%.

4 A Theoretical Model of an Improved Cryptanalysis

Instead of considering separately each bit of the internal status of the DSC as is done in the NTW attack, our attack processes directly 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 first step. Furthermore, the underlying theoretical model used to give score to the potential candidate 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 targeted status 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. The weight of an equation is computed according to the probability that the correct status belongs to the specific subset of candidates that satisfy this 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 a candidate. Starting with the first candidate the remaining bits of the key ($64 - 3(len_c + 1)$ bits) are then brute-forced in the last step of the attack.

4.1 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 achieved 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 that includes 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 keystream 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 clocks twice is 50% as the one that it clocks three times. We 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:

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

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.

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

When at least one clock differs from the targeted one, as stated in the article [6], we could expect that the correct status is in the subset of candidates with a probability of 50%. However, due to the particular behaviour of the output combiner previously described in Section 2.2, the probability that the equation remains verified is not exactly 50%when the clocks differ. Indeed, if two targeted clocks are correct the equation is verified with a probability of 56.25%. 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.

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

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

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

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

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 [4], the weight w_{eqn} associated to 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.

4.2 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 $\mathcal{K}$. $\mathscr{T}$ 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 $\mathscr{T}$. As a reminder, the statuses that verify a given equation are those potentially corresponding to $\tilde{s} = S_{sc}(\text{KEY}, IV)$. The "IV part" of 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 bigger scores.

4.3 Exhaustive Search among the Remaining Bits

A system of $3(len_c + 1)$ linear equations to the key with 64 unknowns can be defined from each candidate in the table $\mathscr{T}$. The key is a solution of the system defined by the correct candidate. To ease the brute-force step the Gaussian reduction of the system is used. All the obtained equations are independent if the length of the range of clock is below 17 (size of the smaller register). Thus $3(len_c+1)$ bits of the key are determined by these equations while the remaining ones need to be brute-forced.

In order to carry out the last step, we have developed a CPU SIMD-based implementation that loads the equations derived in the cryptanalysis step and performs the exhaustive search over the remaining bits of the key. For each possible combination of keybits to be explored it retrieves the remaining bits of the candidate key using the linear relations. Then the implementation compares the generated keystream with a portion of known keystream at least longer than the number of keybits to be explored. Our implementation evaluates 500 million keys/sec in a Core i7 (AVX) workstation.

5 Improved Implementation of the Cryptanalysis

The attack described in the previous section may retrieve the searched key with a relatively good probability of success, whilst requiring a considerable amount of computing time to define enough linear relations among the key bits. In this section we present a success/time trade-off to hasten considerably the execution time whilst maintaining a good success rate.

5.1 Efficiency Consideration

The parameter that ultimately influences the efficiency of the attack is the length len_c of the clock range. Increasing it 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 $\mathscr{T}$, that are both cubic function in len_c, as formally defined below.

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

The evaluation of the equations for all the candidates of $\mathscr{T}$ is a process that can be highly parallelised due to the independency of the actions. Consequently, a HPC may be able to achieve this step in a few hours for a length of 7. However, this length only allows the determination of 20 bits of the key. According to the implementation of [8], the brute-force of the remaining bits would take around 30 hours on a modern GPU.

To reduce the workload, we could split the range of clocks in sub-ranges and apply our attack to them. Combining the best candidates of each frequency table allows the determination of the same amount of status bits as in the initial range while reducing the computing time required. 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 computing time required to carry out the attack and probability of success. The small gain in the probability of success is not really relevant compared to the loss of efficiency experienced following this approach. Indeed, more sub-ranges would be required to obtain the same amount of linear equations.

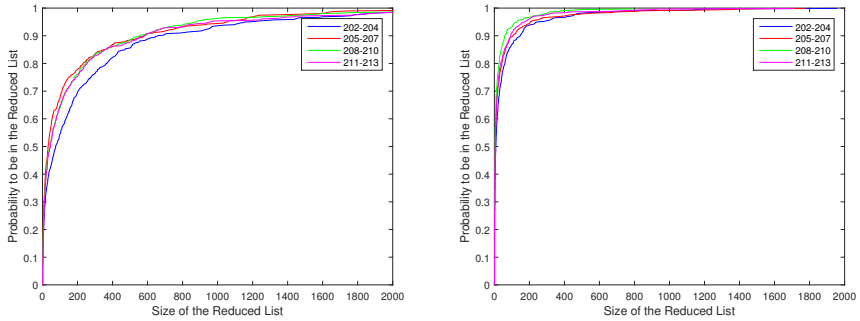
5.2 A Time-Accuracy Trade-Off

Our trade-off use several sub-ranges of reasonable length (typically 3 or 4 clocks each) without any redundancies on which we apply the attack previously defined in Section 4. 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-tables. The score associated to these new candidates is initially set to the sum of the two scores taken from the original candidate tables. The attack described in Section 4.2 is reapplied to this new subset considering the equations that use clocks overlapping amongst 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 next sub-range and so on until all the sub-ranges have been processed.

The numbers N_T and N_R of the most promising candidates to be taken from the table of candidates is a critical parameter as it allows to hasten the experiment to the detriment of the success probability. A small value decreases the number of candidates that will be evaluated against the equations for a wider range of clocks. At the same time, it reduces the chances that the correct candidate belongs to the reduced list and thus that the attack is successful.

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 threshold for the selection of the most promising candidates, we have experimentally determined the probability that the correct candidate was in the reduced list. We have conducted these experiments for different quantity of available plaintexts, for several sub-ranges against both the C-Channel and the B-Field. Figures 2a and 2b present the probabilities that the correct candidate is in the top N_T of the corresponding sub-ranges for respectively 16384 and 32768 available plaintexts when attacking the B-Field.



(a) 16384 plaintexts (b) 32768 plaintexts

Fig. 2: Probability that the correct candidate is in the top N_T for several sub-ranges while attacking the B-Field.

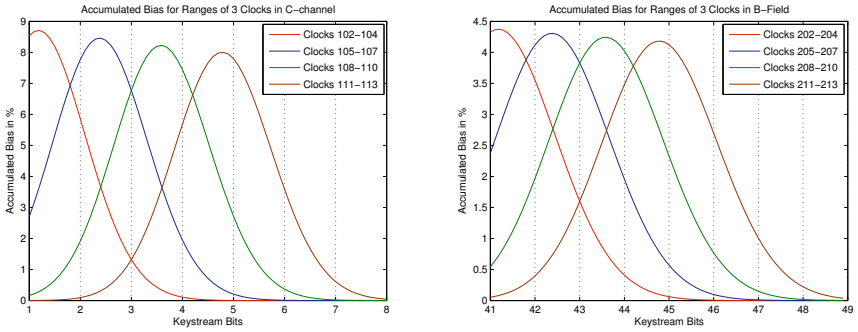
As it can be observed in these figures, the increase of available plaintexts increases the chances that the correct candidate is in the reduced table. However, it should be kept in mind that it also increases the time required for the evaluation of all equations. As a consequence, the size of the reduced 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 we conduct (see Section 6 for more details on these experiments), it can 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.

5.3 Selection of the Relevant Equations

Following the approach described in the previous subsection, 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 3 represents graphically the computed accumulated bias for different keystream bits sub-ranges of clocks. Based on these results, we have selected the list of pertinent equations to be evaluated in each sub-range.



(a) C-Channel (b) B-Field

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

It can be observed in Figure 3 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 successfully attack the B-Field data compared to the C-Channel one, as it will be described in the next section.

6 Experimental Results of Our Attack

In order to test experimentally the cryptanalysis attack we have described, we have conducted several experiments, both with simulated and actual data,

aimed at validating and benchmarking our attack. Both the attacks against the C-Channel and the B-Field have been considered, using the ranges of clocks [102,113] and [202,213] respectively.

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 keystream are used for the attack.

6.1 Results Based on Simulated Data

In this subsection we present the results obtained in our experiments using simulated data. In our experiments we have generated a total of 200 random DSC 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 amount of plaintext is directly linked to the number of minutes of the encrypted voice call that had to be recorded to obtain them. We have performed the experiments both for the C-Channel and the B-Field. 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 were differing 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 success probability 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. Discarding the two extreme bits of the status reduces the number of linear equations among the key bits from 39 to 33, increasing the duration of the final exhaustive search step. Nevertheless, the exhaustive search would still be conducted in seconds while this choice will sensibly increase the success probability, potentially outputting the correct candidate even if it was previously discarded in a preliminary sub-list.

Table 2 displays the percentage of time the correct status was output in first position by our attack so that they can be compared on a fair basis with the results of the NTW attack [6]. These results are slightly better (up to 10% additional keys retrieved) when the brute-force step evaluates the other possible statuses output by the final step. For example, while the success probability is about 69% when extracting 39 equations attacking the B-Field with 32K plaintexts, it reaches 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 around 8192 plaintexts from the C-Channel. In this case, the final exhaustive search step is able to retrieve the correct key with a probability of $1 - 2^{-64} \approx 100\%$ in around 5 seconds using our CPU SIMD-based implementation with an Core i7 (AVX) workstation. To reach a more or less equivalent success rate, the NTW cryptanalysis attack requires at least four

	C-Channel			B-Field		
Number of plaintext	4096	8192	16384	8192	16384	32768
9 equations	35%	85%	98%	19%	69%	94%
21 equations	16%	73%	97%	10%	57%	90%
33 equations	6%	**55%**	95%	3%	**36%**	82%
39 equations	2%	33%	84%	1%	21%	66%

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

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 hours.

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 per second (around 5) are typically sent over the C-Channel, in comparison with the 100 that are sent per second in the B-Field. Consequently, 20 times more plaintext is produced in the B-Field which makes the attack over the B-Field more realistic in terms of minimum call duration required to conduct the attack.

To illustrate this we present some values regarding the communication time. To reach a 50% chance to retrieve the key using C-Channel plaintext, 20 minutes of conversation have to be recorded whilst the NTW attack requires around 1 hour and 50 minutes to reach the same success rate. Considering a B-Field attack, we achieve a success rate of 30% using just 180 seconds of communication when the NTW attack needs more than ten minute for the same result.

6.2 Results Based on Real Data

In order to validate our findings with actual data, we have performed a set of experiments applying our cryptanalysis to break actual DECT encrypted calls. For that purpose 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. [7].

The attacks we have performed were focused on B-Field data. Since our attack, like the NTW one, is a known-plaintext attack, the attacker requires knowledge about the first 9 bits of keystream. The approach followed for the experiments was to use the silence transmitted by the call in order to obtain the plaintext, assuming it would be encoded as all ones by the G726 codec. However, during our experiments, we have noticed that often the sound transmitted was not perfect silence leading to less than 100% accurate prediction of the plaintext.

In a first round, we validated our attack assuming 100% accuracy in the prediction of the plaintext. In order to do so, we followed the approach described by Coisel and Sanchez in [1] to obtain the long-term key shared by the base

station and the handset. With this knowledge the session key derived at the beginning of the communication can be retrieved and thus the communication can be decrypted leading to a full knowledge of the plaintext.

This step is by no means required to conduct the cryptanalysis attack described in this paper and it was performed with the sole purpose of helping us to debug the process and work with a perfect accurate prediction of the plaintext. A more realistic scenario where the complete recovery of the plaintext material is not possible is presented in the experiments described in the next section.

Our first attempts to recover key were performed by analysing 5 minute DECT encrypted calls (corresponding to 32K samples) from several handsets of different brands. In our first attempts, our tests had a success rate of over sixty-six percent. When analysing calls over 10 minute, our tests were all successful.

6.3 Partially-Known Plaintext Attack

As said in the previous section, in practice an attacker would encounter many difficulties predicting the 9 bits of keystream required to conduct the attack against the B-Field. We have found out that depending on the background and equipment 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 keystream bits used in an equation is wrong, the probability that the correct status belongs to the candidate list drops to 50%. We could consider that in this case, the vote cast by that equation is randomly distributed among the possible status 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. Figure 4 displays the results of the experiment. The loss of accuracy can be compensated by increasing the size of the reduced list at the price of the overall efficiency. Analysing more plaintext can as well compensate the loss of accuracy. To measure the decrease of the success probability we have benchmarked our attack using simulated data against the B-Field assuming the knowledge of 32,768 and 64,536 plaintexts for several degrees of accuracy. The results presented in Table 3 are the success probabilities to retrieve the session key among the final output list of candidates.

We have applied the attack on the real capture we made for different parameters to validate that our attack works in practice even if the plaintext is only partially known. The plaintext used in the attack represents mostly silence and is accurate at approximately 90%. All our attempts using 65K plaintexts whilst defining 39 linear relations amongst the key bits were successful. The success rate of the tests we have conducted using 32K plaintexts are in agreement with the results of Table 3.

We have made an interesting discovery analysing decrypted data coming from our practical experiments intercepting DECT communications. The distribution of the zeros, bits differing from pure silence, is not uniform over the 9 bits of the

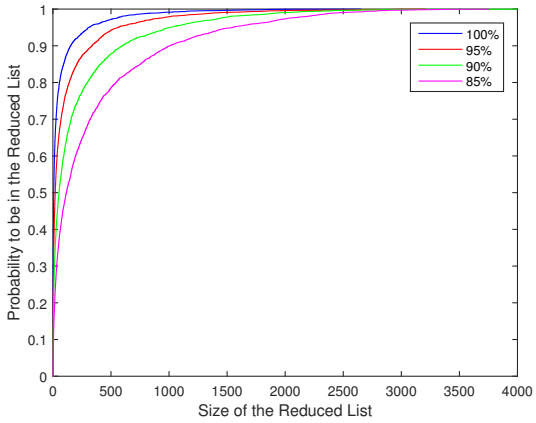


Fig. 4: Probability that the correct candidate belongs to the reduced list

keystream. We have noticed unexpected patterns in groups of 4 bits, probably derived from the way the G726 encodes the voice. This fact opens the door to an optimisation of our cryptanalysis attack for these scenarios where the accuracy in the recovery of the keystream is not 100%. Indeed, given the fact that some keystream bits are more likely to be ones than others, the weight of the equations can be adjusted on the basis of the probability that the keystream bit is guessed correctly. This improvement would reduce the impact of the loss of accuracy and will be explored in future research.

	32768 plaintexts				65536 plaintexts			
Accuracy of plaintext	100%	95%	90%	85%	100%	95%	90%	85%
9 equations	96%	92%	71%	55%	100%	100%	100%	92%
21 equations	91%	78%	57%	37%	100%	100%	96%	81%
33 equations	85%	65%	42%	21%	99%	98%	87%	70%
39 equations	81%	56%	28%	11%	99%	94%	85%	63%

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

7 Conclusions and Future Developments

In this paper we have presented an improved cryptanalysis attack against the DECT Standard Cipher, leveraging over the clock guessing approach introduced by the NTW attack. In comparison, our approach offers better accuracy and requires substantially less amount of keystream material.

One of the limitations of the former NTW attack was precisely the difficulty to obtain enough keystreams. To reach a 50% success probability whilst defining 30 linear equations, the NTW attack requires 1 hour and 50 minutes of communication in the case of the C-Channel and more than 15 minutes in the case of the B-Field. Our attack retrieves the key with better success probabilities requiring over twenty-five percent less plaintext, respectively analysing a 27 minutes call for the C-Channel or 3 minutes one for the B-Field.

We have tested our attack with real interceptions of several DECT phones from different brands. Our tests were successful in many cases even when the prediction of the plaintext was not 100% accurate. We have analysed the impact of this loss of accuracy by randomly modifying the plaintext in our simulation.

However we have noticed that the distribution of the "errors" is not uniform over the bits of the keystream. A more carefully analysis of the origin and the distribution of these patterns would potentially help in minimising the impact form the loss of accuracy. Furthermore, the influence of the inputs to the output combiner in the probability distribution of the output should also be considered to refine and improve the theoretical model behind the computation of weights.

The cryptanalysis presented in this paper shows that passive attacks against the privacy of DECT encrypted communications can be conducted in practice. We strongly believe in the need to migrate to a more secure solution. A continuous renegotiation of the DSC key could help mitigating the risk but, on the basis of our results, it should be done at least every 30 seconds so that an attacker would not be able to retrieve enough plaintext material to launch an effective attack. However, a definitive solution will only arrive with the effective deployment of DECT devices using the new DSC2 algorithm recently included in the standard, which up until now has shown no similar weaknesses.

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. ETSI: ETSI DECT Official Website (2013), http://www.etsi.org/technologies-clusters/technologies/dect/
3. 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 (2009)
4. 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-Verlag (2005)
5. 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)
6. Nohl, K., Tews, E., Weinmann, R.P.: Cryptanalysis of the DECT Standard Cipher. In: Fast Software Encryption. pp. 1–18. Springer (2010)
7. Sanchez, I., Baldini, G., Shaw, D., Giuliani, R.: Experimental Passive Eavesdropping of Digital Enhanced Cordless Telecommunication Voice Communications through Low-Cost Software Defined Radios. Security and Communication Networks (2014)

8. 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-Verlag (2011)