RFID Devices and Cryptography: Analysis of the DST40

Considerations from a Reading of

Bono, S.C., et al, *Security Analysis of a Cryptographically-Enabled RFID Device*,
In P. McDaniel, ed., USENIX Security '05, pp. 1-16. 2005.

and the popularized version of the same paper

Analysis of the Texas Instruments DST RFID, http://www.rfid-analysis.org/

Dennis Galvin
CSEP 590 TU (Practical Aspects of Modern Cryptography)
University of Washington
Winter 2006
07-Mar-2006
Final Paper

Introduction

Radio Frequency Identification (RFID) devices are small wireless devices which emit an identification code when energized and then queried by readers. The utilization of RF signals in civilian remote control applications has been exploited for more than 50 years[1]. In late 1988, keyless vehicle entry systems were introduced[2] to the mass produced automotive market, which utilized high frequency FM radio signals to lock, unlock doors, arm, disarm alarms, and later on to start, kill vehicle engines. The principal problem with the early systems is they were insecure, and a capture of the radio signal could be replayed (or analyzed, digitally resynthesized on demand and then played) to open the door lock, disarm the alarm system, or start the vehicle ignition. In 1999 Texas Instruments began marketing a cryptographically secured RFID system, "TI-RFID™," for use in automotive and other applications. TI's system uses low-frequency FM signal transmission which has the added benefits of better glass penetration and reduction of vehicle "Faraday cage" signal attenuation. TI's cryptographically secured RFID system has been employed in a number of automotive applications to date.

For this paper, I read and attempted to understand, using concepts learned in this class, the Bono, et al paper[3] from the 14th USENIX Security Symposium, and it's precursor draft[4] from the RSA Labs web site. The authors consist of three Johns Hopkins University graduate students, one JHU faculty member, and two RSA Labs scientists. In the papers, they describe: 1) Reverse engineering the cryptographic cipher; 2) Building a key cracker with Xilinx FPGAs to recover the cryptographic key from a DST device using 2 responses to arbitrary challenges; 3) Building a device to simulate the RF protocol of the DST40.

Description of the DST40

The Texas Instruments DST40 (40-bit Digital Signature Transponder) is a cryptographically secured RFID device used in a number of mass market applications. Among those are: 1) Vehicle immobilizers in 2005 Ford vehicles, as well as systems used in some European manufactured automobiles; 2) Exxon Mobil SpeedPass™ fuel purchasing system. It is estimated that as of January 2005, 150×10^6 automotive immobilizer keys had shipped with automobiles, of which some portion use this RFID chip.

[1] For instance, in 1959, radio control systems for overhead cranes was introduced. http://www.berlet.com/belp3.html

[2] TRW designed the first one which was put into production in 1988. http://www.trw.com/whoweare/main/0,1003,1_516%5E2%5E516%5E516,00.html

[3] Bono, SC; Green, M; Stubblefield, A; Juels, A; Rubin, AD; Szydlo, M, *Security Analysis of a Cryptographically-Enabled RFID Device*, In P. McDaniel, ed., USENIX Security '05, pp. 1-16. 2005.

[4] Bono, SC; Green, M; Stubblefield, A; Juels, A; Rubin, AD; Szydlo, M, *Security Analysis of a Cryptographically-Enabled RFID Device*, Draft of a paper. http://www.rsasecurity.com/rsalabs/staff/bios/ajuels/publications/pdfs/DSTbreak.pdf Some details of this paper differed significantly from the published paper referenced above.

The DST40 contains a 40-bit field programmable[5] cryptographic key (which by design is preshared with the reader) and a 24-bit factory programmed ID. The DST RF Ouput is FM-Frequency Shift Keyed (FM-FSK). The output operating frequencies are 134.2 kHz, and 128.2 kHz, with 16 RF cycles per transmitted bit (134.2 kHz = 1, 128.2 kHz = 0). There is a reader synchronization protocol in which the DST emits a series of zero bits (128.2 kHz cycles) followed by 0x7e. The DST RF Input (Reader output) differs substantially running 134.2 kHz AM (amplitude modulated). The amplitude switches between full and low (or "off"), and each bit is represented by the "off" time of the signal (Short off-time = 0 bit, Longer off-time = 1 bit).

High Level Communications protocol between the DST and a reader

Refer to Figure 1 below for this overview. The reader emits an RF pulse to energize the RFID chip. The DST then transmits its 24-bit factory programmed ID. Upon receipt the reader constructs and transmits a 40-bit challenge to the DST. The DST then encrypts challenge with its 40-bit key and transmits the least significant 24-bits of encrypted challenge. The reader then examines response, and decides to provide or deny access.

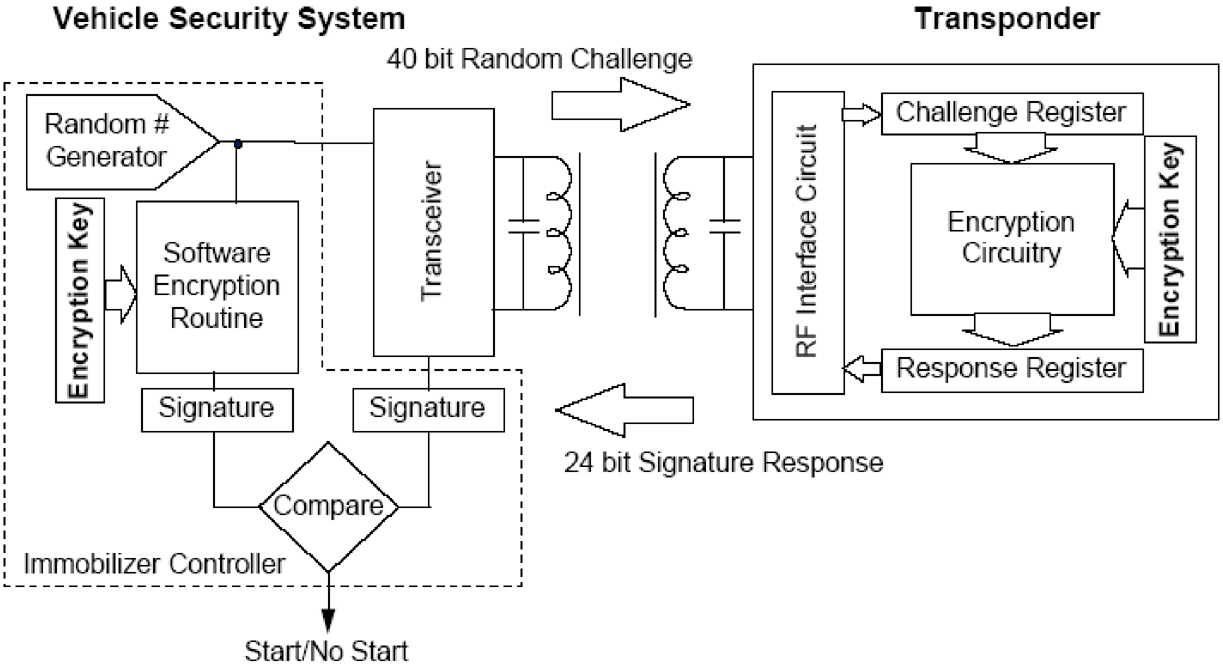


Figure 1: Conceptual diagram of the TI DST based vehicle immobilizer system.[6]

[5] Production DSTs in common use do not have field programmable keys, but rather immutable factory programmed keys.

[6] Knebelkamp, M., Freising, H.M., *Latest Generation Technology for Immobilizer Systems*, TI White Paper, approximate date 1997. Linked from http://rfid.bluestarinc.com/resources/Immobilizer_Systems.pdf

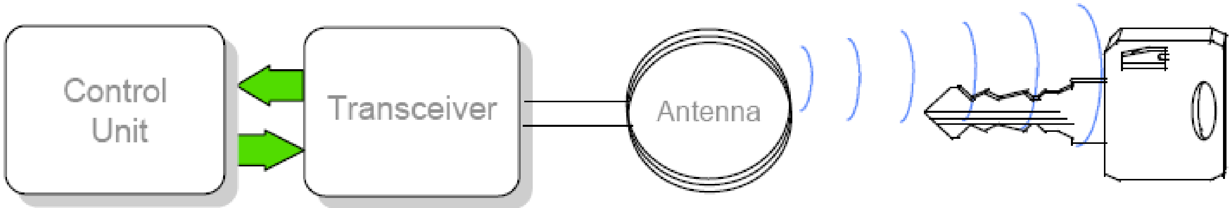


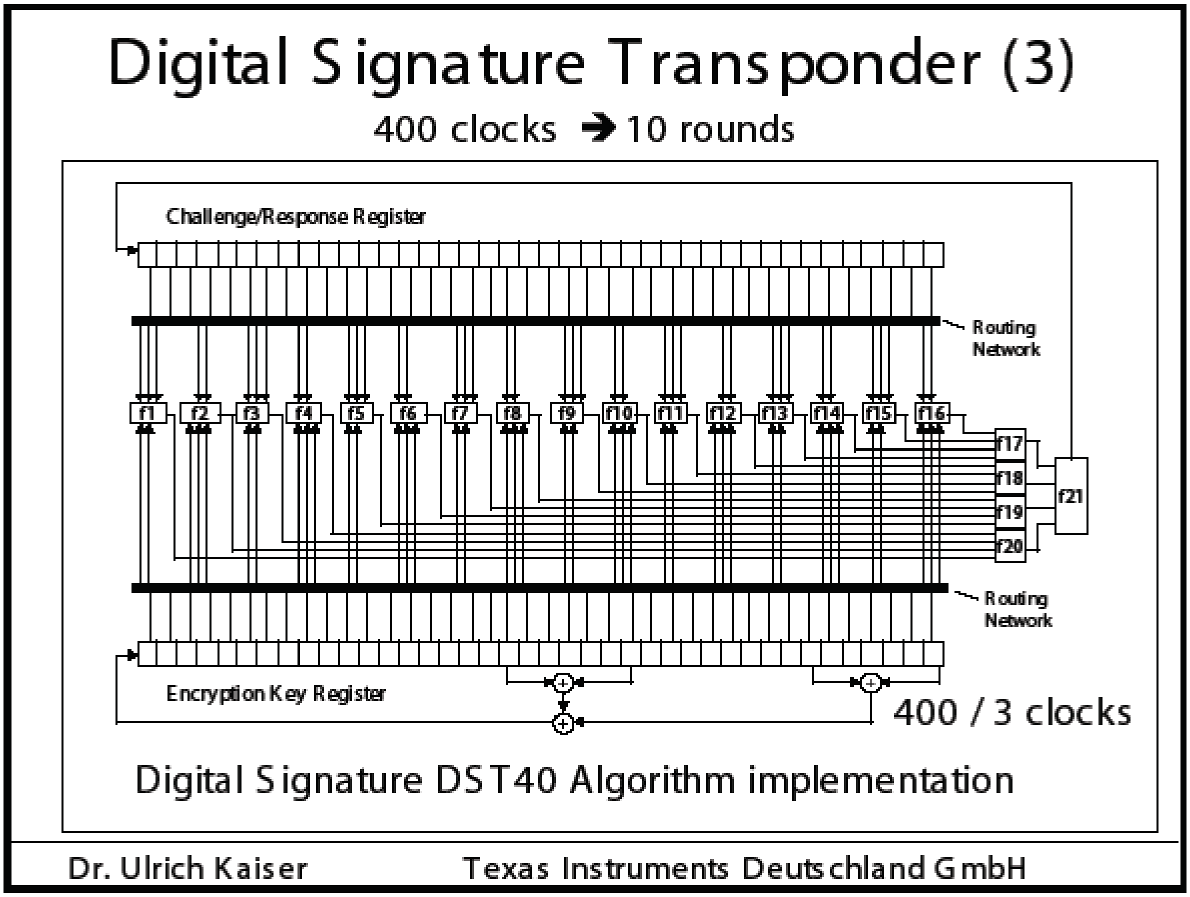
Figure 2: Pictorial representation of the TI DST based vehicle immobilizer system[7]

In Figures 1 and 2, we see diagrams of the active components in a sample vehicle immobilization system. In this system, the engine computer will not start the vehicle unless the DST in the ignition key successfully encrypts the challenge and transmits the result to the transceiver which then emits the boolean (1-bit) signal to permit starting the vehicle to the engine computer[8].

The authors performed a black-box analysis to recover the missing and incorrect details from the DST diagram reproduced in Figure 2. The missing details were: the routing network, the details of boxes f_1 to f_{16} (referred to as the *f*-boxes by the authors), the details of boxes f_{17} to f_{20} (*g*-boxes) and box f_{21} (*h*-box). The diagram resembles the simple LFSR (linear feedback shift register) presented in class with a few added twists: 1) The internal state of the LFSR also includes 40 bits of encryption key; 2) The challenge – response and key registers are loaded with their initialization vectors (challenge and key) before encryption begins; 3) The encryption key is shifted and some information is fed back into the key as the encryption proceeds; 4) The outputs of the *f*-boxes are fed into the *g*-boxes and then into the *h*-box before feeding back into the challenge response register.

[7] Knebelkamp, M., Freising, H.M., ibid.

[8] Depending on how the output of the RFID reader is fed to the engine computer, it might be trivial to bypass this using only a shorting clip or mechanics "cheater" plug.


Digital Signature Transponder (3)

400 clocks ➔ 10 rounds

400 / 3 clocks

Digital Signature DST40 Algorithm implementation

Dr. Ulrich Kaiser Texas Instruments Deutschland GmbH

Figure 3: DST40 Algorithm as presented by U. Kaiser[9].

Cryptanalysis

The authors purchased an evaluation kit, along with DSTs marketed by Texas Instruments. This allowed them to use the DST as an "oracle" because they were able to field program the DST keys. This allows them to obtain chosen plain/enciphered text pairs for cryptanalysis.

Recovering the key schedule

They started by using the Zero key (40 bits of zero). The Zero key would remain all zeros throughout the response generation. Then by using known challenges from the TI reader, they were able to examine the output (response). Early on they determined (by trial and error) the output of the h-box was actually 2 bits (this differed from the 1-bit in the Kaiser diagram). They conducted further experiments which showed the algorithm executed over 200 rather than 400 clocks to generate the output.

Reverse engineering of the key schedule required using non-zero keys. Using the DST as an

[9] Kaiser, U. *Universal immobilizer crypto engine.* In Fourth Conference on the Advanced Encryption Standard (AES) (2004). Guest Presentation. Slides linked from: http://www.aes4.org/english/events/aes4/downloads/AES4_UICE_slides.pdf

“oracle,” they determined the key was updated every three cycles beginning with the second cycle, and recovered the key update polynomial $k_0 = k_{39} \oplus k_{37} \oplus k_{20} \oplus k_{18}$. By examining 150 challenge response pairs in which individual bits of the challenge were experimentally flipped, they determined that bits 38 and 39 of the challenge register were always XORed into the 2-bit response output of the *h*-box for each round. This indicated the algorithm to be a type of unbalanced Feistel Network[10].

Recovering the internal bit routing networks

The working hypothesis (from the Kaiser diagram) was that each of the *f*-boxes, and the *g*-boxes had a single bit output. The diagram also implied each of the f-boxes had inputs composed of 2 challenge register bits and 3 key register bits or vice versa. Running a series of experiments in which they flipped either two specific key bits or two specific challenge bits in all four possible combinations of each 2-bit trial, they were able to determine whether those specific bits had an effect on the output of a given *g*-box. This determination was made from the observation that if two bits were inputs to the same *g*-box, then the output of the *h*-box would have at most 2 possible outputs, and if they were inputs to different g-boxes, the output of the *h*-box could have at most 4 possible outputs. They were thus able to eliminate pairs with more than 2 *h*-box outcomes. By applying this observational test systematically starting with bit 0 of the challenge, they were able to determine that 60 of the 80 available bits from key and challenge were not routed to the first *g*-box. For *g*-boxes 2 to 4 they decreased the search space by those bits already associated with g_1 upto g_{n-1} where g_n was the box being determined. The regular routing pattern inferred by the Kaiser diagram was thus constructed and validated.

An extension to the concept for determining the bit routing patterns for the *g*-boxes was used to determine the bit routing patterns to the *f*-boxes. The basic idea (substantially abbreviated here) was to test individual sets of 5 bits, B for input candidates to each *f*-box. For each candidate set B, a set B_Complement of all other bits was fixed and held invariant. Then by iterating through the possible assignment permutations of the members $b_1..b_5$ and examining the results as the output of *h*, they were able to deduce the membership of individual key and challenge register bits in the *f*-box inputs. As suggested by the Kaiser diagram, each *f*-box had either a combination of 2 key bits and 3 challenge bits or 3 key bits and 2 challenge bits. Two of the f-boxes were special in that they had only 4 inputs (2 key and 2 challenge bits)[11].

Building the tables for the *f*, *g*, and *h*-boxes

Truth tables were then generated for each of the possible inputs to the *f*, *g* and *h*-boxes. For the *f*-boxes, this was simply iterating through all 2^5 possible input value sets. The authors point out that there was no way to absolutely determine if the output of any given *f*-box was actually 1 or 0 because the value was not directly inspectable, however this is just a name mapping convention.

[10] The L and R parts consist respectively of the 38 left most bits, 2 right most bits of the challenge – response register in the Kaiser diagram.

[11] The authors' Figure 4 indicating the inputs to the *f*-boxes does not demonstrate this, but I assume some sort of typo in the construction of the table for publication.

For any given *g*-box, there were 4 inputs from the associated *f*-boxes determined by the routing experiments. Iteration of the 2^4 possible input value sets yielded the truth tables. The same technique was involved in constructing the *h*-box table with the added twist that the *h*-box generates 2 bits of output (4 possible values rather than 2).

Cracking the Keys

If the cipher as a whole has no significant exploitable weaknesses[12], a search of the key space is required to recover the key for a specified challenge – response pair. Assuming the previously detailed reverse engineering work was correct, it should be a matter of sufficient computing cycles to actually recover a DST key given an arbitrary challenge and the DST response.

They opted for using an FPGA-based (Field Programmable Gate Array) implementation rather than a standard PC[13]. The FPGA (Xilinx XC3S1000) and an evaluation board (Xilinx Spartan 3 development board) were available in single quantity pricing for less than $200[14]. They were able to program a hardware solution in a single core which performed a complete encryption in 200 clock cycles. By placing 32 of these cores on the FPGA and using a 100 MHz clock on the board, they were able to crack approximately 16 x 10^6 keys per second (nearly 2 orders of magnitude improvement over a software only solution on a high end Intel based PC). This was significant, as the keyspace to search is 40-bits, and a key could now be recovered on average in ~ 11 hours[15]. As proof of concept, they were able to recover the key from a SpeedPass™, and then trivially compute matching responses to the challenge from the recovered key and the reverse-engineered algorithm.

They then further parallelized their key cracking system by purchasing 15 more FPGAs and boards, and connecting them with ribbon cable to create a 512 core key cracker. After presenting their preliminary result to Texas Instruments[16], TI assisted them by providing 5 DSTs with factory programmed, immutable keys. Their 16 FPGA cracker cracked all five in a total of less than 2 hours using a pair of challenge – response interactions for each DST. The authors are working on a Hellman time-space tradeoff machine they estimate to be “capable of a success rate of 99+%, should require about 10 GB storage[17], and should operate in under a minute on a fast PC.” They were still pre-computing the tables at the time of the USENIX paper[18] in January 2005.

RF Sniffing

Without a way to either eavesdrop or induce DST's to respond to challenges, the threat to the

[12] The authors claim there “may be” exploitable weaknesses in the cipher itself, but leave the investigation of the claim for future work.

[13] A 3.4 GHz Pentium could only do 200,000 encryptions per second.

[14] At this time (01-Mar-2006) the cost is approximately $138 in single quantity.

[15] The time to search the entire keyspace is ~ 21 hours, so on average assuming random distribution of keys, expect about half that.

[16] Must have been an interesting conversation.

[17] 64-bit memory addressing will be useful here.

[18] I was unable to determine if they have finished yet. If so they may not have reported their result.

DST system would not exist. To complete the circle, the authors then produced a system to mimic the reader's RF side of the protocol[19]. This is far simpler, and requires far less theoretical knowledge and expertise than the earlier steps of cryptanalysis and building the key crackers. It required only writing straight-forward RF modulation and demodulation routines to control the signal fed to and received from the antenna in the TI evaluation kit.

This allows them to actively generate challenge – response pairs between a real DST and their simulated reader, and then recover the key using their previously described key cracker[20]. Once the DST key is recovered, they can masquerade as the DST to readers (purchasing gas or defeating the vehicle immobilizer).

With the full RF and logical protocols they could also straight-forwardly, passively eavesdrop on transactions between legitimate readers and transponders. Those challenge – response pairs can then be taken back to their parallel FPGA cracker for key determination.

Implications

The Texas Instruments DST40 cryptographic security is completely broken[21]. It relied upon "security by obscurity." The length of the cryptographic key was insufficient, and the algorithms used were not well reviewed. This led to the complete cryptanalysis of the device.

The authors point out that "The system architects specified as a design criterion that having access to a transponder or reader for short periods of time should not lead to recovery of the secret key. Their stated aim was to make the DST system resistant to signature-guessing attacks using known challenge-response pairs, cryptanalytic attacks, and exhaustive key search – even for an attacker with full knowledge of the encryption algorithm." It seems clear the aims of the system architects were not achieved in this case. To be fair the paper they cite[22] to support this point was presented in 1996. Moore's Law holds in this situation and the cost of the key recovery has fallen to a few hundred dollars in the intervening 10 years. However, automobiles have a long life, and the cryptographic security should remain effective near the end of the expected vehicle lifetime. In this case the DST40 devices were being used in current production Ford vehicles at the point (January 2005) the cryptographic features were broken.

Owners of 2005 Ford vehicles with immobilizer systems relying on this technology are advised the cryptographic security is effectively broken. The security of the entire system is however only as good as its weakest link. There may still be a weaker link yet at the interface between the

[19] Using an inexpensive 12-bit DAC (Digital to Analog Converter) board as an RF receiver which was also capable of generating the output RF signal using the board's converse Analog to Digital capabilities

[20] There is a link to an amusing QuickTime movie of the sniffing process on the web site for the popularized version of this paper at: http://www.rfid-analysis.org/

[21] Although broken, there has been no move on the part of Texas Instruments, nor the vehicle manufacturers involved to replace the already deployed hardware. http://en.wikipedia.org/wiki/Digital_Signature_Transponder

[22] Gordon, J, Kaiser, U., and Sabetti, T.A. *A low cost transponder for high security vehicle immobilizers*. In 29th ISATA Automotive Symposium (3-6 June 1996). [NB: I did not review this paper. I only reference it to make a point about the approximate age of the DST system]

engine computer (Engine Control Unit / aka ECU), and the transceiver: It is more than likely the interface to the ECU may be defeatable with a mechanic's “cheater” plug which shorts a few signal wires for troubleshooting purposes.

For the Exxon Mobil SpeedPass^{TM}, Exxon Mobil had already implemented (prior to the vulnerability disclosure) other measures to detect and prevent fraudulent use similar to those in effect for the use of credit and debit cards for payment.

The authors suggest the use of a “publicly scrutinized algorithm with an adequate key length, e.g., the Advanced Encryption Standard (AES) in its 128-bit form, or more appropriately, HMAC-SHA1.” In light of our class discussions on SHA1's limited lifetime, perhaps one of the SHA2 forms (maybe SHA256?) would be even more appropriate. For the short term, they suggest shielding of the automotive keys and speed passes with aluminum foil when not in use to guard against active attacks. They point out this is ineffective against passive eavesdropping attacks.

Implications for other cryptographically secured RFID applications

Another application for cryptographically secured RFID chips is identification cards. US Passports (as well as those of many other nations) will contain RFID chips with cryptographic security features in the next several years. The US Congress recently enacted legislation[23] which may eventually force states to produce identity cards such as driver's licenses with RFID. The cracking of the DST40 should serve as a clear warning that the security of those identification credential RFID chips must be both very well engineered and executed.

[23] The “Real ID Act” was enacted, as part of other legislation, by the US Congress, as Public Law 109-13 on 11-May-2005. Although the act does not require RFID technology, it does require that data be stored on the cards with “A common machine-readable technology, with defined minimum data elements.” Certainly cryptographic technology should be applied to this information to protect it from exploitation. A proper implementation of a cryptographically secured RFID could provide the required security.